International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 12– No.21, June 2019– www.ijais.org

A Neuro-Fuzzy Model for Predicting Students Performance in Object-Oriented Programming Courses

Olusola Olajide Ajayi
Department of Computer Science, Faculty of Science, Adekunle Ajasin University, Akungba Akoko, Ondo State, Nigeria

Temitope Clement Akindele
Department of Computer Science, Faculty of Science, Adekunle Ajasin University, Akungba Akoko, Ondo State, Nigeria

ABSTRACT
Failure trend in object-oriented programming courses is mostly on the increase side, student's performance in other courses are most times better than in programming courses. One of the ways to improve the student performance is for the instructors to identify the group of students who might not perform well at the later stage of learning. From there the instructor can focus on the students in order to help them to improve their performance. Thus, in this case making the prediction of student performance a major step in identifying the potential students that needs further help such as extra classes or special tutorials and assignments. Therefore, the need for performance prediction in programming courses becomes imperative. The study will use neuro-fuzzy model to evaluate the current performance of students and then predict the students' performances in subsequent object oriented programming courses.

Keywords
Object-oriented, student performance, programming courses, fuzzy-inference, ANFIS, model

1. INTRODUCTION
The high failure rates in many programming courses means there is need to identify struggling students as early as possible. To achieve this, predicting future student performance in programming courses becomes more important. Predicting a student's performance in programming courses has been a well-studied problem, and over the past fifty years, various predictors have been proposed. Early work mainly used standardized aptitude tests to predict performance (Christopher et al, 2013).

As programming became more widespread, researchers began to explore a greater range of cognitive, psychological, and demographic predictors. Researchers over the past two decades extended prior work by exploring similar factors and the predictive potential of new innovations in pedagogy. Most times, the major factor that do influence student failure in programming courses is the performance, character and attitude of teachers or lecturers that do take students in first years programming course or elementary programming courses or that of student.

However, a limitation of studies to date is their tendency to use lengthy tests that often yield inconsistent results. Given potentially high enrolment numbers, the use of tests to gather predictive data can take a considerable amount of time for an instructor to process. Even if a test was indicative of performance, by the time it was processed, it may be too late for students to withdraw, or for instructors to intervene to prevent students from failing. Object-Oriented Programming (OOP) is a programming language model organized around objects rather than "actions" and data rather than logic. Historically, a program has been viewed as a logical procedure that takes input data, processes it, and produces output data. It is a programming paradigm that based on the concept of "*objects*", which may contain data, in the form of fields, often known as attributes; and code, in the form of procedures, often known as methods.

There are many Object-Oriented Programming Languages, e.g. Java, C++, C#, Python, PHP, Ruby, Perl, Delphi, Objective-C, Swift, Common Lisp, and Smalltalk etc. Object-Oriented Programming has many outstanding features. Some of which are:
Object: Object is a collection of number of entities. Objects take up space in the memory. Objects are instances of classes. When a program is executed, the objects interact by sending messages to one another. Each object contains data and code to manipulate the data. Objects can interact without having known details of each other's data or code.
Class: Class is a collection of objects of similar type. Objects are variables of the type class. Once a class has been defined, we can create any number of objects belonging to that class, e.g. grapes bananas and orange are the member of class fruit.
Data Abstraction and Encapsulation: Combining data and functions into a single unit called class and the process is known as Encapsulation. Data encapsulation is important feature of a class. Class contains both data and functions. Data is not accessible from the outside world and only those function which are present in the class can access the data. The insulation of the data from direct access by the program is called data hiding or information hiding. Hiding the complexity of program is called Abstraction and only essential features are represented. In short we can say that internal working is hidden.
Dynamic Binding: Refers to linking of function call with function definition is called binding and when it is take place at run time called dynamic binding.
Message Passing: The process by which one object can interact with other object is called message passing.
Inheritance: it is the process by which object of one class acquire the properties or features of objects of another class. The concept of inheritance provide the idea of reusability means we can add additional features to an existing class without modifying it. This is possible by driving a new class from the existing one. The new class will have the combined features of both the classes.
Polymorphism: A Greek term means ability to take more than one form. An operation may exhibit different behaviours

in different instances. The behaviour depends upon the types of data used in the operation.
Example: Operator Overloading, Function Overloading.
Object-oriented programming (OOP) is one of the core courses in computer science and technology, which is also one of the most important specialty courses for science and engineering university students (Anquan et al, 2010).

