A Functional Approximation Comparison between Neural Networks and Polynomial Regression

ONG HONG CHOON[1], LEONG CHEE HOONG[2] AND TAI SHEUE HUEY[3]
[1,2,3] School of Mathematical Sciences
Universiti Sains Malaysia,
11800 USM, Pulau Pinang,
MALAYSIA.
[1] hcong@cs.usm.my [2] lecho_leong@hotmail.com [3] tcwcll_shtai@hotmail.com

Abstract: - Multi-layered perceptron (MLP) neural networks are well known as universal approximators. They are often used as estimation tools in place of the classical statistical methods. The focus of this study is to compare the approximation ability of MLP with a traditional statistical regression model, namely the polynomial regression. Comparison among the single hidden layer MLP, double hidden layer MLP and polynomial regression is carried out on the basis of similar number of weights or parameters. The performance of these three categories is measured using fraction of variance unexplained (FVU). The closer the FVU value is to zero, the better the estimation result and this is associated with a higher degree of accuracy. From the empirical results obtained in this study, we conclude that overall polynomial regression performs slightly better than MLP for a similar number of parameter except for the complicated interaction function. Meanwhile, double hidden layer MLP outperforms single hidden layer MLP. The MLP is more appropriate than the polynomial regression in approximating the complicated interaction function.

Key words: - Artificial neural network, Multi-layered perceptrons, Polynomial regression

1 Introduction

1.1 Neural Network (NN)

A Neural Network (NN) is a system composed of many simple processing elements operating in parallel whose function is determined by the network structure, connection strengths and the processing performed at computing elements or nodes [1]. In other words, NNs are able to learn and generalize from noisy data, which is similar to statistical methods such as polynomial regression, kernel discriminant analysis, k-mean cluster analysis, projection pursuit regression and component analysis [2][3]. Furthermore, NN is employed to recognize some complex nonlinear functions. The excellent performance of NN can be attributed to the nonlinear nature of the network. Hence, we can model a smooth nonlinear mathematical function by adjusting the number of hidden layers and the number of nodes in each layer.

In our study, we use supervised learning technique to train the NN. Supervised learning technique refers to when the correct results (target values or desired outputs) are known and are given to the NN so that the NN can adjust its weights to match its outputs to the target values. After training, only input values are given when the NN is tested. Output values are then to be compared with the target values. Also in our study, we use feed forward neural networks (FANN) as our main topology where the connection between units do not form cycles and FANN must be trained to map input values into desired output values [4].

1.2 Multi-layered Perceptron (MLP)

Multi-layered perceptron (MLP) is a type of NN that is most widely used. It consists of a number of interconnected processing elements, which are also known as neurons. The neurons interact with each other through weighted connections and they are arranged in two or more layers. These weights verify the nature and strength of the influence between the interconnected neurons. Each neuron is connected to all the neurons in the next layer. The data is presented to the neural network in an input layer. Meanwhile, an output layer holds the response of the

network to the input values. The intermediate layers (hidden layers) enable this network to compute complicated associations between neurons.

1.3 Error Backpropagation Algorithm

Error backpropagation algorithm is a supervised learning technique used to train a NN [5][6]. In error backpropagation NN, each hidden and output neuron processes its inputs by multiplying each input by its weight, summing the product and then passing the sum through a nonlinear function to produce a result. NN learns by adjusting the weights of the neurons in response to the errors between the actual output values and the target output values. This is performed through the gradient descent on the sum of squares of the error for all the training sets. The training stops when average sum of squares of the error is minimized.

1.4 Polynomial Regression

A polynomial regression consists of constants and variables that are combined using operations such as addition, subtraction and multiplication. An example of polynomial regression is $\hat{Y} = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4 + \ldots + \beta_m x^m$ where $\hat{Y}$ is the predicted outcome value for the polynomial model with regression coefficients β_1 to β_m for each degree m and $\hat{Y}$ intercept β_0. It has m predictors raised to the power of i where i=1 until m. A first order polynomial (m=1) forms a linear expression. Meanwhile, a second order (m=2) polynomial forms a quadratic expression (parabolic curve), a third order (m=3) polynomial forms a cubic expression and a fourth order (m=4) forms a quartic expression [7] [8].

