IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 4, No. 1, March 2015, pp. 14~19
ISSN: 2252-8938

Design and Development of an Agorithm for Prioritizing the Test Cases Using Neural Network as Classifier

Amit Verma, Er. Simranjeet Kaur
Department of Computer Science & Engineering, Chandigarh University, India

Article Info

Article history:

Received Nov 4, 2014
Revised Jan 5, 2015
Accepted Feb 5, 2015

Keyword:

Artificial neural network
Classification
Software testing
Test case prioritization
TF-IDF

ABSTRACT

Test Case Prioritization (TCP) has gained wide spread acceptance as it often results in good quality software free from defects. Due to the increase in rate of faults in software traditional techniques for prioritization results in increased cost and time. Main challenge in TCP is difficulty in manually validate the priorities of different test cases due to large size of test suites and no more emphasis are made to make the TCP process automate. The objective of this paper is to detect the priorities of different test cases using an artificial neural network which helps to predict the correct priorities with the help of back propagation algorithm. In our proposed work one such method is implemented in which priorities are assigned to different test cases based on their frequency. After assigning the priorities ANN predicts whether correct priority is assigned to every test case or not otherwise it generates the interrupt when wrong priority is assigned. In order to classify the different priority test cases classifiers are used. Proposed algorithm is very effective as it reduces the complexity with robust efficiency and makes the process automated to prioritize the test cases.

Corresponding Author:

Er. Simranjeet Kaur,
Department of Computer Science & Engineering,
Chandigarh University, India.
Email: simranjeetgill23@gmail.com

1. INTRODUCTION

Software engineering is the study and application of engineering to the design, development and maintenance of the software. The main aim of software testing is to help the designers, developers to build a system of higher quality free from defects. Software defects are caused due to the inadequacy of software testing. Different tools, algorithms and techniques are needed for efficient and effective testing in order to complete the testing process within limited time and budget constraints. Increase the likelihood of revealing faults related to specific code changes in the regression testing process.The field of neural networks has a history of some five decades but has found application only in the past fifteen years, and the field is still developing continuously. Neural networks are collection of elements. Neural network is able to use some hidden unknown information in the data. This process of capturing hidden information is called learning or training network. We can train a neural network to perform a particular function by adjusting the values of the connections (weights) between elements. Many techniques and technologies have been proposed and evaluated but so none of the technology focused on to assign the priorities based on their frequency in a particular case and to check the accuracy using back propagation algorithm in artificial neural network and to differentiate between the highest and lowest priority using classifiers.

The paper's organization is as follows:

1. Section 2 describes the overview of the literature.
2. Section 3 describes the problem statement.
3. Section 4 describes the proposed algorithm for prioritization.

4. Section 5 describes the result and analysis.
5. Section 6 covers the conclusion and future work

2. LITERATURE SURVEY

G. Rothermel, R. Untch, C.Chu, M.J.Harrold (2001**)** discussed the use of test case prioritization in regression testing and describes various techniques such as based on total coverage of code components and based on fault detection ability. The author concludes that version-specific prioritization produce improvements in fault detection rate. Investigation is done to find the alternative prioritization goals and to increase the effectiveness.

Meenakshi vanmali, Abrahem Kandel (2002**)** presented a new concept of using an artificial neural network as an automated oracle for testing software system. Author represents experimental results using a two layer neural network to detect the faults within mutated code of small credit card approval application. This approach orders the test cases into similar groups instead of ordering them according to their preference degree, so these test cases are classified using MLP neural network.

Erick Cantú-Paz, Chandrika Kamath (2005) presented a comparison of eight combinations of EAs and NNs to 8 classification problems. The author experimented with real and binary encoded EAs to train the networks. EA Algorithms used to train the network, design their architecture and select the feature subsets. Using genetic algorithms to select the feature subset yielded the most accurate classifiers. EA and NN combinations were not accurate than networks trained with simple back propagation.

