Final Grade Prediction Based On Quiz Average and Final Exams Using Fast Artificial Neural Networks (FANN)(November2015)

Marvin Jay O. Limjuco, and Dex Miguel F. Samson., *Students, DLSU-Manila*

***Abstract*—Final grade prediction has always been an integral part of LaSallian Engineering culture. Fast Artificial Neural Networks can be used to give students an insight on the most probable grade they can achieve given a set of data composed of their quiz average and final exam mark. The method by which the predicted grade is calculated is accomplished by training the Artificial Neural Network with data gathered from previous quiz averages and final exam marks of students together with the final grade they received for the specific subject. The goal of the project is to achieve at least an 80% accuracy in predicting the final grade of the student. This project encompasses the lessons and theories that were discussed in Machine Intelligence and provides an avenue for the students to showcase what they have learned throughout the term.**

I. Introduction

Balancing academics and proper prioritization are necessities of students in the University. Through the use of Fast Artificial Neural Networks, students are given an efficient way of knowing the probable grade they are to receive in a subject given their quiz average and how well they perform in their final exam. By having an idea on what grade they would most probably get, students can prioritize subjects that need higher final exam marks and can therefore lessen their chances of failing subjects during the term. This, in turn, can help keep their academic flowchart in-tact and effectively reduce their chances of having back-subjects which could prolong their stay in the University. The benefits of knowing which subjects to properly prioritize may also reduce the financial burden incurred by their parents – thereby allowing for money to be invested in other important matters such as health and insurance, among others.

II. Dataset

The dataset to be used in the project was gathered from Lasallian Computer Engineering students who primarily enrolled in the year 2010. Their quiz averages, final exam marks, and final grades in their major subjects: DIGITDE, CSARCHI, MPROSYS, OSYSLEC, and FCONSYS were compiled and are used as the training data for the project. Plotting these data resulted in a scattered plot that cannot be realized by simple linear regression. The use of Artificial Neural Networks was found to be the most effective means in understanding this data.

III. Artificial Neural Network

A neural network is a network made up of three layers: the input layer, the hidden layer, and the output layer. However, the hidden layer may be made up of several layers, depending on the number of neurons being used in a certain function. In contrast, an artificial neural network is used to represent non-linear hypotheses and when there are numerous sample data. The Fast Artificial Neural Network (FANN) library is used in this project. Additionally, the FANN library provides several functions that essentially handle the given neural network.

There are usually two modes in which a neural network operates: training and execution. During training, the data set is used to calculate the weights of each neuron before forward propagating it to the next layer. In FANN, the function “fann_train_rprop” is responsible for performing these operations. It is an adaptive function which means it does not have a standard learning rate. In this project, the function will be used in order to get the most probable output. The two inputs, the quiz average and finals score, is used to train and validate the most probable grade. In the project, 80% of the data will be allotted for training, 10% will be allotted for validation, and the remaining 10% will be allotted for testing. Testing works hand-in-hand with training and where the mean squared error (MSE) is computed.

IV. Model Representation and Calculations

$$h_\theta(x)=\frac{1}{\left(1+e^{-z}\right)} \qquad \text{eq. 1}$$

$$a_1^{(2)}=g\left(\theta X_0+\theta X_1+\theta X_2\right) \qquad \text{eq. 2}$$

The neurons contained in the hidden layers uses the sigmoid function given in Equation 1. In this project, a single neuron in the second layer is basically a sum of the weighted activation or sigmoid functions from the previous layer. This source code demonstrates how the weighted neurons from layer 1 is computed in layer 2:

```
//a neuron in layer 2 having a distinct weight
Layer2In[j] = Weight[0][j] ;
for( i = 1 ; i <= NumUnits1 ; i++ ) {
        //summation of the weighted neurons
from layer 1
```

```
        Layer2In[j] += Layer1Out[i] * Weight[i][j];
    }
    //each neuron in layer 2 outputs a sigmoid function
    Layer2Out[j] = 1.0/(1.0 + exp(-Layer2In[j])) ;
```

Figure 1 shows the representation of how the total weights are computed and equation 2 shows the activation function of a certain neuron in the second layer.

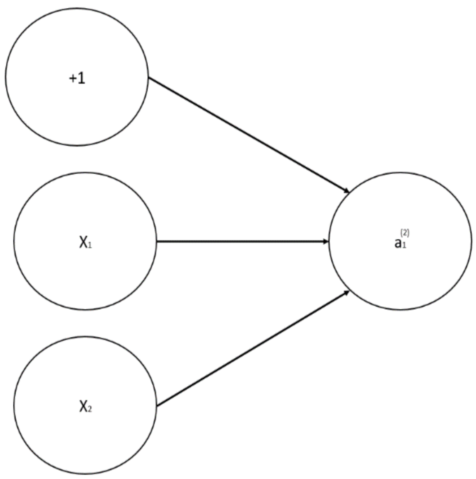

Fig. 1 – Representation of the Weighted Neurons as Inputs to the First Activation Function in the Second Layer