1.5 Data Background

In this study a total of 225 sets of data for five different nonlinear functions are generated. There are 3 factors in the sets of data generated, which are X_1, X_2 (input values) and Y (output values). The input data are independent of each other. The abscissa values were generated as uniform random variates on [0, 1] which are independent of each other. They are all translated to be non negative and scaled so that the standard deviation is 1 (for a large regular grid with 2500 points on the unit square $[0, 1]^2$).

2 Methodology

2.1 Equation of the Five Non-linear Functions for Training

Five non-linear functions taken from [9] are chosen as the training sets in this study. These functions are as follows:

1. Simple Interaction Function:
$$g^{(1)}(x_1, x_2) = 10.391[(x_1 - 0.4)(x_2 - 0.6) + 0.36)]$$

2. Radial Function:
$$g^{(2)}(x_1, x_2) = 24.234 r^2 (0.75 - r^2),$$
$$r^2 = (x_1 - .05)^2 + (x_2 - 0.5)^2$$

3. Harmonic Function:
$$g^{(3)}(x_1, x_2) = 42.659[(2 + x_1)/20 + \mathrm{Re}(z^5)],$$
where $z = x_1 + i x_2 - 0.5(1 + i)$
or equivalently to
$$g^{(3)}(x_1, x_2) = 42.659[0.1 + \tilde{x}_1(0.05 + \tilde{x}_1^4 - 10\tilde{x}_1^2\tilde{x}_2^2 + 5\tilde{x}_2^4)],$$
where $\tilde{x}_1 = x_1 - 0.5, \quad \tilde{x}_2 = x_2 - 0.5$

4. Additive Function:
$$g^{(4)}(x_1, x_2) = 1.33561[1.5(1 - x_1) + e^{2x_1 - 1} \sin(3\pi(x_1 - 0.6)^2) + e^{3(x_2 - .5)} \sin(4\pi(x_2 - .9)^2)]$$

5. Complicated Interaction Function:
$$g^{(5)}(x_1, x_2) = 1.9[1.35 + e^{x_1} \sin(13(x_1 - 0.6)^2) \cdot e^{-x_2} \sin(7x_2)]$$

2.2 Multi-layered Perceptron

A comparison is made between the data generated from multi-layered perceptron (MLP) and a nonlinear regression model, namely polynomial regression using fraction of variance unexplained (FVU). Five non-linear functions $y^{(j)}$, j = 1, 2, …, 5 in this study are used to test the performance of the single and double hidden layered MLP and the polynomial regression models as well. The error back-propagation is chosen as the supervised training algorithm and hyperbolic tangent function is the activation function that is used in the multi-layered perceptrons. Just as in [10], these five functions are, $g^{(j)} : [0,1]^2 \to R$ where j = 1, 2, …,5. 225 generated points $\{(x_{q1}, x_{q2})\}$ where q = 1,

2, …, 225, are from the range $[0, 1]^2$ and are independent of each other. This same set of values is used to train the MLP. In short, $y^{(j)} = g^{(j)}\left(x_{q1}, x_{q2}\right)$ where q = 1, 2, …, 225 and j = 1, 2, …, 5. Since the MLP model has two inputs and one output, the MLP with single hidden layer of J neurons will result in (2 × J) + (J × 1) weights; however, double hidden layer with (J,K) neurons (i.e. J neurons in the first hidden layer and K neurons in the second hidden layer) will produce (2 × J) + (J × K) + (K × 1) weights. Results are tested for single hidden layer with 6, 8, 9, 16, 18, 54 and 60 neurons whereas double hidden layer are with (3,3) , (4,4) , (6,6) and (8,8) neurons. These different sets of neurons are used to check whether the number of neurons is the determining factor for the accuracy.

2.3 Polynomial Regression

Approximation of these 5 non-linear functions can be carried out by estimating the parameters of the polynomial regression models. Statistical software MINITAB version 14 is used to perform the statistical analysis based on the four polynomial regression models in this study. The polynomial fit is done by using an approximately similar number of parameters (weights). There are four types of the polynomial regression model that are used in this study:

i. Without interaction (18 parameters except X_1X_2 term)

$$\hat{Y} = \beta_0 + \beta_1X_1 + \beta_2X_2 + \beta_3X_1^2 + \beta_4X_2^2 + \beta_5X_1^3 + \beta_6X_2^3 + \beta_7X_1^4 + \beta_8X_2^4 + \beta_9X_1^5 + \beta_{10}X_2^5 + \beta_{11}X_1^6 + \beta_{12}X_2^6 + \beta_{13}X_1^7 + \beta_{14}X_2^7 + \beta_{15}X_1^8 + \beta_{16}X_2^8 + \beta_{17}X_1X_2$$