Dr. Arvinder Kaur and Shivangi Goyal (2011) presented a bee colony optimization algorithm for the fault coverage regression test suite prioritization which is done by studying food foraging behavior of bees. The Effective of this algorithm has been presented using APFD metrics and chart. The challenge faced in this algorithm is the requirement of manual interface for test input data.

Shifa-e-Zehra Haidry, Tim Miller (2013) proposed a new set of techniques for functional test case prioritization based on the inherent structure of dependencies between the tests. The author concludes that test suites prioritized by this technique outperforms the random and untreated test suites but not efficient as greedy test suites.

Nida Gokce, Mubariz Eminli(2014) solved the problem by applying classification approach to the functional relationship between the test case prioritization group membership and author established the important index and frequency for all the events belonging to given group are established. The author improved the new test case model based approach in which where instead of ordering test cases according to their preference degree, they are automatically divided into groups and hence dataset is classified using MLP neural network.

3. CURRENT AREAS TO BE WORK UPON

The problem stated in this paper can be formally stated as:

There is a document having large number of words. We have to prioritize the words according to their frequency in a document and check the accuracy using back propagation algorithm. Differentiate the highest and lowest priority words with the help of classifiers for easy predictions.

Need and Significance:

Various Test case prioritization techniques are implemented and discussed but yet no such algorithm in which test case prioritization is done using neural networks. Artificial neural network helps to correctly assign the priorities and it generates the interrupt when wrong priorities are assigned to different test cases. There is also a need to make the process automate as no efforts are yet made.

4. ALGORITHM

Calculation of TF-IDF is the most important demanding part of the implemented algorithm. For Calculating the TF-IDF value, it is essential to perform tokenization, stop word removal and stemming with the help of appropriate algorithm.Optimization TF-IDF was performed using Java language, in order to calculate the frequency of a particular word in a document. TF-IDF calculates the relative frequency of each word in a document and IDF calculates in formativeness of each word over the entire document. Thus, TF-IDF value of each word is assigned as a weight in this implemented algorithm in which highest frequency word is assigned with highest priority. Different repositories are created to store the words having different priority but having same colour as used to classify them.

Table 1. Psuedo Code of Proposed Algorithm

Algorithm_Pseudo code	Steps
Initialize T_f(T_f ⟶ Text file)	// Intialize the Pdf File
Begin	
For each D_f	
{	
(D_f= D_{tok},;D_f=D_{stem};D_f++)	
}	
Calculate D_f(TF-IDF)	// TF-IDF calculation
Assign $W_{i.....n}$ ⟶ W_d	// Assign Weights to every word
If($W_{d(freq)}$ == highest)	// highest frequency word has highest priority
{	
W_d = = $H_{p;}$	
W_d - - ;	
}	
Else	
{	
W_d = = $L_{p;}$	
W_d - - ;	
}	
Neural Network ⟵ initialize Weights (word , value)	
For (i=1 to n)	
{	
If ($W_{d(Freq)}$ = = $NN_{(Wi)}$)	
{	
Assign correct priority;	
}	
Else	
{	
Back Propagation Error (Re – initialize weights (word,	// Back propagation to generate the interrupt
} value)	
}	
}	
Assign $C_{(Red)}$ ⟶ $Wd_{(Highest)}$	// Classification of highest and lowest priority words by using classifier
Assign (Green) ⟶ $W_{d(Lowest)}$	

5. RESULTS AND ANALYSIS

Figure 5a: After uploading the Pdf file, Tokenization, stop word removal and stemming is done to calculate term frequency of every word in the document in order to assign the weight to each word for prioritization meansTF-IDF of file is calculated.

Figure 5b: Apply Back Propagation algorithm to check whether correct weights are assigned or not. In case priority is different from the actual priority of the word, neural networks back propagates the error to increase the accuracy.

Figure 5c: After assigning priorities to different words, repositories are created to store higher and lower priority test cases for easy prediction.

Figure 5d: After assigning the priorities, highlight them in the PDF file uploaded.

Figure 5e: The proposed technique has higher APFD metric as compared to existing technique.

