NIPES

Journal of Science and Technology Research

Journal homepage: www.nipesjournals.org.ng

Comparative Analysis of Software Components Reusability Level using GFS and ANFIS Soft-Computing Techniques

[1]Ajayi, Olusola O., [2]Chiemeke, Stella C., [3]Ukaoha, Kingsley C.

[1]Dept. of Computer Science, Adekunle Ajasin University Akungba-Akoko, Ondo State, Nigeria.

[2,3]Dept. of Computer Science, University of Benin, Benin City Edo State, Nigeria

Corresponding Emails: olusola.ajayi@aaua.edu.ng, schiemeke@uniben.edu kingsley.ukaoha@uniben.edu

Article Info

***Keywords*:** software component, reusability, soft-computing, adaptive neuro-fuzzy, genetic algorithm, genetic-fuzzy, agile development

Received 29 January 2020
Revised 17 March 2020
Accepted 06 April 2020
Available online 1 June 2020

https://doi.org/10.37933/nipes/2.2.2020.2

https://nipesjournals.org.ng
ISSN-2682-5821/© 2020 NIPES Pub.

Abstract

The quest to develop software of great quality with timely delivery and tested components gave birth to reuse. Component reusability entails the use (re-use) of existing artefacts to improve the quality and functionalities of software. Many approaches have been used by different researchers and applied to different metrics to assess software component reusability level. In addition to the common quality factors used by many authors, such as customisability, interface complexity, portability and understandability, this study introduces and justifies stability, in the context of volatility as a factor that determines the reusability of software components. Sixty-nine software components were collected from third party software vendors and data extracted from their features were used to compute the metric values of the five (5) selected quality factors. Genetic-Fuzzy System (GFS) was used to predict the level of the components' reusability. The performance of the GFS was compared with that of Adaptive Neuro-Fuzzy Inference System (ANFIS) approach using their corresponding average RMSE (Root Mean Square Error), in order to ascertain the level of accuracy of the prediction. The results of the findings showed that, GFS with an RMSE of 0.0019 provides better reusability prediction accuracy compare to ANFIS with an RMSE of 0.1480.

1. Introduction

Reusability is the degree to which a software component can be reused [1, 2, 3]. This consequently leads to reduced software development cost and less development time as it enables less writing but more of assembly. Reusability plays an important role in component based software development (CBSD) and also acts as the basic criterion for evaluating component. [4] asserted that reusability of a component is an important aspect, which gives the assessment to reuse the existing developed component, thereby reducing the risk, cost and time of software development. If a component is not reusable, then the whole concept of component-based software development fails [5]. Reusability is one of the quality attributes of CBSD. It can measure the degree of features/components that are reused in building similar or different new software with minimal change [6]. To realize the reuse of components effectively, reusability estimation has to be carried out. For systematic reuse process, the use of metrics is very germane. Without metrics, evaluating the quality and qualification of the selected components for reuse becomes an uphill task [6]. [7] defined reusability as the quality of any software component to be used again with slight or no modification. Software reuse is the process of creating software systems from existing software assets rather than building them from scratch. Reusability was also viewed as the quality

factor of software that qualifies it to be used again in another application, be it partially modified or completely modified. In other words, software reusability is a measure of the ease with which previously acquired concepts and objects can be used in new contexts. [8] sees reusability of a component as an important aspect, which gives the assessment to reuse the existing developed component. [6] view reusability as a physical replaceable part of a system that adds functionality to the system, through the realization of a set of interfaces. The components having well defined interfaces can be considered good for reuse. The interfaces have strong significance in context of reusability of components.
Metrics however plays indispensable role in the successful evaluation of software component reusability. According to [1], it is necessary to measure reusability of software components in order to realize the effective reuse of such components. According to the author, metrics are used to determine quality factors that affect reusability. A component alone has certain characteristics that tend to affect its reusability. Quality factors are chosen to provide an analysis of the reusability of a component. The choice of factors affecting reusability are considered based on activities carried out while reusing the components.