ii. Without interaction (24 parameters except X_1X_2 term)

$$\hat{Y} = \beta_0 + \beta_1X_1 + \beta_2X_2 + \beta_3X_1^2 + \beta_4X_2^2 + \beta_5X_1^3 + \beta_6X_2^3 + \beta_7X_1^4 + \beta_8X_2^4 + \beta_9X_1^5 + \beta_{10}X_2^5 + \beta_{11}X_1^6 + \beta_{12}X_2^6 + + \beta_{13}X_1^7 + \beta_{14}X_2^7 + \beta_{15}X_1^8 + \beta_{16}X_2^8 + \beta_{17}X_1^9 + \beta_{18}X_2^9 + \beta_{19}X_1^{10} + \beta_{20}X_2^{10} + \beta_{21}X_1^{11} + \beta_{22}X_2^{11} + \beta_{23}X_1X_2$$

iii. Without interaction (25 parameters)

$$\hat{Y} = \beta_0 + \beta_1X_1 + \beta_2X_2 + \beta_3X_1^2 + \beta_4X_2^2 + \beta_5X_1^3 + \beta_6X_2^3 + \beta_7X_1^4 + \beta_8X_2^4 + \beta_9X_1^5 + \beta_{10}X_2^5 + \beta_{11}X_1^6 + \beta_{12}X_2^6 + + \beta_{13}X_1^7 + \beta_{14}X_2^7 + \beta_{15}X_1^8 + \beta_{16}X_2^8 + \beta_{17}X_1^9 + \beta_{18}X_2^9 + \beta_{19}X_1^{10} + \beta_{20}X_2^{10} + \beta_{21}X_1^{11} + \beta_{22}X_2^{11} + \beta_{23}X_1^{12} + \beta_{24}X_2^{12}$$

iv. With interaction (28 parameters)

$$\hat{Y} = \beta_0 + \beta_1X_1 + \beta_2X_2 + \beta_3X_1X_2 + \beta_4X_1^2 + \beta_5X_2^2 + \beta_6X_1^3 + \beta_7X_2^3 + \beta_8X_1^2X_2 + \beta_9X_1X_2^2 + \beta_{10}X_1^4 + \beta_{11}X_2^4 + \beta_{12}X_1X_2^3 + \beta_{13}X_1^2X_2^2 + \beta_{14}X_1^3X_2 + \beta_{15}X_1^5 + \beta_{16}X_2^5 + \beta_{17}X_1X_2^4 + \beta_{18}X_1^2X_2^3 + \beta_{19}X_1^3X_2^2 + \beta_{20}X_1^4X_2 + \beta_{21}X_1^6 + \beta_{22}X_2^6 + \beta_{23}X_1X_2^5 + \beta_{24}X_1^2X_2^4 + \beta_{25}X_1^3X_2^3 + \beta_{26}X_1^4X_2^2 + \beta_{27}X_1^5X_2$$

The main null hypothesis of a regression is that there is no relationship between independent variables and that particular dependent variable.

The hypothesis is written as:

H_0: $\beta_i = \beta_j = 0$

H_1: at least one $\beta_i \neq \beta_j \neq 0$
for $i \neq j$ for i, j = 0,1,2,3,4…27

p-values are used to test whether the regression coefficients and the constant are equal to zero. If corresponding p-value is less than 0.05 at 5% significant level, we reject the null hypothesis and accept the alternative hypothesis that indicates that at least one of the regression coefficients is not equal to zero. Then there is a statistically significant relationship between these variables.

In regression model, the coefficient of determination, R Squared (R^2) is a measure of the variation of the dependent variable that can be well explained by the regression line and the independent variables. It is used to determine if the equation is of adequate accuracy. The closer the R Squared is to 100%, the better the equation fits the data and the better the regression model is. R Squared of α% carries the meaning of about α% of the total variation of the dependent variable is caused by the deviation in the independent variables. Adjusted R Squared is a modification of R Squared that adjusts for the number of dependent variables in a model. The adjusted R Squared increases only if the new term (explanatory variable) improves the model more than would be expected by chance.