Mohd et al, 2012, in his study "Predicting Student Performance in Object Oriented Programming using Decision Tree: A Case study of KTPM, Kuantan". Identify and implement the most accurate algorithm, and the most relevant rules were identified from the model.

Due to the unique nature of the different available object-oriented languages, many of which are deployed in the academic setting for teaching, it is very imperative to study the trend of students' performance in subsequent object-oriented consequences in other to further critically examined the indices of students' performance. To this end, this study will apply Adaptive Neuro-Fuzzy approach in assessing and predicting trends of students' performance in object-oriented programming courses. The mapping then provides a basis from which decisions can be made, or patterns discerned.

2. BACKGROUND

Fuzzy Logic is a technique that facilitates the control of a complicated system without knowledge of its mathematical description. Fuzzy logic differs from classical logic in that statements are no longer black or white, true or false, on or off. In traditional logic an object takes on a value of either zero or one. In fuzzy logic, a statement can assume any real value between 0 and 1, representing the degree to which an element belongs to a given set.
Fuzzy system handles imprecise and uncertain information using the theory of fuzzy sets, (Han et al, 2006). The output of a fuzzy system is represented by an output fuzzy set. Fuzzy inference is the process of formulating the mapping from a given input to an output using fuzzy logic. The mapping then provides a basis from which decisions can be made.

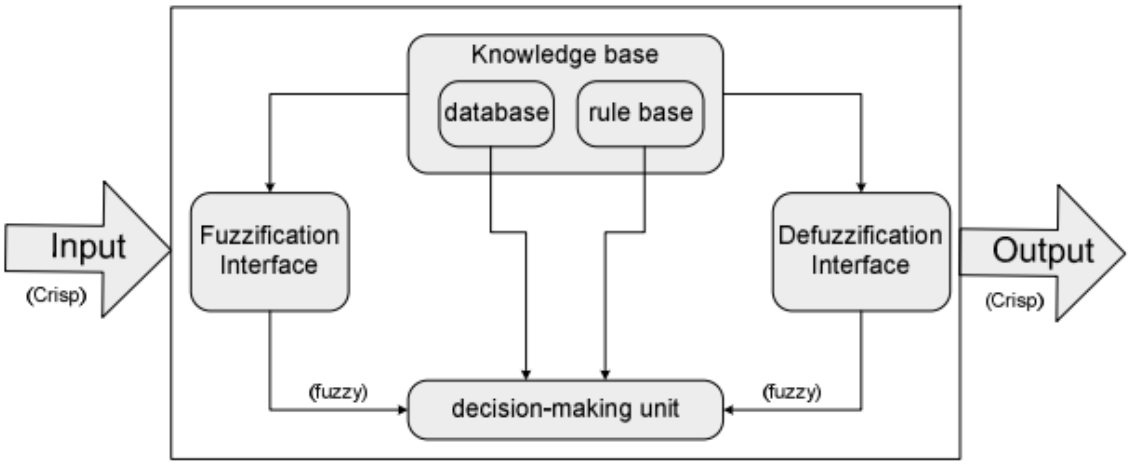

Figure 1: Fuzzy Inference System Architecture
Source: Kamyar, 2008

There are two major types of the Fuzzy models. The Takagi-Sugeno Fuzzy Model and the Mamdani Fuzzy Model.
Takagi-Sugeno Fuzzy Model: This fuzzy model proposed by Takagi and Sugeno (Takagi et al, 1985) is described by fuzzy IF-THEN rules which represents local input-output relations of a nonlinear system. The main feature of a Takagi-Sugeno fuzzy model is to express the local dynamics of each fuzzy implication (rule) by a linear system model. The overall fuzzy model of the system is achieved by fuzzy "blending" of the linear system models. In this tutorial, the reader will find, by some examples, that almost all nonlinear dynamical systems can be represented by Takagi-Sugeno fuzzy models to high degree of precision (Kamyar, 2008). In fact, it is proved that Takagi-Sugeno fuzzy models are universal approximators of any smooth nonlinear system (Fantuzzi et al, 1996).
Mamdani Fuzzy Model: is widely known and used in developing fuzzy models. It consists of rules of the form "IF (X1 is A1) AND (X2 is B1) AND (X3 is C1) THEN Y is F ", where X1, X2, and X3 are inputs, Y is output, then A1, B1, C1, and F are linguistic terms with MFs Triangular, Trapezoidal, and Gaussian. That represents the premise and consequent parts of the rule base. The implication is applied for each rule, generally min operator representing the (AND) and (OR) logic is used for implication. Aggregation is used to unify the output of all the rules resulting in a single fuzzy set (Sirwan et al, 2013). The aggregated output function is defuzzified in a single crisp number using a defuzzification method (Hamam et al, 2008).
A neural network is a powerful computational data model that is able to capture and represent complex input/output relationships. The motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain (Xinghuo et al, 2001). Neural networks resemble the human brain in the following two ways:

1. A neural network acquires knowledge through learning.
2. A neural network's knowledge is stored within inter-neuron connection strengths known as synaptic weights.