V. Cost Function Plot

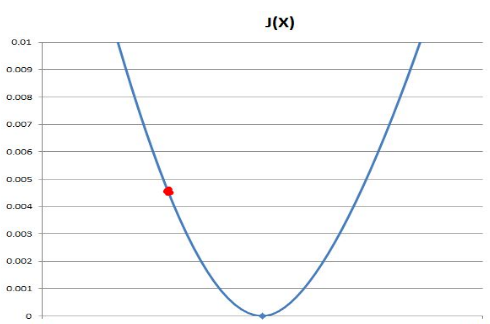

Fig 2. CSARCHI Cost Function Plot of 1.0

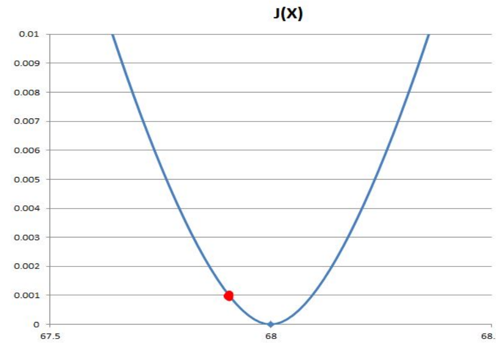

Fig 3. CSARCHI Cost Function Plot of 1.5

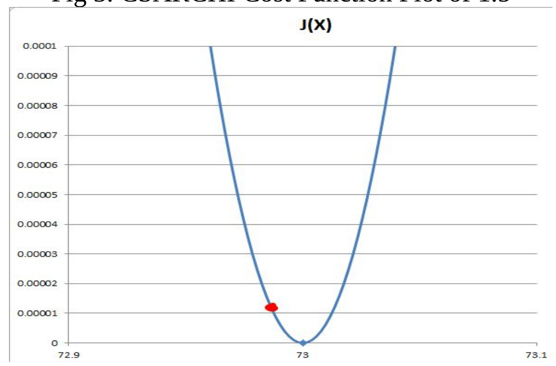

Fig 4. CSARCHI Cost Function Plot of 2.0

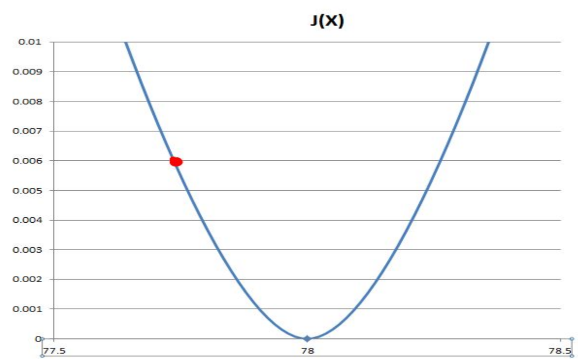

Fig 5. CSARCHI Cost Function Plot of 2.5

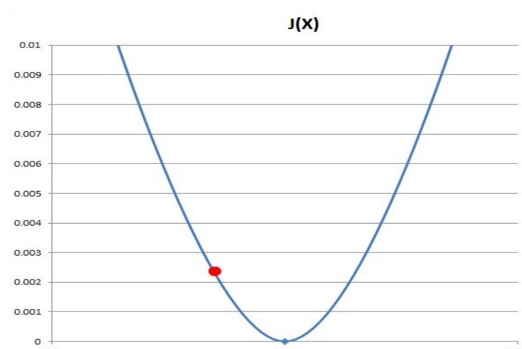

Fig 6. CSARCHI Cost Function Plot of 3.0

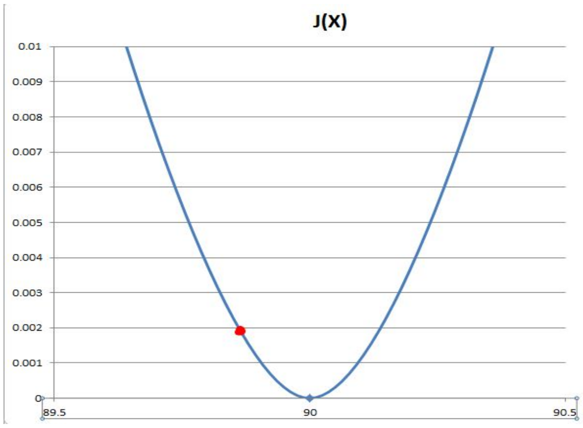

Fig 7. CSARCHI Cost Function Plot of 3.5

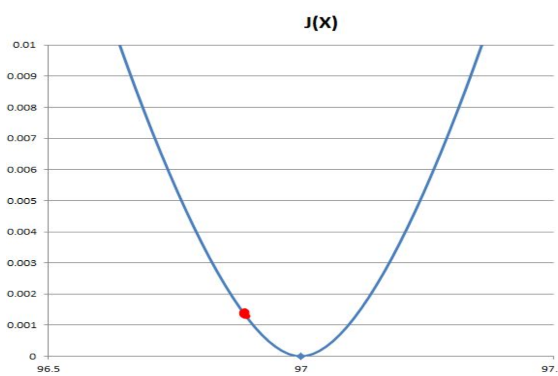

Fig 8. CSARCHI Cost Function Plot of 4.0

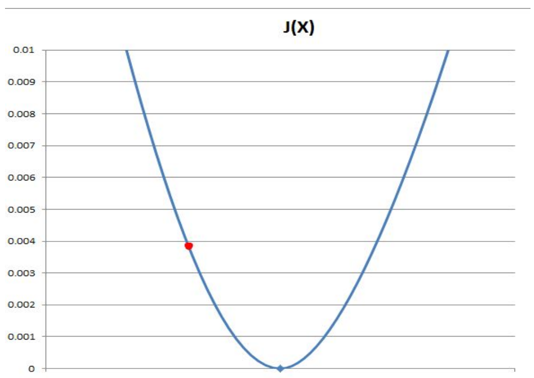

Fig 9. DIGITDE Cost Function Plot of 1.0

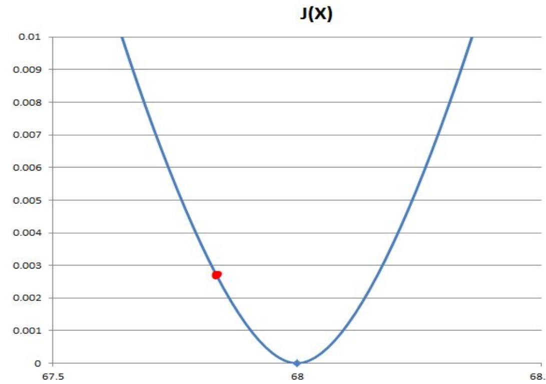

Fig 10. DIGITDE Cost Function Plot of 1.5

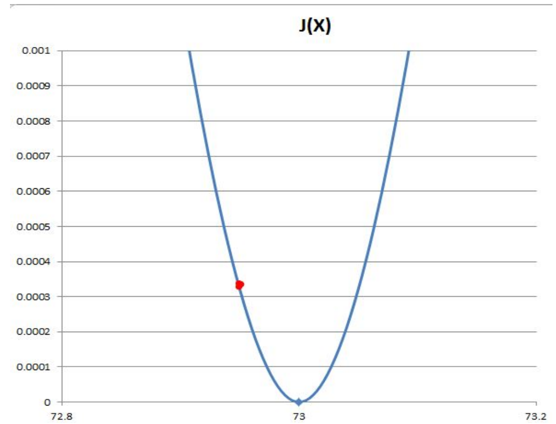

Fig 11. DIGITDE Cost Function Plot of 2.0