2.4 Criteria for Comparison

According to [9], the fraction of variance unexplained (FVU) on the training set is used for the comparison of the accuracy in the simulations of the five non-linear functions. Fraction of Variance Unexplained (FVU) is the proportion of the variation among the observed values $y_1, y_2 \ldots y_n$ that remains

unexplained by the fitted regression. When the FVU is close to 0, the variation of the observed values of *y* around the fitted regression function is much smaller than their variation around $\overline{y}$. It is defined as FVU = $\frac{\sum_{i=1}^{n}\left(y_i-\hat{y}_i\right)^2}{\sum_{i=1}^{n}\left(y_i-\overline{y}\right)^2}$

where y_i, $\hat{y}_i$ and $\overline{y}$ represent true *y*, predicted *y* and the mean of *y* respectively.

The FVU values of these three categories are compared by using the similar number of parameter or weights as the basis of comparison. The effectiveness of the approximation nature of a neural network (multi-layered perceptrons) versus a classical statistical method (polynomial regressions) can be determined. The smaller the FVU value, the better the approximation ability. Table 1 shows the comparison criteria in our study. For example, single hidden layer MLP with 6 neurons, double hidden layer MLP with 3,3 neurons and polynomial regression with 18 parameters are compared because all of them have 18 weights.

Table 1: Comparison of Three Methods by using Comparable Number of Parameters or Weights

Single Hidden Layer Perceptron (J neurons)	Corresponding Weights for Single Hidden Layer (2 x J) + (J x 1)	Double Hidden Layer Perceptron (J,K neurons)	Corresponding Weights for Double Hidden Layer (2 x J) + (J x K) + (K x 1)	Polynomial Regression (Parameters)
6	18	3,3	18	18
8	24	-	-	24, 25
9	27	4,4	28	28
18	54	6,6	54	-
16	48	8,8	88	-
54	162	-	-	-
60	180	-	-	-

Table 2: Summary of Results for Polynomial Regression

		Without interaction (18 parameters) except the X_1X_2 term	Without interaction (24 parameters) except the X_1X_2 term	Without interaction (25 parameters)	With interaction (28 parameters)
*g*1	P-value of coefficient of term <0.05	Constant, X_1, X_2, X_1X_2	Constant, X_1, X_2, X_1X_2	Constant	Constant, X_1, X_2, X_1X_2, X_1^2, X_2^2, X_1^3, $X_1^2X_2$, $X_1X_2^2$, $X_1X_2^3$, $X_1^3X_2$, $X_1X_2^4$, $X_1^4X_2$, $X_1X_2^5$, $X_1^5X_2$
	R-Sq (R^2)	100.0%	100.0%	17.4%	100.0%
	R-Sq(adj)	100.0%	100.0%	7.4%	100.0%

*g*2	P-value of coefficient of term <0.05	Constant	Constant	Constant	Constant, X_1, X_2, X_1X_2, X_1^2, X_2^2, X_1^3, X_2^3, $X_1^2X_2$, $X_1X_2^2$, X_1^4, X_2^4, $X_1^2X_2^2$, X_1^5, X_2^5, X_1^6, X_2^6
	R-Sq (R^2)	88.1%	88.1%	88.1%	100.0%
	R-Sq(adj)	87.1%	86.7%	86.7%	100.0%
*g*3	P-value of coefficient of term <0.05	Constant	Constant	Constant	Constant, X_1, X_1X_2, X_1^2, X_2^2, X_2^3, $X_1^2X_2$, X_1^4, X_2^4, $X_1X_2^3$, $X_1^2X_2^2$, $X_1^3X_2$, X_1^5, $X_1X_2^4$, $X_1^3X_2^2$
	R-Sq (R^2)	25.8%	25.8%	25.8%	100.0%
	R-Sq(adj)	19.7%	17.3%	16.9%	100.0%
*g*4	P-value of coefficient of term <0.05	Constant, X_1, X_2, X_1^2, X_2^2, X_1^3, X_2^3, X_1^4, X_2^4, X_2^5, X_2^6, X_2^7, X_1^8, X_2^8	Constant, X_1, X_2, X_1^2, X_2^2, X_2^3, X_2^4, X_1^5, X_2^5, X_1^6, X_2^6, X_1^7, X_2^7, X_1^8, X_2^8, X_1^9, X_2^9, X_1^{10}, X_2^{10}, X_1^{11}, X_2^{11}	Constant, X_1, X_2, X_1^2, X_2^2, X_1^3, X_2^3, X_2^4, X_2^5, X_2^6, X_2^7, X_2^8, X_2^9, X_2^{10}, X_2^{11}, X_2^{12}	Constant, X_1, X_2, X_2^2, X_2^3, X_2^4, X_2^5 and X_2^6
	R-Sq (R^2)	100.0%	100.0%	100.0%	97.9%
	R-Sq(adj)	99.9%	100.0%	100.0%	97.6%
*g*5	P-value of coefficient of term <0.05	Constant, X_1X_2	Constant, X_1X_2	Constant	Constant, X_2, X_1X_2, X_1^3, $X_1^2X_2$, $X_1X_2^2$, X_1^4, $X_1^2X_2^2$, $X_1^3X_2$, $X_1X_2^4$, $X_1^2X_2^3$, $X_1^3X_2^2$, $X_1^4X_2$, $X_1X_2^5$, $X_1^3X_2^3$, $X_1^4X_2^2$, $X_1^5X_2$
	R-Sq (R^2)	48.2%	48.2%	40.0%	91.0%
	R-Sq(adj)	43.9%	42.3%	32.8%	89.8%