The true power and advantage of neural networks lies in their ability to represent both linear and non-linear relationships and in their ability to learn these relationships directly from the data being modelled. Traditional linear models are simply

inadequate when it comes to modelling data that contains non-linear characteristics. (Santhosh et al, 2011).
An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information (Mohsen et al, 2007). The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. (Santhosh et al, 2011).

A neural network model is a structure that can be adjusted to produce a mapping from a given set of data to features of or relationships among the data. The model is adjusted, or trained, using a collection of data from a given source as input, typically referred to as the training set. After successful training, the neural network will be able to perform classification, estimation, prediction, or simulation on new data from the same or similar sources.

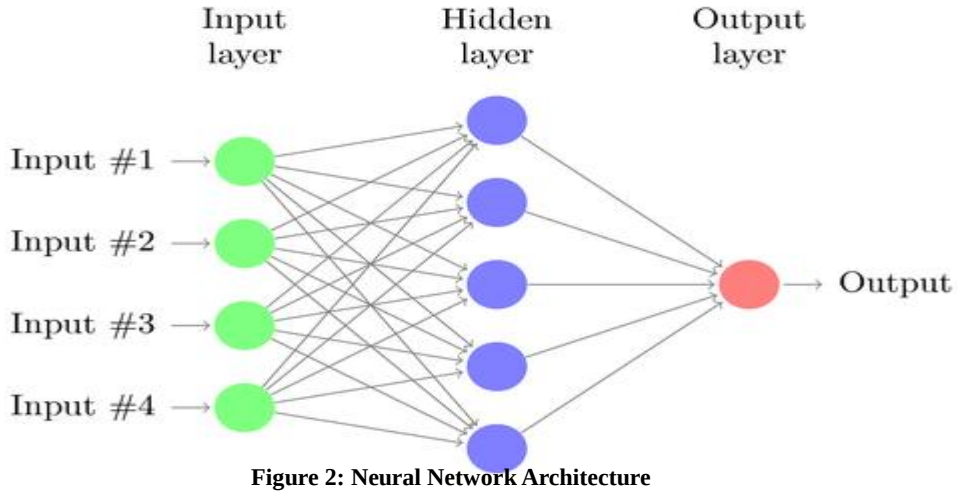

Figure 2: Neural Network Architecture
Source: texample.net

THE SYNERGY

Since the moment that fuzzy systems become popular in industrial application, the community perceived that the development of a fuzzy system with good performance is not an easy task (Jose, V. et al, 2004). The problem of finding membership functions and appropriate rules is frequently a tiring process of attempt and error. This lead to the idea of applying learning algorithms to the fuzzy systems. The neural networks, that have efficient learning algorithms, had been presented as an alternative to automate or to support the development of tuning fuzzy systems.

Neural networks and fuzzy systems can be combined to join its advantages and to cure its individual illness. Neural networks introduce its computational characteristics of learning in the fuzzy systems and receive from them the interpretation and clarity of systems representation. Thus, the disadvantages of the fuzzy systems are compensated by the capacities of the neural networks. These techniques are complementary, which justifies its use together.

In general, all the combinations of techniques based on neural networks and fuzzy logic can be called neuro-fuzzy systems. According to Nauk et al, 1997, the different combinations of these techniques can be divided, into the following classes:
Cooperative Neuro-Fuzzy System: In the cooperative systems there is a pre-processing phase where the neural networks mechanisms of learning determine some sub-blocks of the fuzzy system. For instance, the fuzzy sets and/or fuzzy rules, that is, fuzzy associative memories or the use of clustering algorithms to determine the rules and fuzzy sets position (Czogala et al, 2000). After the fuzzy sub-blocks are calculated the neural network learning methods are taken away, executing only the fuzzy system.