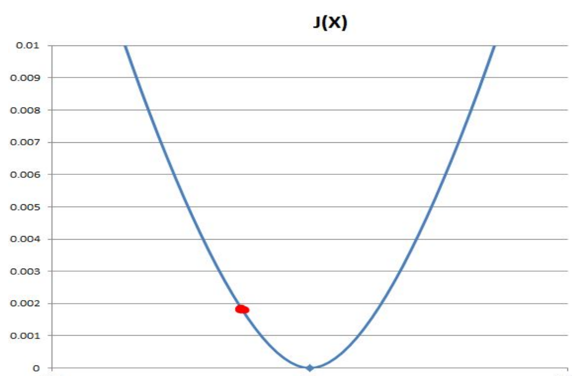

Fig 12. DIGITDE Cost Function Plot of 2.5

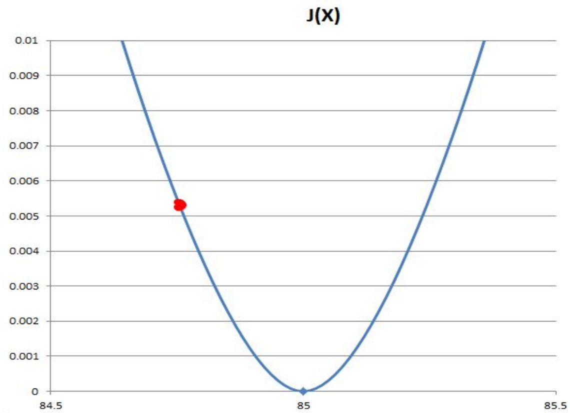

Fig 13. DIGITDE Cost Function Plot of 3.0

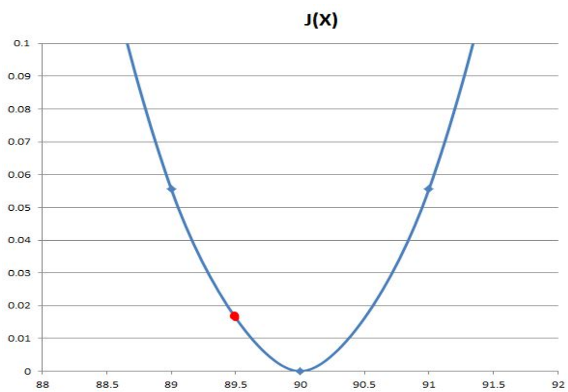

Fig 14. DIGITDE Cost Function Plot of 3.5

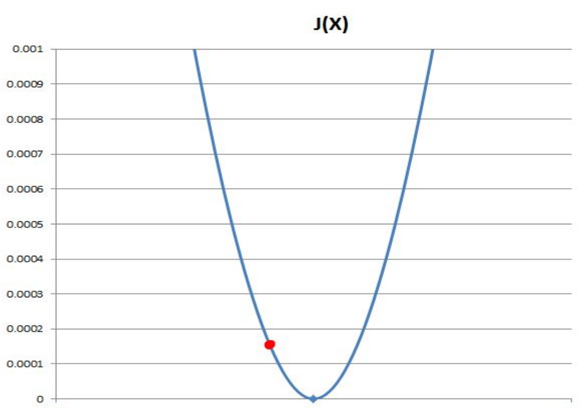

Fig 15. DIGITDE Cost Function Plot of 4.0

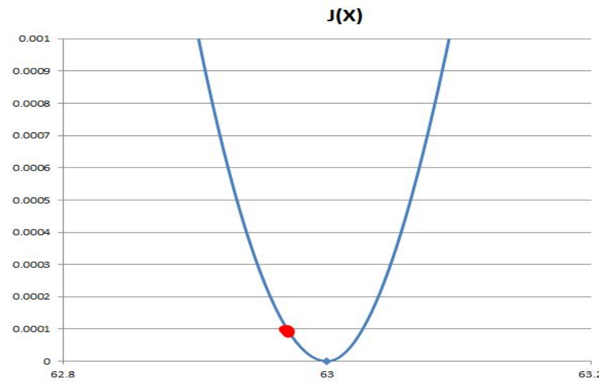

Fig 16. FCONSYS Cost Function Plot of 1.0

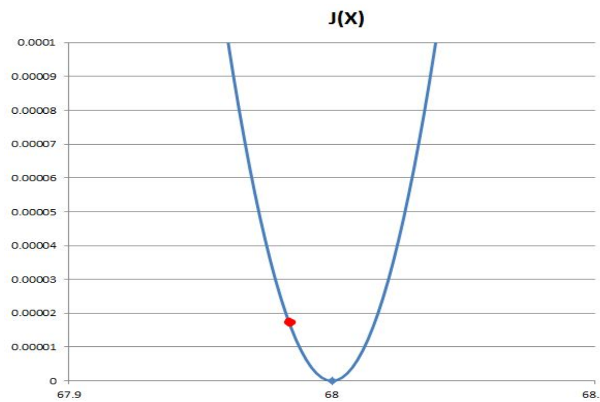

Fig 17. FCONSYS Cost Function Plot of 1.5

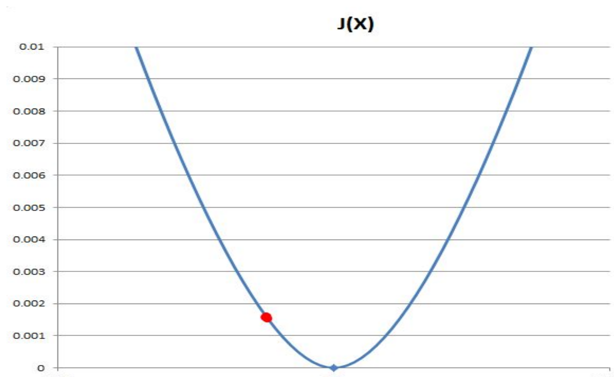

Fig 18. FCONSYS Cost Function Plot of 2.0

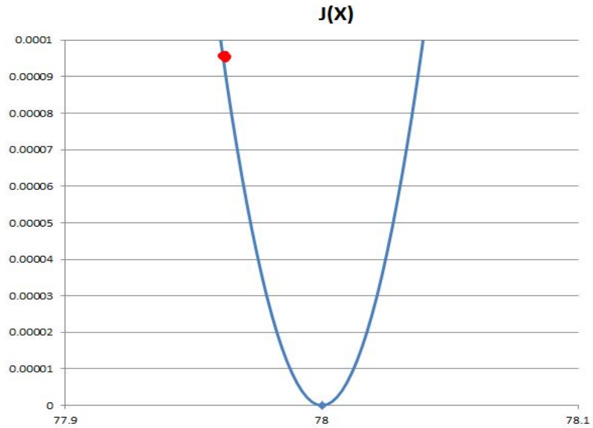

Fig 19. FCONSYS Cost Function Plot of 2.5

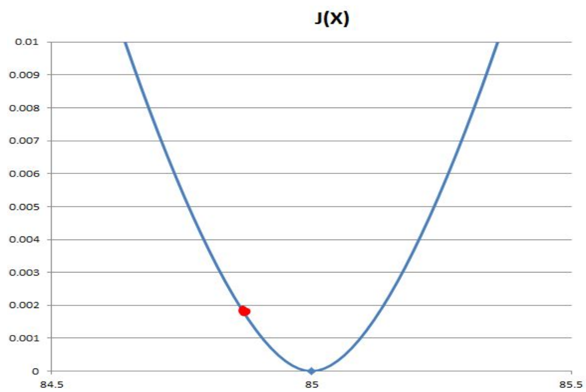

Fig 20. FCONSYS Cost Function Plot of 3.0

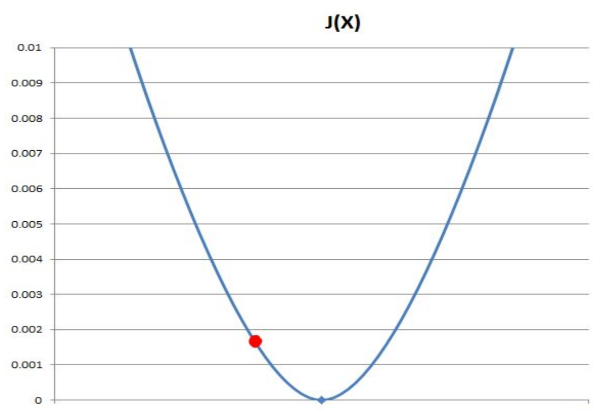

Fig 21. FCONSYS Cost Function Plot of 3.5

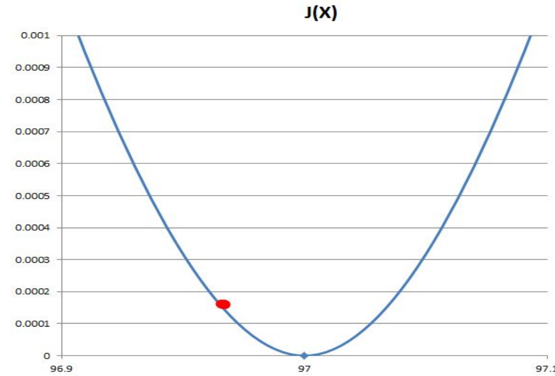