Unlike in the past where researchers employed statistical methods of predicting reusability [9][1], recent interdisciplinary techniques such as fuzzy logic, Artificial Neural Network (ANN), Neuro-fuzzy etc. have taken the lead due to their power of predictability [10][11][4][7]. This work investigates the works of [10],[11],[4] and [7], who all adopted soft-computing approach to predict reusability of software component, but with varying degree of accuracy

1.1. Related work

Researchers have adopted the use of statistical approaches like correlation analysis, while some made use of soft-computing techniques such as ANN, Fuzzy Logic etc. to evaluate component reusability.
[1] applied statistical method to component reusability assessment issue. Metric suites for measuring reusability of software components were developed. In implementing the work, Component Overall Reusability (COR) model was developed to assess and evaluate Java web components. The study proposed three quality factors as criteria for measuring reusability characteristic, while five metrics were deployed for the measurement. The factors are: understandability, adaptability, and portability, while the metrics include: EMI (Existence of Meta-Information) and RCO (Observability) – for measuring Understandability, RCC (Customisability) – for measuring Adaptability, SCCr (Self-Completeness of Component's return Value) and SCCp (Self-Completeness of Component's Parameter) – for measuring Portability. The result of the analysis conducted using one hundred and twenty-five Java web components from www.jars.com, shows that the proposed metrics were suitable. However, the empirical study was limited to evaluation with Java beans components; as other component technologies like .Net, ActiveX etc were not explored for further validation.
[12] addressed reusability from the perspective of adaptability, compose-ability, and complexity metrics. The work aimed to cover the main aspects of reusable software components, which in their opinion are compose-ability and adaptability. Both factors were evaluated based on the complexity of the component interface. The major contribution of the work, which adopted qualitative approach, was the formulation of metrics and design of a mathematical model for practical assessment of the specified software component characteristics. The proposed model is however, required to be validated by assessing several software components based on it.
[10] contributed largely to software component reusability works by proposing an artificial neural network (ANN) soft-computing-based approach to assess the reusability of software components. The work aimed at aiding developers to select the best component in terms of its reusability. In their research, four factors on which reusability of components depend, were identified. These are:

customisability, interface complexity, portability, and understandability. The empirical work was carried out with forty (40) components collected from www.jars.com and www.elegantjbeans.com. Applying ANN soft-computing approach, network is trained on training data by considering different number of hidden neurons for two training functions namely, *trainlm* and *trainbr*, to get the best results. Results obtained showed that the network was able to predict the reusability of components with optimum performance and with an RMSE of 0.1348 using *trainlm* as the training function. The limitation of the work was in the limited number of data used to train the network. It was submitted that using a greater number of components may produce better results/accuracy for the training and testing.

[11] discussed reusability in relation to Component-Based Development (CBD) and proposes a reusability metrics for black-box components. It identifies factors affecting reusability as: customisability, interface complexity, portability, and document quality. In the study, Fuzzy Logic based approach was used to estimate the reusability of components using Triangular Membership Functions (TMF). The authors used two classroom-based Java beans components, namely Calculator and ChartB for validation. Reusability values of 0.71 and 0.3124 were arrived at proving that FIS (Fuzzy Inference System) is able to predict reusability of components with an acceptable level of accuracy. Further, it was submitted that the adopted approach can be validated against other approaches for estimated reusability of components.

[13] applied Neuro-Fuzzy technique on a case study which they took from a reputed journal. The case study was concerned with the reusability of software components. The reusable components/attributes were coupling, complexity, volume regularity and reuse frequency. They proved that Neuro-Fuzzy model yields less percentage average error as compared to standalone fuzzy logic and neural network. It also produces greater accuracy for software reusability as compared to FIS and ANN.

[14] developed an automated process of component selection by using Adaptive Neuro-Fuzzy interference system (ANFIS) based technique using 14 reusable components' parameters. Neuro-Fuzzy based approach was adopted to select optimal reusable components efficiently. The developed approach was validated with three data sets for three proposed software architectures. The results showed that the proposed approach was able to predict the reusability of these components with an acceptable accuracy. However, stability was used as a fuzzy input with variables such as Low, Medium and High in the ANFIS structure, without reference to porting of the components as suggested in their definition.