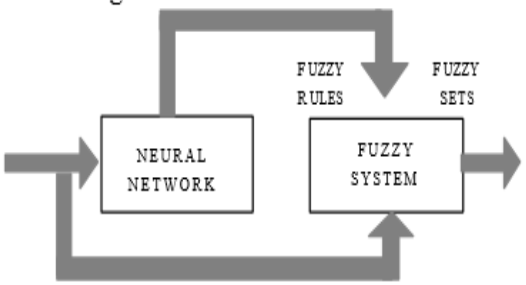

Figure 2: The Cooperative NF System
Source: Jose, V. et al, 2004

Concurrent Neuro-Fuzzy System: In the concurrent systems the neural network and the fuzzy system work continuously together. In general, the neural networks pre-processes the inputs (or post-processes the outputs) of the fuzzy system.

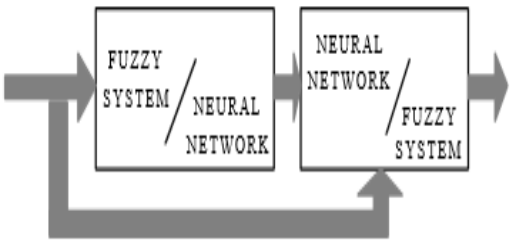

Figure 3: The Concurrent NF System
Source: Jose, V. et al, 2004

Hybrid Neuro-Fuzzy System: In this category, a neural network is used to learn some parameters of the fuzzy system (parameters of the fuzzy sets, fuzzy rules and weights of the rules) of a fuzzy system in an iterative way. The majority of the researchers uses the neuro-fuzzy term to refer only hybrid neuro-fuzzy system.

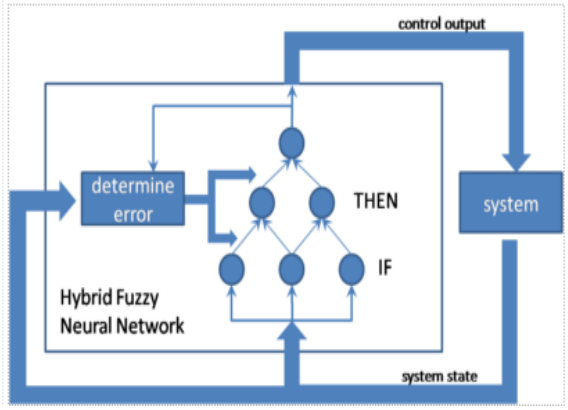

Figure 4: The Hybrid NF System
Source: Scholarpedia.com

While fuzzy logic provides an inference mechanism under cognitive uncertainty, computational neural networks offer exciting advantages, such as learning, adaptation, fault tolerance, parallelism and generalization (Robert, 2001).

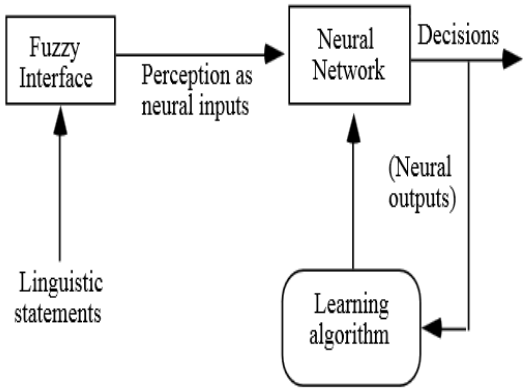

Figure 5: The Neuro-Fuzzy System Model
Source: Robert, 2001

3. RESEARCH PROBLEM

Many researchers have shown that, the rate at which students fail in Object Oriented Programming Course is high, as a result, it has become imperative to come up with model for predicting student performance. Predicting a student's performance in programming courses has been a well-studied problem. Different researchers have use different methodology and techniques but accuracy plays an important role to prove the best predictive method. Therefore, this study will use Neuro-Fuzzy model to predict student performance in Object Oriented Programming courses.

4. RESEARCH GOAL

This study will aim to design models and find a suitable methodology to predict the performance of students in object-oriented programming courses. The students' performance trend in programming courses will be analyzed using existing students results in object-oriented courses as data. The assessment will be used to forecast their likely future performance in object-oriented programming courses.