Figure 5f: The proposed technique has very less execution time.

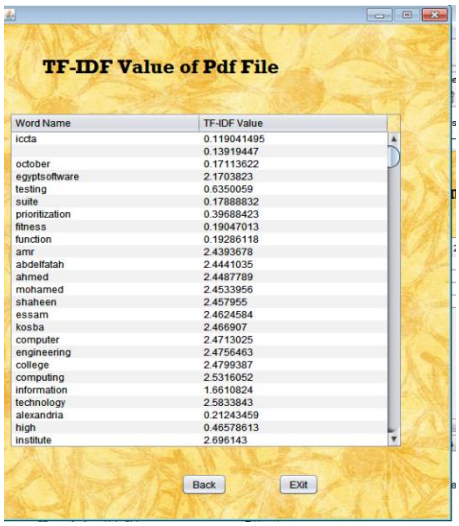

Figure 5a. TF-IDF of PDF File

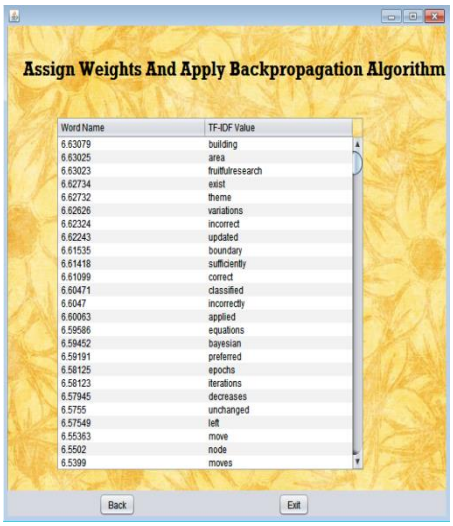

Figure 5b. Back Propagation Algorithm

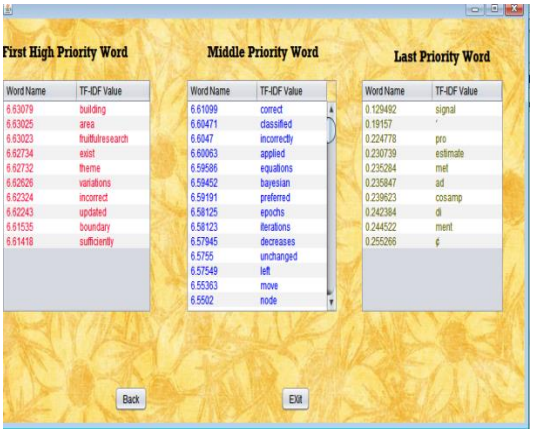

Figure 5c. Classification of Different Priorities

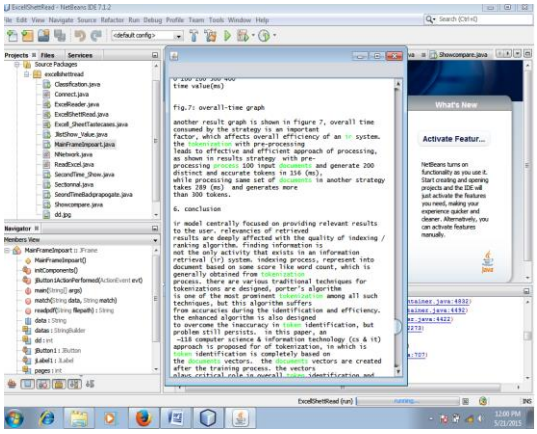

Figure 5d. Highlight the Priorities

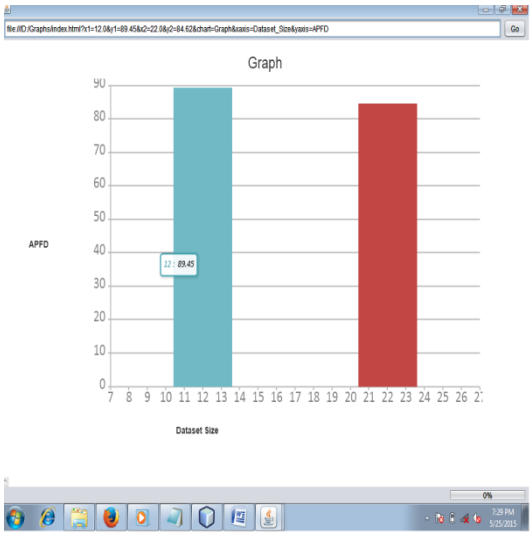

Figure 5e. APFD Metric

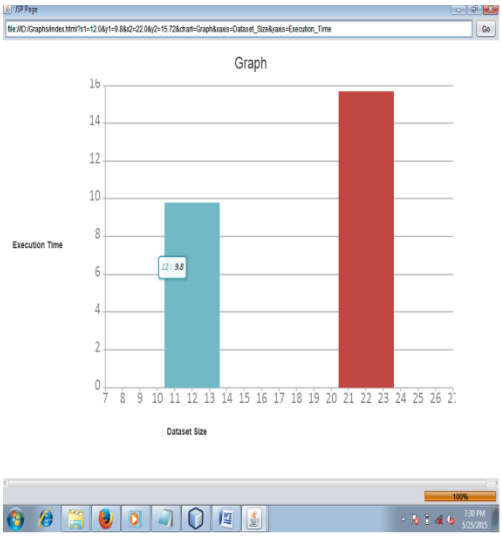

Figure 5f. Execution Time

6. CONCLUSION

In this paper we represent a framework to prioritize the test cases based on TF-IDF, Artificial neural networks back propagation algorithm and naïve bayes classifier algorithm. The framework with embedded methods gave good results, confirmed our concepts and initial expectations. Evaluation of the algorithm is done on various documents. The framework was very stable and reliable. Performed tests have detected a sensitivity of the implemented algorithm. The analysis of the documents content showed that the amount of unusable words in documents has significant impact on the classification so it is necessary to improve the preprocessing of documents to achieve the better results. Classification is very important to easily detect the highest priority and lowest priority word in a document. The implemented algorithm increases the robustness, efficiency and reduces the complexity.

REFERENCES