[15] proposed a multi-criteria fuzzy-based approach for predicting software requirement stability based on complexity point measurement and for finding out the complexity weight based on requirement complexity attributes such as functional requirement complexity, non-functional requirement complexity, input-output complexity, interface and file complexity. The research paper discussed the importance of measuring the requirements changes for the lack of instability in the requirements. The prediction model for requirements stability approach provides the solution for measuring the requirements changes based on the complexity point measurement model. The work, however, did not justify nor demonstrate the applicability of the model for developing maintenance and transition projects based on different complexity attributes and different adjustment factors.

[4] however adopted multi-disciplinary technique of Adaptive Neuro Fuzzy Inference System (ANFIS) in the assessment of component reusability. In the study, four dependent factors, namely: customisability, interface complexity, understandability, and portability were used to estimate reusability of software components. The result obtained using ANN approach and using data from Sharma et al (2009) was a RMSE of 0.1852. Applying ANFIS approach to the same set of data yielded a RMSE of 0.1695, which shows that Neuro-Fuzzy gives a better and more accurate reusability result. The comparative analysis of the proposed ANFIS and the existing ANN was

carried out on forty-eight (48) Java components. It was however, opined that, accuracy of the used method is subject to availability of substantial number of data/components.

[7] takes into account three different factors for determining reusability of software components and then proposed a model for reusability assessment using the Adaptive Neuro-Fuzzy Inference System (ANFIS). The quality factors used include: coupling, complexity, and portability. The experiment used 338 records retrieved from open source produced a RMSE of 0.042482. It was suggested that, new factors like understandability, cohesion, clarity, generality etc. can also be added, and the cumulative effect of those factors can be seen on the future predictions. Also, different techniques can be used other than ANFIS to predict reusability such as Support Vector Machine, etc. Lastly, it was submitted that, a much better generalised approach is expected if real time data is considered.

[6] identified four attributes for estimating reusability of a black-box components. The reusability metric was parameterised using: component interface complexity, component understandability, component customisability, and component reliability. The project made use of file upload component of the apache commons project. The work proved that the proposed metrics were able to determine reusability. It was however reported that the work requires further validation, suggesting that the weight values for the estimation of reusability be adjusted using neural networks.

2. Methodology

This study adopts:

i. Component-based development approach. This methodology helps to build component analysis tool for accessing common software components;
ii. Metric-based approach. This methodology aids to measure the degree to which a component is reusable;
iii. Soft-computing approach. This methodology predicts the certainty for reusability.

The following procedures were followed in ensuring a successful implementation of the work:

i. Commercial Off-The Shelf Software (COTS) Components were extracted from third party software vendors. According to [2], the key to the success of Component-Based Software Development (CBSD) is its ability to use software components that are often developed by and purchased from third party.

   Component Data Extraction;
   Sixty-nine (69) software components were gotten from four (4) different third-party component development organisations (www.elegantjbeans.com, www.jidesoft.com, www.math.hws.edu, and www.codeproject.com). Table 1 shows the sources, nature and numbers of the components.

Table 1: Components Used

Component Source	Nature of Components	Number of Components	Period of Extraction
www.elegantjbeans.com	Java Components	48	Mar., 2016
www.jidesoft.com	Java Components	4	April, 2016
www.math.hws.edu	Web Components	13	Oct., 2016
www.codeproject.com	.Net Components	4	Nov., 2016

ii. Appropriate metrics for each quality factor that qualifies the characteristic, reusability, were applied. We consider the same quality factors as used by the duo of [10] and [4], with Stability (in the context of volatility) as an addendum.

iii. Genetic-fuzzy soft-computing approach was deployed for evaluating the level of reusability of the selected components. Genetic fuzzy system is a system that exploits genetic algorithms to automatically generate or optimise the knowledge base of a fuzzy system; since the fuzzy system is not able to learn on its own. Researches have shown that hybridised genetic algorithm gives a more accurate predictive result [17][18][19].

2.1 Design

Adapting Kumar *et al*. (2013)'s approach and establishing the need for stability as a factor for component reusability measurement,

Let $R_{cn} = F_{cn}[X_n, Y_n, Z_n, J_n, K_n]$ (1)

Where:

R_{cn} is the reusability of component.