Fig 22. FCONSYS Cost Function Plot of 4.0

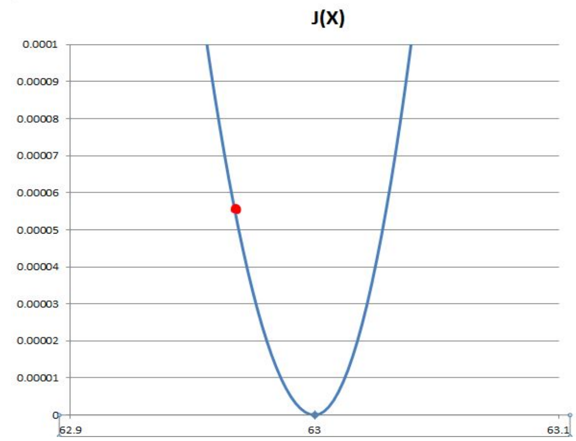

Fig 23. MPROSYS Cost Function Plot of 1.0

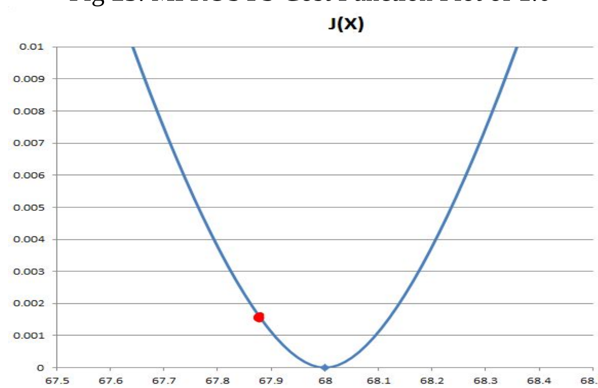

Fig 24. MPROSYS Cost Function Plot of 1.5

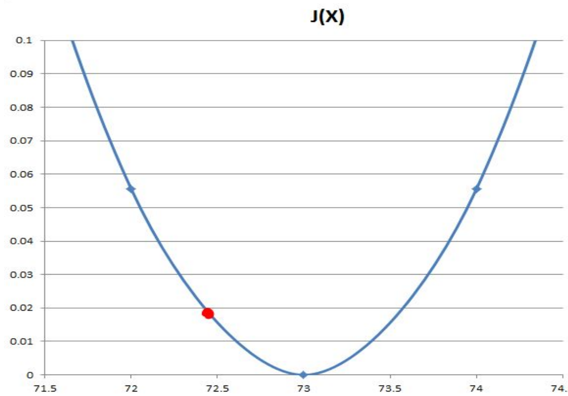

Fig 25. MPROSYS Cost Function Plot of 2.0

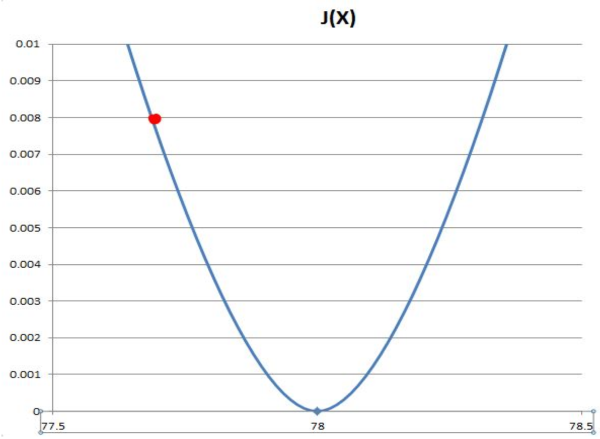

Fig 26. MPROSYS Cost Function Plot of 2.5

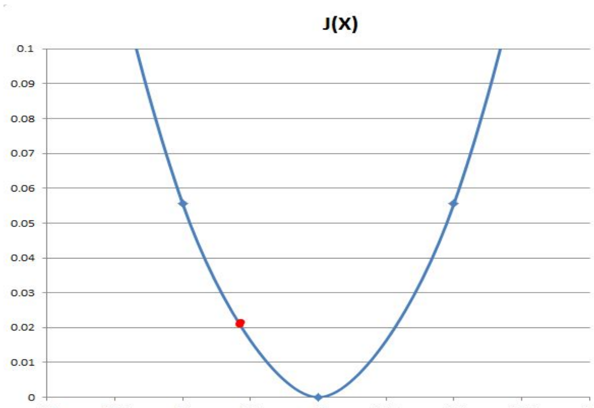

Fig 27. MPROSYS Cost Function Plot of 3.0

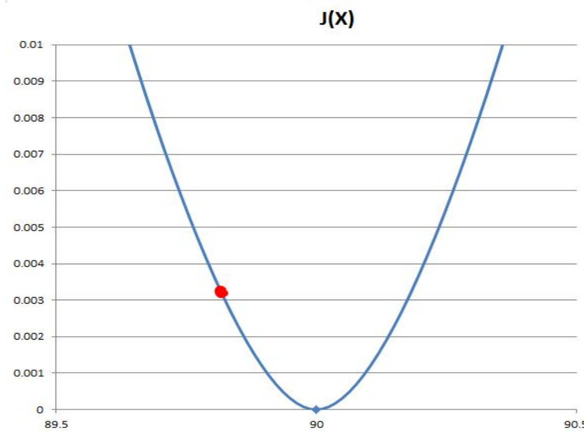

Fig 28. MPROSYS Cost Function Plot of 3.5

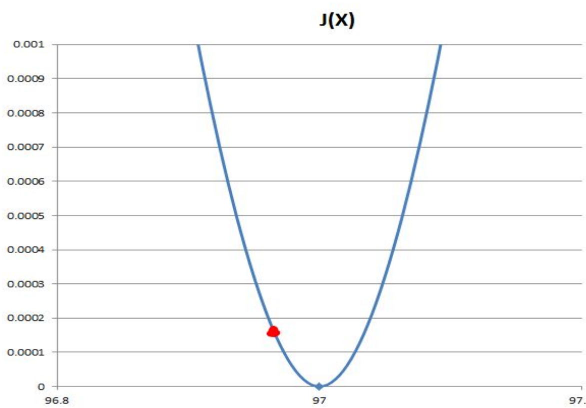

Fig 29. MPROSYS Cost Function Plot of 4.0

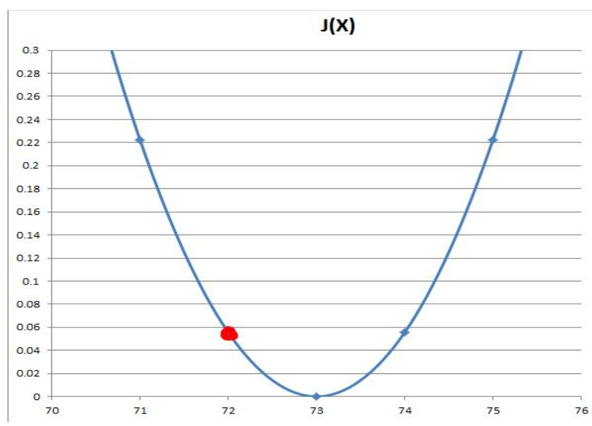

Fig 30. OSYSLEC Cost Function Plot of 2.0

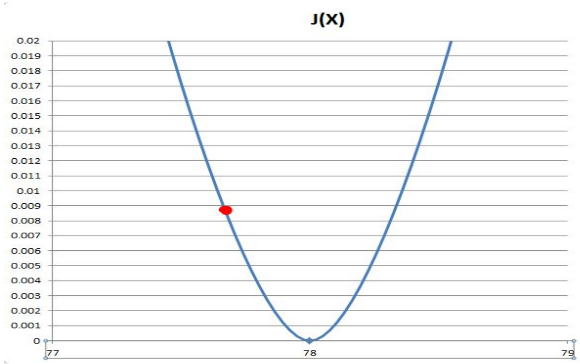

Fig 31. OSYSLEC Cost Function Plot of 2.5

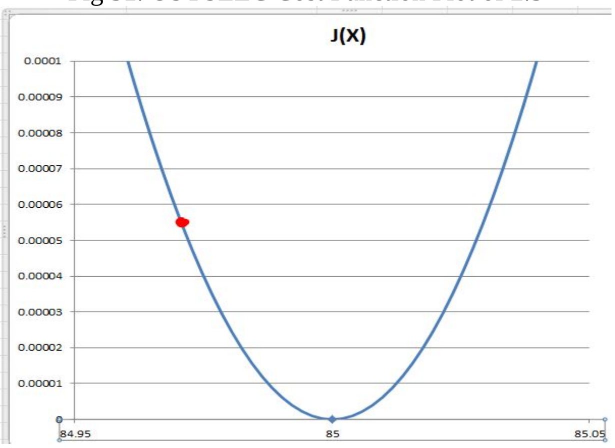