where *gi* , i = 1,2,3,4,5 represents the five non-linear training functions in [9].

Table 3: FVU Values for Five Non-linear Functions with MLP and Polynomial Fits

Single hidden layer MLP							
	MLP (6 hidden neurons)	MLP (8 hidden neurons)	MLP (9 hidden neurons)	MLP (16 hidden neurons)	MLP (18 hidden neurons)	MLP (54 hidden neurons)	MLP (60 hidden neurons)
g1	0.25574	0.25620	0.03528	0.03113	0.03123	0.03137	0.02954
g2	1.01019	1.01010	1.01005	1.00991	1.00988	1.00965	0.99550
g3	0.87975	0.87713	0.81884	0.40545	0.55329	1.00874	0.11256
g4	0.51714	0.47955	0.30814	0.29690	0.29249	0.27076	0.29381
g5	0.55677	0.32834	0.19475	0.05738	0.06437	0.13110	0.04997

Double hidden layer MLP					
		MLP (3,3 hidden neurons)	MLP (4,4 hidden neurons)	MLP (6,6 hidden neurons)	MLP (8,8 hidden neurons)
	g1	0.03774	0.02857	0.02833	0.02812
	g2	0.02773	0.01343	0.00294	0.00254
	g3	0.41188	0.48098	0.17491	0.18651
	g4	0.38449	0.23379	0.03878	0.01713
	g5	0.25665	0.12465	0.04318	0.02250

Polynomial regression fit					
		Without interaction (18 parameters except X_1X_2 term)	Without interaction (24 parameters except X_1X_2 term)	Without interaction (25 parameters)	Interaction (28 parameters)
	g1	1.65E-10	1.09E-09	0.90230	2.92E-10
	g2	0.14411	0.16250	0.20569	1.64E-09
	g3	0.74230	0.83265	0.89502	1.68E-07
	g4	0.00095	0.07538	0.13753	0.02139
	g5	0.84031	0.90431	0.66272	0.09193