F_{cn} is implemented using Genetic-Fuzzy with X_n, Y_n, Z_n, J_n, and K_n as input dependent variables, representing Customisability (COCU), Interface Complexity (COIC), Portability (CORE), Understandability (COUS) and Stability (COST) respectively.

In the proposed model, Genetic-Fuzzy System is developed, trained and tested using MATLAB software. The steps involved in the development of the system are:

i. Extract component data
ii. Compute the metric value of X_n, Y_n, Z_n, J_n, and K_n
iii. Represent the variables in Fuzzy format
iv. Load values of X_n, Y_n, Z_n, J_n, and K_n into Fuzzy toolbox
v. Apply the Genetic Optimiser to tune the knowledge base
vi. Compute the fitness value until the threshold/termination is reached

```
Algorithm (ANFIS)

    Select Loader
    If loader = ANFIS, load cipus-run.m
        browse to retrieve training data
        load training data
        if fileext = '*.csv', 'load successful'
        else 'load unsuccessful', reload
    endif
    browse to retrieve testing data
    load training data
        if fileext = '*.csv', 'load successful'
        else 'load unsuccessful', reload
        endif
    End Select
    RUN Reusability RMSE
    VIEW Reusability RMSE

Algorithm (GFS)
    Select Loader
    If loader = GFS, load myga.m
```

```
        load fuzzy-excel formatted file (loaddata.m)
if fileext = '*.csv', 'load successful'
else 'load unsuccessful', reload
                endif
        load/call/invoke ga_fitfunc.m
                if load_status = 'correct', proceed
else re-load/re-call/re-invoke fitness function
                endif
        End Select
        RUN Reusability RMSE
        VIEW Reusability RMSE
```

2.2. Experimental Evaluation

2.2.1 The FIS Properties

Table 3 presents the details/structure of the Fuzzy Inference System design properties.

Table 3: FIS Structure/Properties

Parameter	**FIS Names (s)**	**Property Default/Range Value/Parameter Range**
Input Parameter 1	COCU	[0 1]
Input Parameter 2	COIC	[0 1]
Input Parameter 3	CORE	[0 1]
Input Parameter 4	COUS	[0 1]
Input Parameter 5	COST	[0 1]
Input FIS Type:		Sugeno
MF Type:		Triangular
Output Name:		Reusability
Output Type:		Linear
Input Parameters:		Low [0 0.25 0.5] Medium [0.25 0.5 0.75] High [0.5 0.75 1]
		Low [0 0.25 0.5] Medium [0.25 0.5 0.75] High [0.5 0.75 1]
		Low [0 0.25 0.5] Medium [0.25 0.5 0.75] High [0.5 0.75 1]
		Low [0 0.25 0.5] Medium [0.25 0.5 0.75 0] High [0.5 0.75 1 0]
		Low [0 0.25 0.5 0] Medium [0.25 0.5 0.75 0] High [0.5 0.75 1 0]
Output Parameters:		Low [0 0 0 0 0 0] Medium [0.5 0.5 0.5 0.5 0.5 0.5] High [1 1 1 1 1 1]

2.2.2 The ANFIS Evaluation Parameters

Table 4 shows the specifications of the ANFIS evaluation parameters.

Table 4: ANFIS Specifications

PARAMETERS	Main Attribute	Others
Testing Data	20 data	29% of the entire data used
Training Data	49 data	71% of the entire data used
No of Epoch	50	
Error Tolerance	0	
Rules	243	
Logical Operator	AND	
Inputs	5	C.I.P.U.S (Customizability, Interface complexity, Portability, Understandability, Stability)
Input MF	3	Low, Medium and High
Output	1	Reusability
Output MF	3	Low, Medium and High
Optimisation Method	Hybrid	

2.2.3 The GA Optimisation Parameters and Algorithm

Table 5 shows the specifications of the parameters used for the GA.