5. RESEARCH METHODOLOGY

The outlined procedures were taken in order to meet up with the goal of this study:
Step 1: Extensive literature search were made to ascertain level and trend of related work done on the subject matter
Step 2: Data collection of students score (in MS-Excel format) in some selected objected oriented programming courses (CSC212 – C++, 2013/2014 Academic Session; and CSC305 – Java, 2014/2015 Academic Session; with a total of Ninety-Two (92) Students) from the Department of Computer Science, Faculty of Science, Adekunle Ajasin University, Akungba-Akoko. The data were filtered to contain only regular students (COMPUTER MAJOR and COMPUTER EDUCATION).
Step 3: Analysing the collated data using Neuro-Fuzzy Model in MATLAB
Step 4: Interpretation of the results of the data analysed.

6. DESIGN MODEL

6.1 System Architecture

A system architecture or systems architecture is the conceptual model that defines the structure, behaviour, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviours of the system. Systems architecture provides a framework for designing changeable systems even when these systems do not span the entire enterprise (Somerville, 2011). The need for systems to evolve over time and the requirement for ongoing enhancements and maintenance create growing complexity in information systems. A good systems architecture provides a framework for change, and change is one of the few constants in today's software business (Chapel et al, 2013).

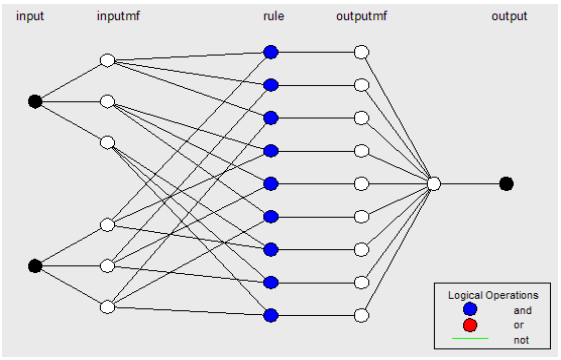

Figure 6: System Architecture

6.2 IMPLEMENTATION

The study made use of two variables for the input, OOP1performance (C++) and OOP2performance (Java); while PredictPerformance represents the output variable. Their specifications and data were coded into MATLAB for the experimentation. For efficiency and suitability purposes, TSK FIS type was applied (Figure 7). The input variables

have three input linguistic variables LowPerformance (0,0.25,0.50), FairPerformance (0.25,0.50,0.75) and HighPerformance (0.50,0.75,1.0) (Figures 8-9), while the output variable has two variables as its output linguistic, namely LikelyPerformPoorly (0,0.25,0.50), and LikelyPerformWell (0.5,0.75,1) (Figure 10). Having two (2) input variables and three (3) linguistic values, we derived 3^2 rules (9 rules), generated thus:

If OOP1performance is LowPerformance AND OOP2performance is LowPerformance
Then PredictPerformance is LikelyPerformPoorly
If OOP1performance is LowPerformance AND OOP2performance is FairPerformance
Then PredictPerformance is LikelyPerformPoorly
If OOP1performance is LowPerformance AND OOP2performance is HighPerformance
Then PredictPerformance is LikelyPerformWell
If OOP1performance is FairPerformance AND OOP2performance is LowPerformance
Then PredictPerformance is LikelyPerformPoorly
If OOP1performance is FairPerformance AND OOP2performance is FairPerformance
Then PredictPerformance is LikelyPerformWell
If OOP1performance is FairPerformance AND OOP2performance is HighPerformance
Then PredictPerformance is LikelyPerformWell
If OOP1performance is HighPerformance AND OOP2performance is LowPerformance
Then PredictPerformance is LikelyPerformPoorly
If OOP1performance is HighPerformance AND OOP2performance is FairPerformance
Then PredictPerformance is LikelyPerformWell
If OOP1performance is HighPerformance AND OOP2performance is HighPerformance
Then PredictPerformance is LikelyPerformWell