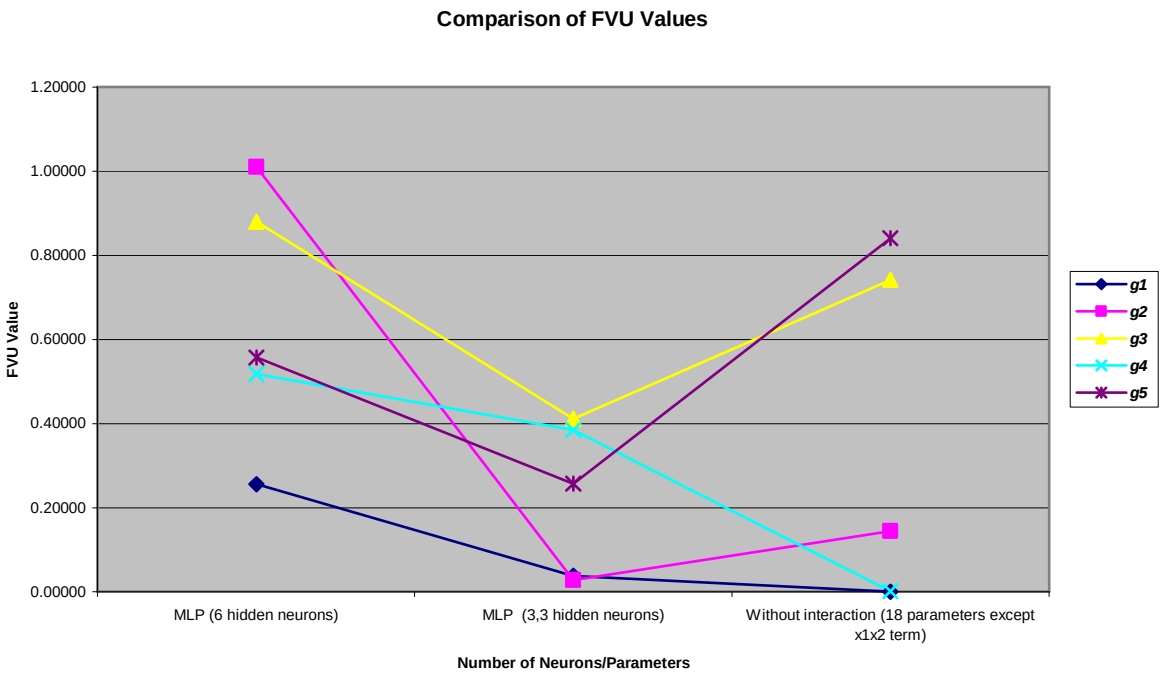

Figure 1: FVU Values for Five Non-Linear Functions with 18 Weights/Parameters

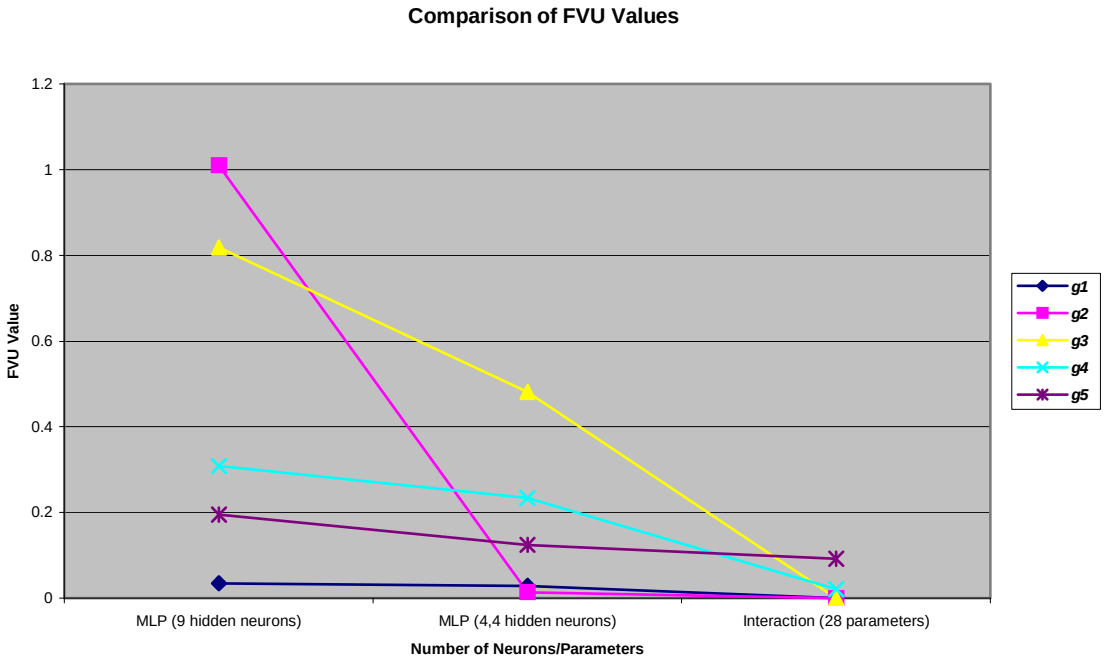

Figure 2: FVU Values for Five Non-Linear Functions with around 28 Weights/Parameters