Fig 32. OSYSLEC Cost Function Plot of 3.0

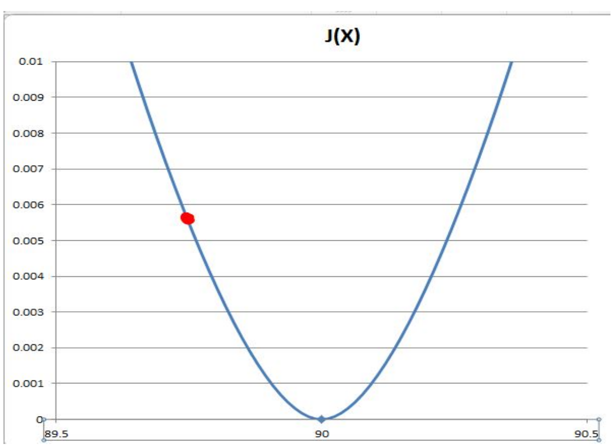

Fig 33. OSYSLEC Cost Function Plot of 3.5

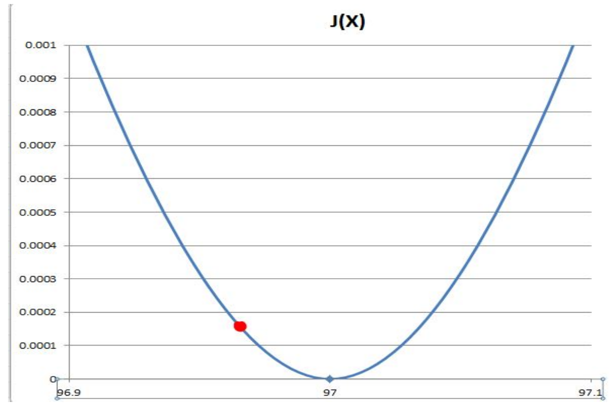

Fig 34. OSYSLEC Cost Function Plot of 4.0

$$J(\theta)=\frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x_i)-y_i\right)^2 \qquad \text{eq. 3}$$

Figures 2 to 34 shows the cost function for each grade in each subject. This function is represented in equation 3. It basically shows the difference of the generated output from the real/actual output. For example, in figure 27, it shows the cost function plot for 3.0. If the generated output produced a grade of 85, then the cost is at minimum. Additionally, a certain mark is essentially made up of several possible grades. With that being said, there is a region where in a certain cost is still accepted, as long as it is inside the grade range.