[1] Bing Cheng, DM Titterington. Neural Networks: A Review from Statistical Perspective. Statistical science. 1994; 9(1): 2-54.

[2] Ken-ichi Funahashi. Multilayer neural networks and Bayes decision theory. *Elsevier Science*. 1997: 209-213.

[3] Guoqiang Peter Zhang. Neural Networks for Classification: A Survey. *IEEE Trans.On Man, Systems, and Cybernetics*. 2000; 30(4).

[4] G Rothermel, R Untch, C Chu, MJ Harrold. Prioritizing Test Cases for Regression Testing. *IEEE Trans. Software Eng*. 2001; 27(10): 929-948.

[5] Meenakshi Vanmali, Mark Last, Abraham Kandel. Using a Neural Network in the Software Testing Process. *International Journal of Intelligent Systems*. 2002; 17: 45-62.

[6] Sebastian Elbaum, Alexey G Malishevsky, G Rothermel. Test Case Prioritization: A Family of Empirical Studies. *IEEE Trans. Software Eng*. 2002; 28(2).

[7] Isabelle Guyon, Jason Weston, Stephen Barnhill. Gene Selection for cancer classification using support vector machine. Machine Learning. Kluwer Academic Publishers. Manufactured in The Netherlands. 2002; 46: 389–422.

[8] James A Jones, Mary Jean Harrold. Test-Suite Reduction and Prioritization for Modified Condition/Decision Coverage. *IEEE transactions on software engineering*. 2003; 29(3): 195-209.

[9] Li Baoli, Yu Shiwen, Lu Qin. An Improved k-Nearest Neighbor Algorithm for Text Categorization. China 2003.

[10] D Richard Kuhn, Senior Member, Dolores R Wallace. Software Fault Interactions and Implications for Software Testing. *IEEE Trans. on software engineering.* 2004; 30(6).