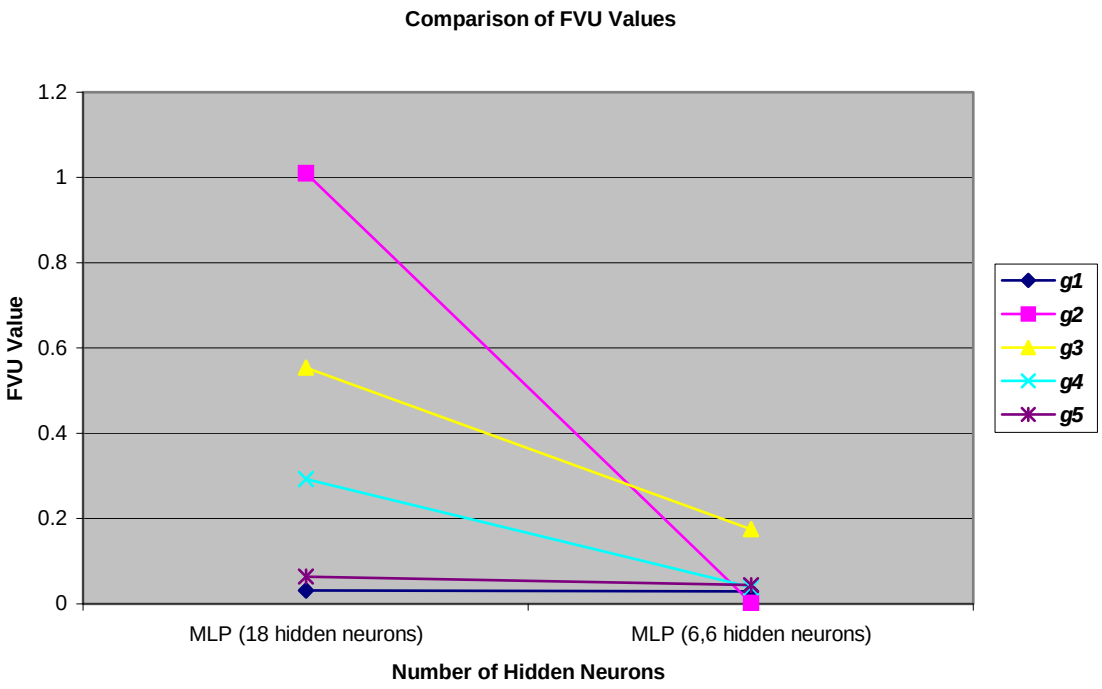

Figure 3: FVU Values for Five Non-Linear Functions with 54 Weights

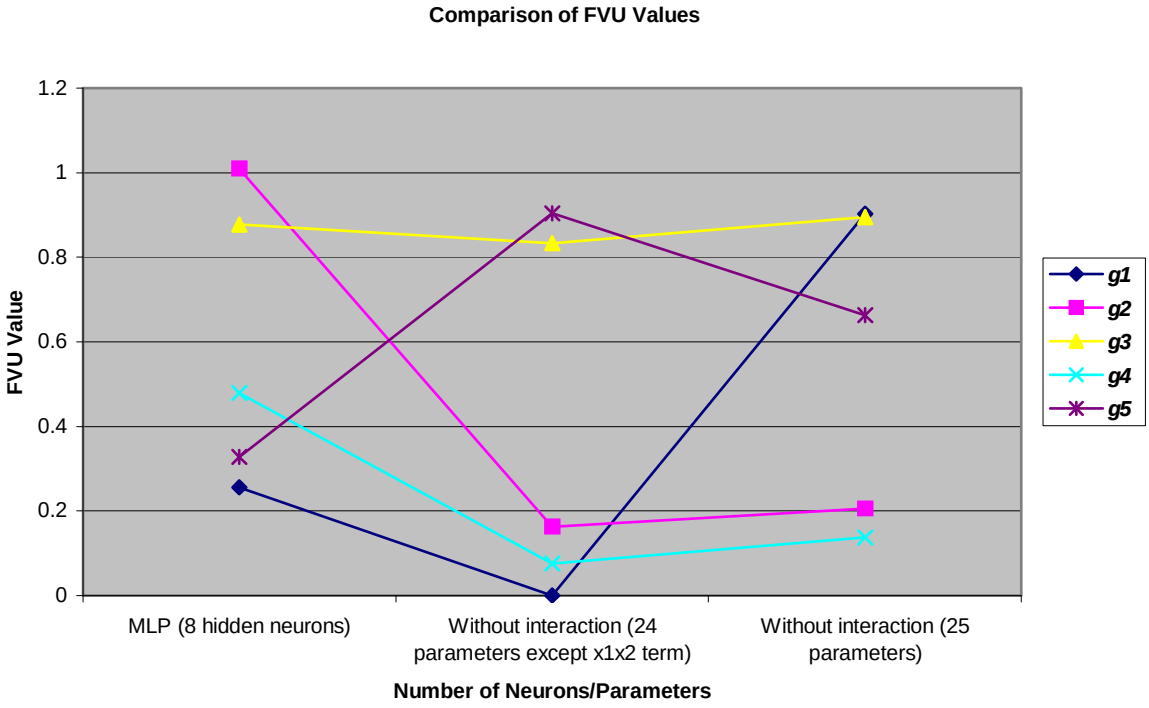

Figure 4: FVU Values for Five Non-Linear Functions with around 24 Weights/Parameters