VI. Mean – Squared Error Plots

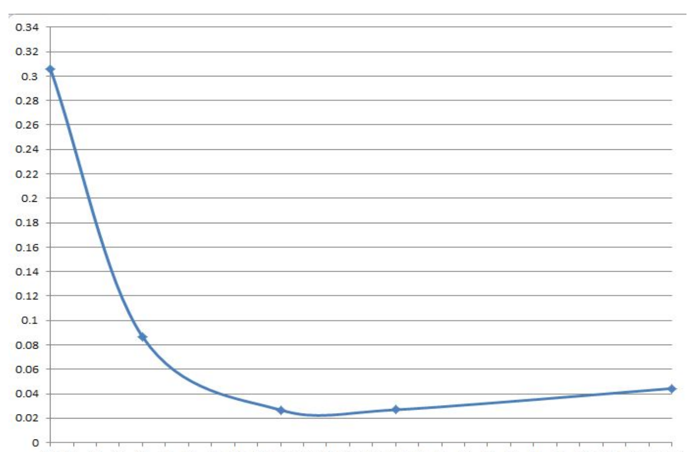

Fig 35. CSARCHI MSE Plot

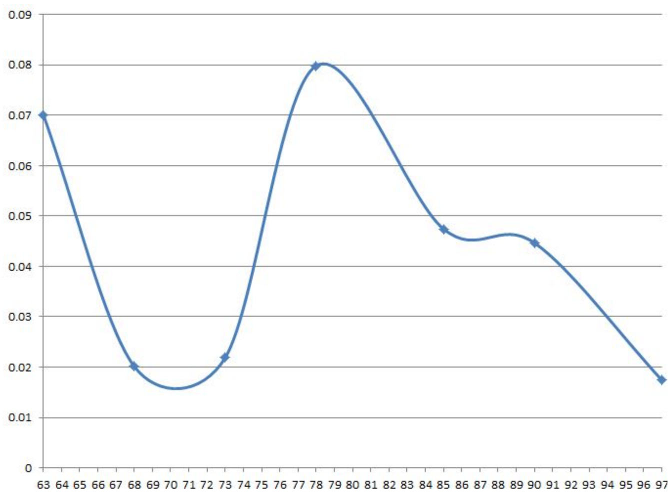

Fig 36. DIGITDE MSE Plot

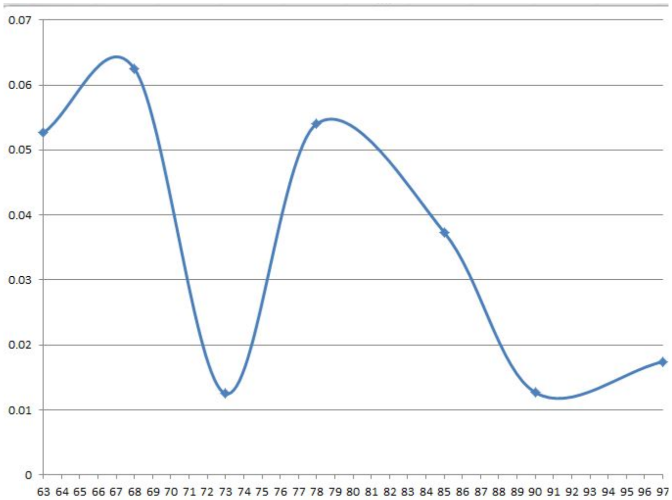

Fig 37. FCONSYS MSE Plot

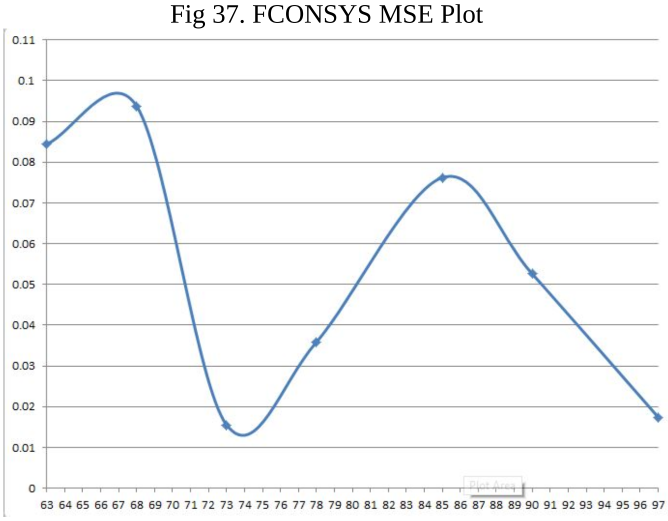

Fig 38. MPROSYS MSE Plot

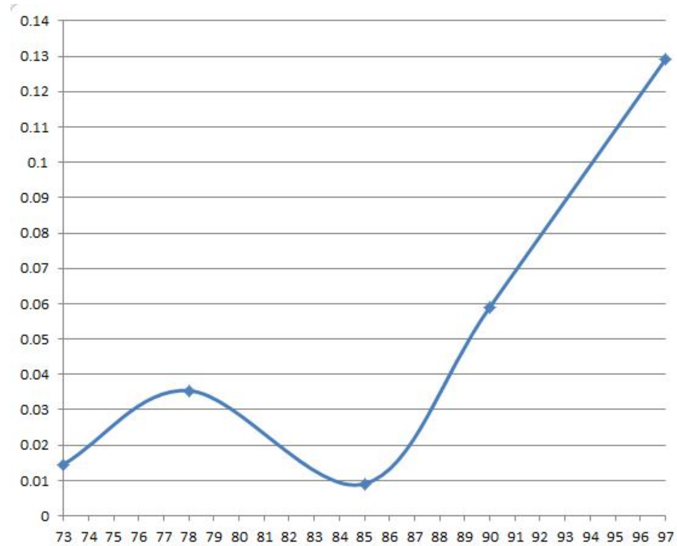

Fig 39. OSYSLEC MSE Plot

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(h_\theta\left(x_i\right) - y_i\right)^2 \qquad \text{eq. 4}$$

The mean squared error was computed after the final testing. Equation 4 shows the formula to compute the MSE. This equation basically shows the squared difference of the expected output from the actual output. The main reason why each of the MSE plots shows a unique curve is because of the data set it follows. Grades vary from each subject which affects how the FANN essentially produces it weights. For example, in figure 39, the x-axis ranged from 73 to 97, which means, students who took this class got at least a 2.0. In other subjects such as MPROSYS, as shown in figure 38, the x-axis ranged from 63 to 97. Meaning, students who took this class got final grades ranging from 1.0 to 4.0. Additionally, an MSE plot shows how the neural network is based. Looking at figure 37, the curve is at minimum when it reaches 73 and 90. This means that most of the grades in the data set got a 2.0.

VII. Confusion Matrix

	1	1.5	2	2.5	3	3.5	4	total
1	0	0	0	0	0	0	1	1
1.5	0	0	0	1	0	0	0	1
2	0	0	2	0	0	0	0	2
2.5	0	0	0	3	0	0	0	3
3	0	0	0	0	2	0	0	2
3.5	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0

Fig 40. CSARCHI Test Confusion Matrix

	1	1.5	2	2.5	3	3.5	4	total
1	0	1	0	0	0	0	0	1
1.5	0	2	0	0	0	0	0	2
2	0	0	1	0	0	0	0	1
2.5	0	2	0	0	0	0	0	2
3	0	0	0	1	0	0	0	1
3.5	0	0	0	0	0	1	0	1
4	0	0	0	0	0	0	1	1

Fig 41. DIGITDE Test Confusion Matrix

	1	1.5	2	2.5	3	3.5	4	total
1	0	1	0	0	0	0	0	1
1.5	0	0	1	0	0	0	0	1
2	0	0	2	0	0	0	0	2
2.5	0	0	0	1	1	0	0	2
3	0	0	0	0	0	1	0	1