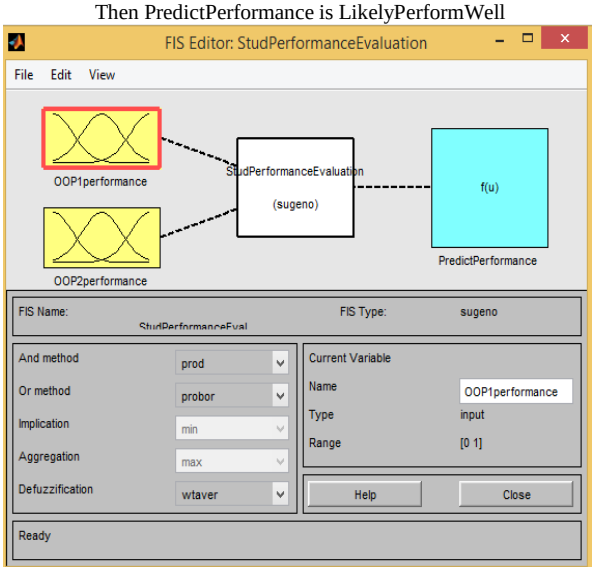

Figure 7: Variables Specifications

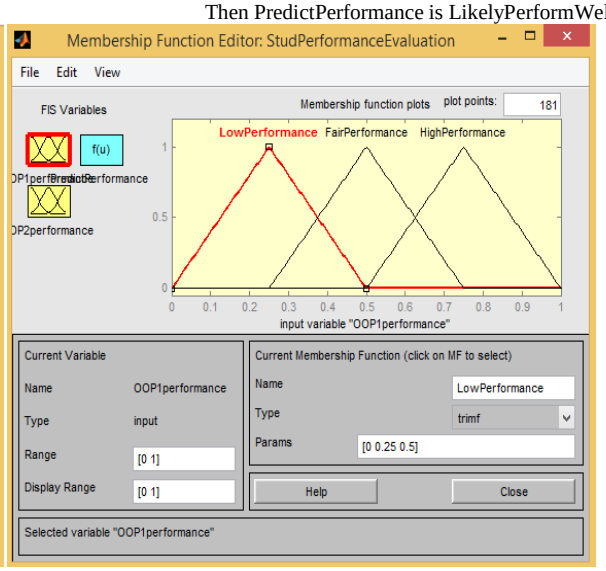

Figure 8: Input Membership Specification (OOP1Performance)

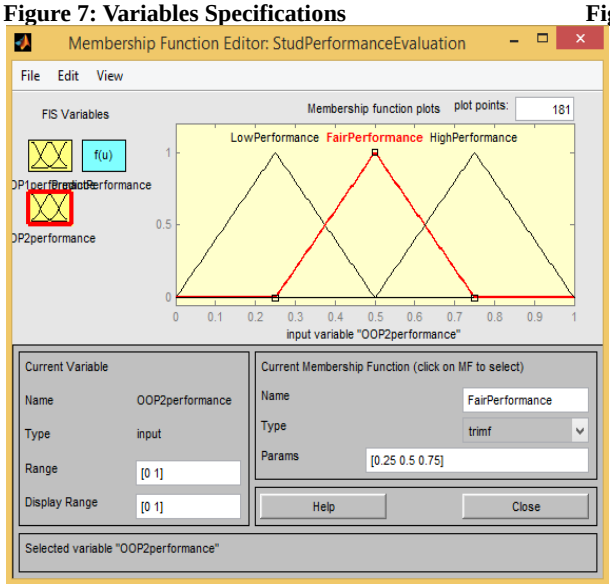

Figure 9: Input Membership Specification (OOP2Performance)

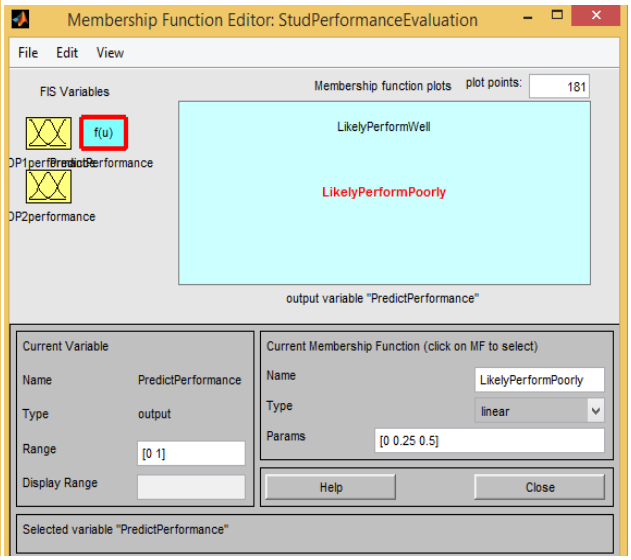

Figure 10: Output Membership Specification (PredictPerformance)