Table 5: GA Specifications

PARAMETERS	Main Attribute	Others
Data	loaddata.m (matlab file)	x = csvread('data.csv')
Fitness Function	ga_fitfunc (matlab function)	y = (x(1)+x(2)+x(3)+x(4)+x(5))/5
Population	Randomized	Constraint Dependent
Bounds	Lower: [0 0 0 0 0] Upper: [1 1 1 1 1]	
Selection	Tournament	Size: 4
Mutation	Adaptive Feasible	
Crossover	Two-point (Double)	
Stopping Criteria	Generation	
Fitness Scaling	Scaling Function	Rank

2.2.4 Statistical Representation and Comparative Analysis

Appendix A shows the RMSE values of the two approaches (ANFIS and GFS) for the selected components.

Figure 11 represent the comparative chart for the ANFIS and GFS's RMSE in which GFS proved to have lower RMSEs (0.0019), implying better predictor.

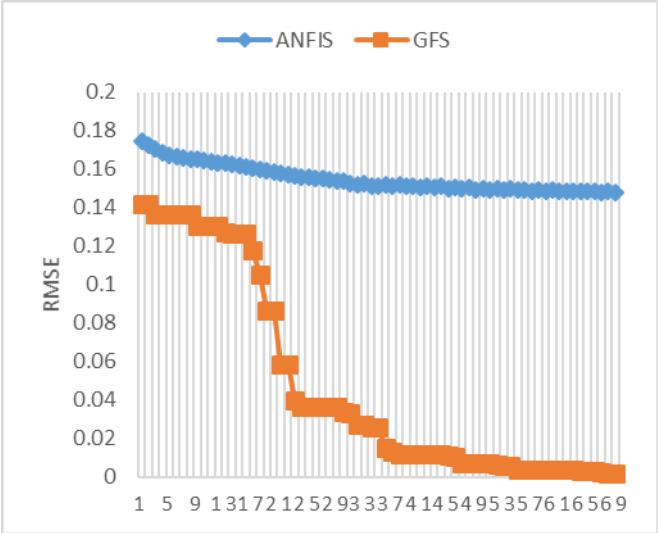

Figure 11: ANFIS and GFS' RMSE

Table 6 shows the aggregate values of Table A1 in Appendix for the three components selected and for the five quality factors in use.

Table 6: Computed Aggregate Values of Component Types

COMPONENT TYPES	COCU	COIC	CORE	COUS	COST
Java	0.88	0.86	0.48	0.92	0.79
Web	0.66	0.27	2.08	0.73	0.69
.Net	0.79	0.47	2	0.79	0.75

Analysing with SPSS, and using ANOVA (Analysis of Variance), the result is shown in Table 7

Table 7: ANOVA analysis of Component Types' Aggregated Values

Component Types	N	Mean	Std. Deviation	Std. Error
Java	5	.7860	.17743	.07935
Web	5	.8860	.69263	.30975
.Net	5	.9600	.59657	.26680
Total	15	.8773	.50318	.12992

From Table 8, Java Components proved more reusable as it recorded the least standard error (0.07935) compare to .Net Component's 0.26680 and Web Component's 0.30975. Figure 12 shows the reusability prediction level of the various software components used.

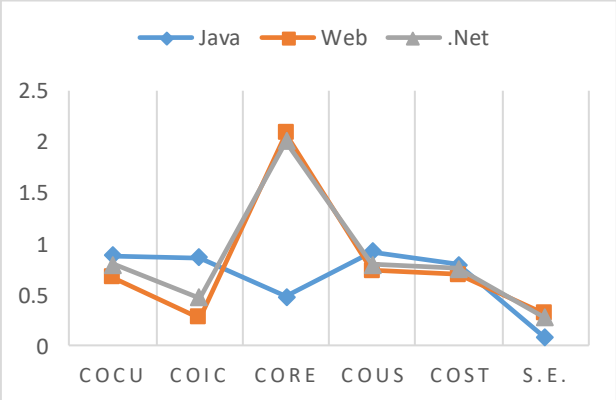

Figure 12: Components' Reusability Prediction Level

3. Finding

The finding from the study show that, GFS with an RMSE of 0.0019 provides better reusability prediction accuracy compared to ANFIS with an RMSE of 0.1480. The experiments conducted showed that Java Components, with a S.E. of 0.07935 proved more reusable compare to Web Component's S.E. of 0.30975; and .Net Component's S.E. of 0.26680.