3.5	0	0	0	0	0	1	0	1
4	0	0	0	0	0	0	1	1

Fig 42. FCONSYS Test Confusion Matrix

	1	1.5	2	2.5	3	3.5	4	total
1	0	0	1	0	0	0	0	1
1.5	0	0	0	1	0	0	0	1
2	0	0	2	0	0	0	0	2
2.5	0	0	1	1	0	0	0	2
3	0	0	0	1	0	0	0	1
3.5	0	0	0	0	0	0	1	1
4	0	0	0	0	0	0	1	1

Fig 43. MPROSYS Test Confusion Matrix

	1	1.5	2	2.5	3	3.5	4	total
1								0
1.5								0
2	0	0	1	0	0	0	0	1
2.5	0	0	2	0	0	0	0	2
3	0	0	0	0	4	0	0	4
3.5	0	0	0	0	1	0	0	1
4	0	0	0	0	1	0	0	1

Fig 44. OSYSLEC Test Confusion Matrix

	1	1.5	2	2.5	3	3.5	4	total
1	0	2	0	0	0	0	0	2
1.5	0	1	0	1	0	0	0	2
2	0	0	0	4	0	0	0	4
2.5	0	0	0	6	0	0	0	6
3	0	0	0	0	4	0	1	5
3.5	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	3	3

Fig 45. CSARCHI Train Confusion Matrix

	1	1.5	2	2.5	3	3.5	4	total
1	0	0	3	0	0	0	0	3
1.5	0	0	3	0	0	0	0	3
2	0	0	0	7	0	0	0	7
2.5	0	0	0	0	3	0	0	3
3	0	0	0	0	4	0	0	4
3.5	0	0	0	0	0	0	1	1
4	0	0	0	0	0	0	1	1

Fig 46. DIGITDE Train Confusion Matrix

	1	1.5	2	2.5	3	3.5	4	total
1	4	0	0	0	0	0	0	4
1.5	0	6	0	0	0	0	0	6
2	0	3	2	0	0	0	0	5
2.5	0	0	1	1	0	0	0	2
3	0	0	0	0	2	0	0	2
3.5	0	0	0	0	0	0	1	1
4	0	0	0	0	0	0	2	2

Fig 47. FCONSYS Train Confusion Matrix

	1	1.5	2	2.5	3	3.5	4	total
1	0	0	0	1	0	0	0	1
1.5	0	0	0	3	0	0	0	3
2	0	0	7	0	0	0	0	7
2.5	0	0	0	5	0	0	0	5
3	0	0	0	3	0	0	0	3
3.5	0	0	0	0	0	0	1	1
4	0	0	0	0	0	0	2	2

Fig 48. MPROSYS Train Confusion Matrix

	1	1.5	2	2.5	3	3.5	4	total
1	0	0	0	0	0	0	0	0
1.5	0	0	0	1	0	0	0	1
2	0	0	3	3	0	0	1	7
2.5	1	0	0	4	0	0	0	5
3	0	0	0	3	2	0	0	5
3.5	0	0	0	0	0	1	1	2
4	0	0	0	0	0	0	2	2

Fig 49. OSYSLEC Train Confusion Matrix

Figures 40 to 49 represent a confusion matrix after training and testing the data set. This matrix is used to describe the performance of the training and testing process where the values are classified as to whether it achieved the correct data. For example, in figure 47, it is shown in the first row that the training process generated four correct 1.0 grades out of the possible four. Consequently, the third row only produced two correct 2.0 grades out of the possible five.