Some of the ANFIS outputs are as shown in the Figures 11-14 below

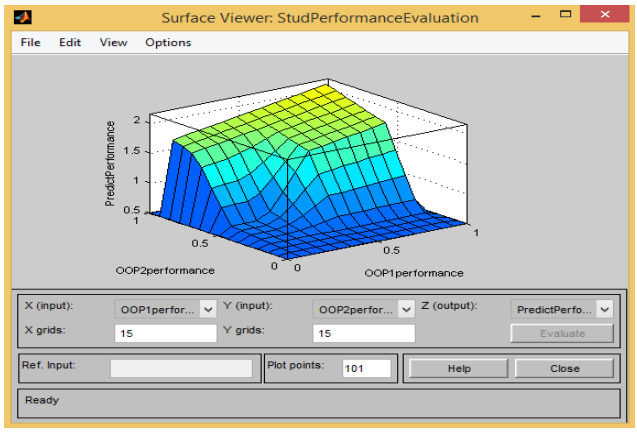

Figure 11: Result Data

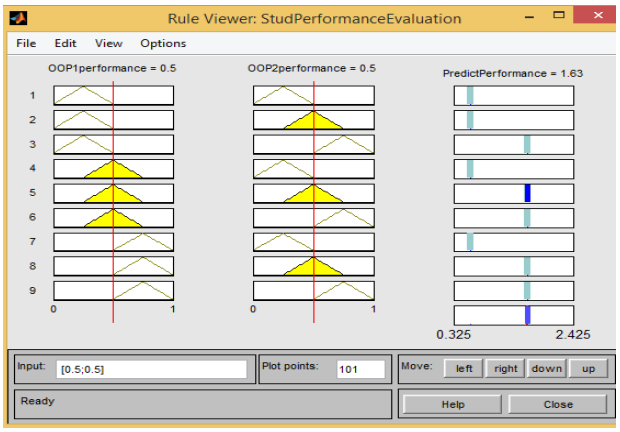

Figure 12: Output Chart

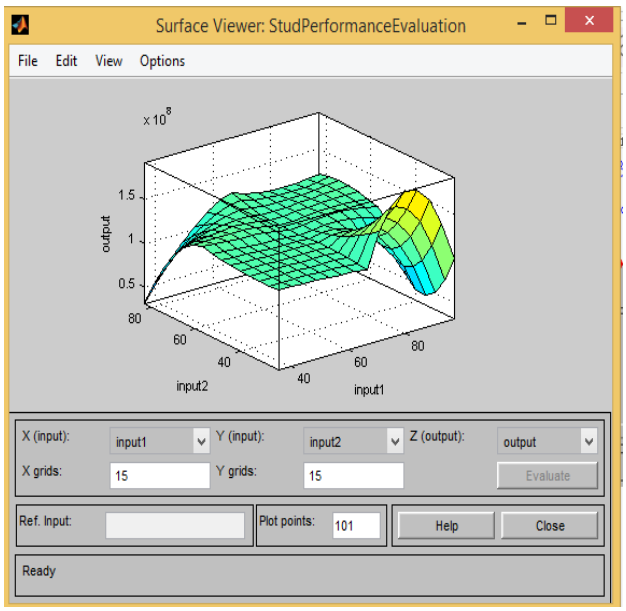

Figure 13: Result Output – Training and Testing Data

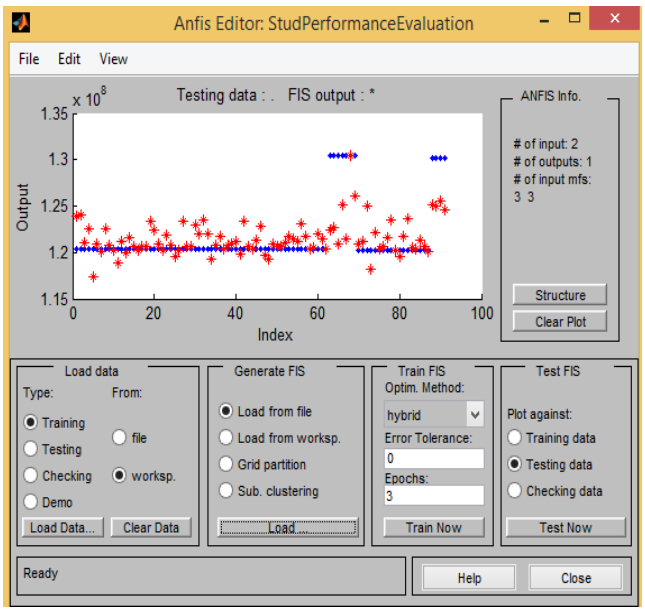

Figure 14: Output Surface Chart

7. DISCUSSION AND CONCLUSION

The result of the finding shows slight correlations in the students' performances in the two object-oriented programming courses. This implies that the level of performances in the two will likely be re-produced in subsequent object-oriented programming courses. We intend fortifying this work by working more on the data used and adopt stricter training method.