[11] Erick Cantu-Paz, Chandrika Kamath. An Empirical Comparison of Combinations of Evolutionary Algorithms and Neural Networks for Classification Problems. *IEEE Trans. On Systems, Man, and Cybernetics*. 2005.

[12] Zheng Li, Mark Harman, Robert M. Hierons. Search Algorithms for Regression Test Case Prioritization. *IEEE Trans. Software Eng.* 2007; 33(4).

[13] Randall S Sexton, Robert E Dorsey. Reliable Classification Using Neural Networks: A Genetic Algorithm and Back propagation Comparison.

[14] Usha Badhera, GN Purohit, Debarupa Biswas. Test case prioritization algorithm based upon modified code coverage in regression testing. *International Journal of Software Engineering & Applications (IJSEA)*. 2012; 3(6): 29-37.

[15] Stefan Wallin, Leif Land´en. *Telecom Alarm Prioritization using Neural Networks*. 22nd International Conference on Advanced Information Networking and Applications – Workshops. 2008: 1468-1473.

[16] Adam Slowik, Michal Bialko. Training of Artificial Neural Networks Using Differential Evolution Algorithm. *IEEE Krakow, Poland.* 2008: 60-65, May 25-27.

[17] Shi Yu, Steven Van Vooren, Leon-Charles Tranchevent, Bart De Moor, Yves Moreau. Comparison of vocabularies, representations and ranking algorithms for gene prioritization by text mining. 2008; 24, ECCB: 119-125.

[18] Usman Farooq, CP Lam. *Evolving the Quality of Model Based Test Suite*. IEEE International conference on Software Testing verification and validation workshops. 2009.

[19] Sira Vegas, Natalia Juristo, Victor R Basili. Maturing Software Engineering Knowledge through Classifications: A Case Study on Unit Testing Techniques. *IEEE transactions on software engineering*. 2009; 35(4): 551–565.

[20] N Suguna1, K Thanushkodi. An Improved k-Nearest Neighbor Classification Using Genetic Algorithm. *IJCSI International Journal of Computer Science Issues*. 2010; 7(4).

[21] Ajitha, TV Suresh Kumar, D Evangelin Geetha, K Rajnikanth. *Performance Prediction in Early Stages of Software Systems: Artificial Neural Network Model.* ICCCCT. 2010: 743-747.

[22] Improving Testing Efficiency: Agile Test Case Prioritization. Software Benchmarking Organization, 2009-2010: 1-7.

[23] Amit Ganatra, YP Kosta Gaurang Panchal, Chintan Gajjar. Initial Classification Through Back PropagationIn a Neural Network Following OptimizationThrough GA to Evaluate the Fitness of an Algorithm. *International Journal of Computer Science & Information Technology (IJCSIT)*. 2011; 3(1): 98-116.

[24] Ashwin G Raiyani, Sheetal S Pandya. Proritization technique for minimizing number of test cases. *International Journal of Software Engineering Research & Practices*. 2011; 1(1): 3-9.

[25] Arvinder Kaur, Shubhra Goyal. A genetic algorithm for regression test case prioritization using code coverage. *International Journal on Computer Science and Engineering.* 2011; 3(5): 1839-1847.

[26] Sonali Khandai, Arup Abhinna Acharya, Durga Prasad Mohapatra. Prioritizing Test Cases Using Business Criticality Test Value. *International Journal of Advanced Computer Science and Applications*. 2011; 3(5): 103-110.
[27] Zhen-Guo Che, Tzu-An Chiang, Zhen-Hua Che3. Feed-forward neural networks training: a comparison between genetic algorithm and Back-propagation learning algorithm. *International Journal of Innovative Computing, Information and Control*. 2011; 7(10): 5839-5850.
[28] Arvinder Kaur, Shivangi Goyal. A Bee Colony Optimization Algorithm for Fault Coverage Based Regression Test Suite Prioritization. *International Journal of Advanced Science and Technology*. 2011; 29.