VIII. ACCURACY

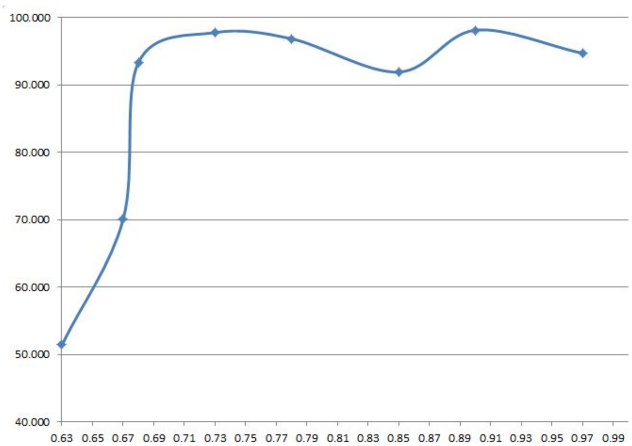

Fig 50. CSARCHI Accuracy

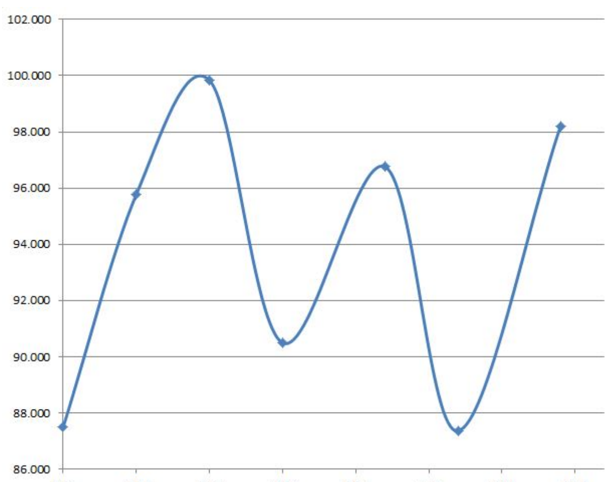

Fig 51. DIGITDE Accuracy

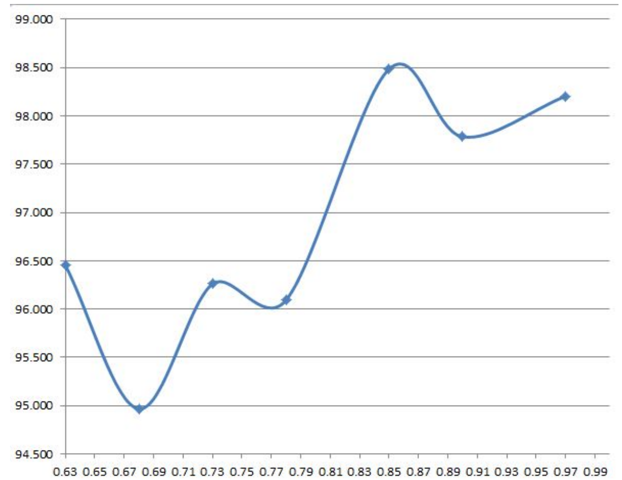

Fig 52. FCONSYS Accuracy

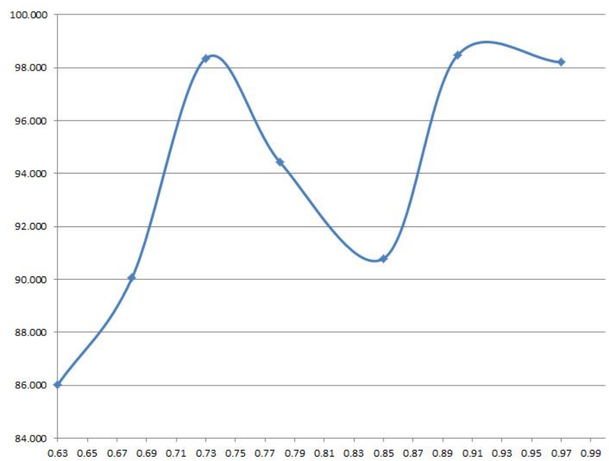

Fig 53. MPROSYS Accuracy

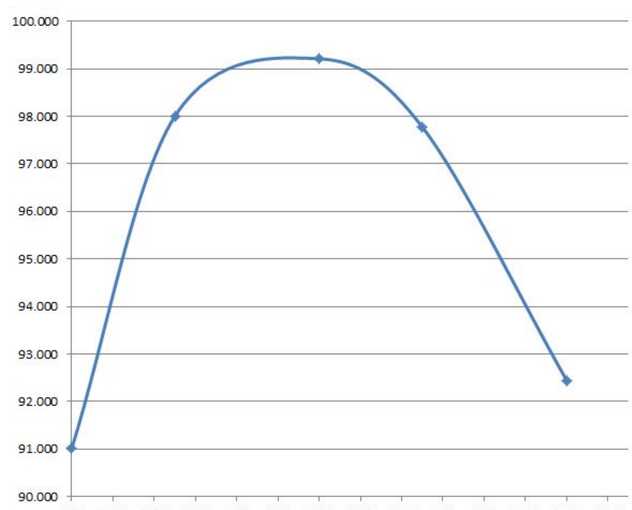

Fig 54. OSYSLEC Accuracy

$$Accuracy = 1 - \frac{|Actual - Experimental|}{Actual} \quad \text{eq. 5}$$

Figures 50 to 54 represent the accuracy of the neural network and how well it produces the correct grade. The values were based after the validation and testing process. Collectively, each subject produced an average above 80% accuracy, which is one of the objectives of this project. Each plot represents the accuracy of the generated values of the validation and testing process compared to the actual data. For example, in figure 53, the accuracy of getting a grade of 1.0 is roughly 86%. The formula is shown is equation 5.