4. Conclusion

The study established a Genetic-Fuzzy System (GFS) for the evaluation of software component reusability, with the results proving the new system a better predictor than the most commonly used system (ANFIS). Software component assessment with other component types other than Java components. With researches showing that most studies on reusability of software components were done experimenting only with Java Components, this study was able to carry out its assessment of component reusability using Java, Web and .Net Components. The research took a leap to evaluate the level of reusability of each component, with Java Components proving more reusable than the rest two component types. The study therefore contributed to the increasing body of knowledge that Java Components are more reusable than other component types.

The practice of software component reusability no doubt aid software development cost and time. However, of greater necessity is the issue of measuring to ascertain the level of reusability of the selected software components for reusability. This, many researchers agreed with and deployed different evaluation techniques in assessing the level of reusability of software components.

Consequently, this work presented a comparative analysis of software components reusability using genetic-fuzzy system and adaptative neuro-fuzzy inference system. The study utilised five quality factors in measuring the reusability of sixty-nine (69) software components.

The result of the evaluation carried out shows genetic-fuzzy system predicts more accurately with an RMSE of 0.0019 as against the commonly used method, ANFIS, with an RMSE of 0.1480, adjudging genetic-fuzzy system as a better predictor.

4.1 Direction for further studies

Five quality factors were used in the determination of the reusability of the selected components, other quality factors as related to software components (e.g. operability, statelessness etc.), can also be considered in future research work in the prediction of software component reusability.

References

[1] Washizaki, H., Yamamoto, H., and Fukazawa, Y. (2003). A metrics suite for measuring reusability of software components. Proceedings of the 9th International Symposium on Software Metrics. Sept 3-5, Sydney, Australia, pp. 201-211

[2] Sharma, A., Kumar R., Grover P. S. (2006). Investigation of reusability, complexity and customisability for component-based systems", ICFAI Journal of IT, 2(1).

[3] Fazal-e- Amin, Mahmood, A. K., and Oxley, A. (2011). A Review of Software Component Reusability Assessment Approaches. *Research Journal of Information Technology*, 3(1):1-11.
[4] Kumar, V., Kumar, R., and Sharma, A. (2013). Applying Neuro-Fuzzy Approach to build the Reusability Assessment Framework across Software Component Releases – An Empirical Evaluation. International Journal of Computer Applications. 70(15): 41-47
[5] Thakral, S., Sagar, S., and Vinay (2014). Reusability in Component Based Software Development – A Review. World Applied Sciences Journal. 31(12):2068-2072.
[6] Singh, A. P. and Tomar, P. (2014). Estimation of Component Reusability through Reusability Metrics. International Journal of Computer, Control, Quantum and Information Engineering. 8(11):1865-1872
[7] Goel, S., and Sharma, A. (2014). Neuro-Fuzzy based Approach to Predict Component's Reusability. International Journal of Computer Applications, 106(5)
[8] Kumar, A., Chaudhary, D., and Kumar, A. (2014). Empirical Evaluation of Software Component Metrics. International Journal of Scientific and Engineering Research. 5(5):814-820
[9] Aman, H. (2002). A Quantitative Method of Verifying Metrics Using Principal Component Analysis and Correlation Analysis. Journal of IEICE. J85-D-1(10): 1000-1002
[10] Sharma, A., Kumar, R. and Grover, P. S. (2009). Reusability assessment for software components. ACM SIGSOFT Software Engineering Notes. 34(2):1-6.
[11] Sagar, S., Nerurkar, N.W., and Sharma, A. (2010). A soft computing based approach to estimate reusability of software components. ACM SIGSOFT Software Engineering Notes, 35(4):1-5
[12] Rotaru, O. P. and Dobre, M. (2005). Reusability Metrics for Software Components. AICCSA '05 Proceedings of the ACS/IEEE 2005 International Conference on Computer Systems and Applications. Pgs. 24-I. Washington, USA.
[13] Singh, H. and Toora, V. K. (2011). Neuro-Fuzzy Logic Model for Component Based Software Engineering. International Journal of Engineering.
[14] Ravichandran, K., Suresh, P., and Sekr, K. R. (2012). ANFIS Approach for Optimal Selection of Reusable Components. Research Journal of Applied Sciences, Engineering and Technology, 4(24): 5304-5312
[15] Christopher, D., and Chandra, E. (2012).Prediction of software requirements stability based on complexity point measurement using multi-criteria fuzzy approach, International Journal of Software Engineering & Applications (IJSEA), Vol.3, No.6, November 2012.
[16] Sandhu, P. S., Dalwinder, S. S., and Singh, H. (2008). A Comparative Analysis of Fuzzy, Neuro-Fuzzy and Fuzzy-GA Based Approaches for Software Reusability Evaluation. Proceedings of World Academy of Science, Engineering and Technology (WASET). 2:292-295
[17] Hegazy, O., Soliman, O. S., Toony, A. A. (2014). Hybrid of neuro-fuzzy inference system and quantum genetic algorithm for prediction in stock market. Issues in Business Management and Economics. 2(6):094-102. www.journalissues.org/IBME accessed on 13/10/2017
[18] Fazlic, L. B., Avdagic, K., Omanovic, S. (2015). GA-ANFIS Expert System Prototype forPrediction of Dermatological Diseases. European Federation for Medical Informatics (EFMI). Pgs. 622-626