8. REFERENCES

[1] Chapel, V., Ronald C. A., and Mark, A., (2013). "*Understanding System Architecture – A Business Brief*" A White Paper for Business and Technology Professionals.

[2] Christopher, W. and Li, Frederick W. B. and Godwin, J. L. (2013), *"Predicting performance in an Introductory Programming Course by Logging and Analysing Student Programming Behaviour"*, in Proceedings of the 2013 IEEE 13th International Conference on Advanced Learning Technologies (ICALT2013). Piscataway, NJ: IEEE Computer Society, pp. 319-323.

[3] Czogala, E. and Leski, J. (2000), "*Neuro-Fuzzy Intelligent Systems, Studies in Fuzziness and Self Computing"*. Springer Verlag, Germany.

[4] Fantuzzi, C., and Rovatti, R., (1996). *"On the Approximation Capabilities of Homogeneous Takagi-Sugeno Model"*. Proceeding of the Fifth IEEE International Conference on Fuzzy System, Pages 1067 – 1072.

[5] Hamam, A., and Georganas, N. D., "*A Comparison of Mamdani and Sugeno Fuzzy Inference Systems for Evaluating the Quality of Experience of Hapto-Audio-Visual Applications"*. HAVE 2008 – IEEE International Workshop on Haptic Audio Visual Environments and their Applications, 2008.

[6] Han. J and Kamber M. (2006), *"Data Mining: Concepts and Techniques"*. 2nd ed. San Francisco, California: Morgan Kaufman.

[7] Jie A., Li Y., Chen B., Ye J. and Zou J. (2010), *"The Education Reform and Innovation of Object Oriented Programming Courses in Normal"*. The 5th International Conference on Computer Science & Education, Hefei. 978-1-4244.

[8] Jose, V., Fernando, M. D. and Alexander, M., (2004) *"Neuro-Fuzzy Systems – A Survey"*. 5th WSEAS NNA Interactions Conference.

[9] Kamyar, M., (2008). *"Takagi-Sugeno Fuzzy Modeling for Processing Control, Industrial Automation, Robotics and Artificial Intelligence"*. IEEE 8005.

[10] Mohd H. R., and Abdullah, E., (2013). *"Predicting Student Performance in Object Oriented Programming Using Decision Tree: A Case at Kolej Poly-Tech Mara, Kuantan".* 3rd International Conference on Software Engineering & Computer Systems (ICSECS - 2013), 20-22 August 2013, Universiti Malaysia Pahang. pp. 11-8.

[11] Mohsen, H. and Zahra, M., (2007). *"Application of Artificial Neural Network for Temperature Forecasting"*, World Academy of Science, Engineering and Technology, 2007.

[12] Nauk, D., Klawon, F. and Kruse R. (1997), *"Foundations of Neuro-Fuzzy Systems"* J. Wiley & Sons. Robert, F., (2001). *"Neuro Fuzzy Methods for Modeling and Fault Diagnosis"*, Eotuos Lorand University, Buderpest, September, 2001.

[13] Rojas R. (1996), *"Neural Networks"*. Springer Verlag, Berlin. Santhosh, B., and Kadar, S. I., (2011), *"An Effective Temperature Prediction System Using BPN Neural Network"*. IJES, Vol. 2, No. 1, February, 2011. ISSN 2010-0264.

[14] Sirwan, A. M., and Sattar B. S., (2013). *"A Comparison of Mamdani and Sugeno Fuzzy Inference Systems Based on Block Cipher Evaluation"*. International Journal of Science & Engineering Research. Vol. 4, Issue 12, December, 2013, ISSN 2229-5518.

[15] Somerville, I., (2011*). "Software Engineering – Ninth Edition"*. Pearson Education Incorporation (2011).

[16] Takagi, T., and Sugeno, M., (1985). *"Fuzzy Identification of Systems and Its Application to Modeling and Control."* IEEE Transactions on Systems, Man, and Cybernetics, 15, (1985):116-132.

[17] Xinghuo, Y., Onder, M. E., and Okyay, K., (2001). *"A Backpropagation Learning Framework for Feedforward Neural Network"*. IEEE Transactions on Neural Networks – No. 0-7803-6685-9101, 2001.