[19] Dhokley, W., Ansari, T., Fazlic, N., Mohd.Hafeez, H. (2016). New Improved Genetic Algorithm for Coronary Heart Disease Prediction. International Journal of Computer Applications 136(5): 0975-8887

Appendix A

Table A1: Components' RMSE Values for ANFIS and GFS

Component Type	RMSE (ANFIS)	RMSE (GFS)
Java Components	0.1741	0.142
Java Components	0.1727	0.142
Java Components	0.1703	0.1367
Java Components	0.1687	0.1367
Java Components	0.1674	0.1367
Java Components	0.1665	0.1367
Java Components	0.1656	0.1367
Java Components	0.1652	0.1367
Java Components	0.1648	0.1305

Java Components	0.1644	0.1305
Java Components	0.1639	0.1305
Java Components	0.1633	0.1302
Java Components	0.1628	0.1273
Java Components	0.1623	0.1263
Java Components	0.1617	0.1263
Java Components	0.1611	0.1263
Java Components	0.1605	0.1177
Java Components	0.1599	0.1052
Java Components	0.1592	0.08672
Java Components	0.1585	0.08672
Java Components	0.1578	0.05859
Java Components	0.1571	0.05859
Java Components	0.1563	0.03984
Java Components	0.1558	0.03672
Java Components	0.1556	0.03672
Java Components	0.1551	0.03672
Java Components	0.1549	0.03672
Java Components	0.1543	0.03672
Java Components	0.154	0.03672
Java Components	0.1535	0.03359
Java Components	0.1525	0.03325
Java Components	0.1515	0.02754
Java Components	0.1523	0.02754
Java Components	0.1513	0.02583
Java Components	0.1513	0.02583
Java Components	0.1515	0.01489
Java Components	0.1512	0.01333
Java Components	0.1515	0.01191
Java Components	0.1511	0.01191
Java Components	0.1514	0.01191
Java Components	0.1507	0.01191
Java Components	0.1513	0.01191
Java Components	0.1505	0.01191
Java Components	0.1508	0.01191
Java Components	0.1499	0.01141
Java Components	0.1507	0.01025
Java Components	0.1497	0.007224
Java Components	0.1506	0.007224
Java Components	0.1494	0.007224
Java Components	0.15	0.007224
Java Components	0.1492	0.007224
Java Components	0.1497	0.006442
Web Components	0.1489	0.005661
Web Components	0.1496	0.005661
Web Components	0.1488	0.004099

Web Components	0.1493	0.004099
Web Components	0.1485	0.004099
Web Components	0.1492	0.003925
Web Components	0.1485	0.003925
Web Components	0.149	0.003925
Web Components	0.1484	0.003925
Web Components	0.1487	0.003832
Web Components	0.1483	0.00368
Web Components	0.1485	0.003362
Web Components	0.1481	0.003362
.Net Components	0.1483	0.003362
.Net Components	0.148	0.00266
.Net Components	0.1482	0.001879
.Net Components	0.1479	0.001879