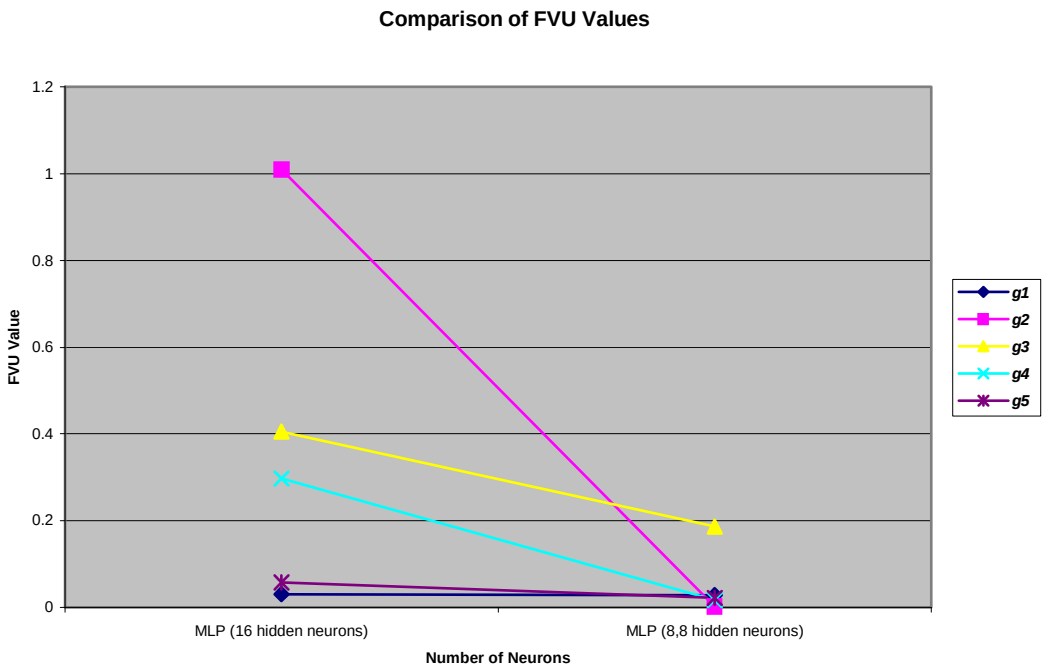

Figure 5: FVU Values for Five Non-Linear Functions with Total 16 hidden neurons

3 Results

From Table 2, the polynomial regression model with interaction terms (28 parameters) reveals that when interactions exist in the model, more significant terms are found at 95% significant level. This model provides the good fit for these five non-linear functions with higher values of R Squared and adjusted R Squared. Its variability is well explained by the independent variables. The existence of interaction in regression model contributes to a better estimate ability for non-linear functions.

Referring to Table 3, the FVU values generally decrease as the number of hidden neurons increase in the single hidden layer MLP as well as the double hidden layer MLP. This fact is true for all the 5 different non-linear functions. However, for the case of the polynomial regression, the FVU values of all 5 functions are relatively small and close to 0 respectively when model of 28 parameters with interaction is employed. As these FVU values approach 0, the predicted output will be very close to the value of target output. From the optimal FVU values, polynomial regression outperforms the MLP with single hidden layer MLP and double hidden layer MLP for all functions except for the complicated interaction function (g5). Meanwhile, double hidden layer MLP is more suitable to be employed in the complicated interaction function g5. Double hidden layer MLP outperforms single hidden layer MLP. The MLP is more appropriate than the polynomial regression in approximating the complicated interaction function.

From Figure 1, we observe that simple interaction function (g1) and additive function (g4) have the smallest FVU value when the approximation is done by using polynomial regression model without interaction (18 parameters except X_1X_2 term). However, MLP with double hidden layer (3, 3) hidden neurons gives the smallest FVU value to radial function (g2), harmonic function (g3) and complicated interaction function (g5).

Figure 2 shows that polynomial regression with interaction (28 parameters) contributes the lowest FVU values to all five non-linear functions in our study compared to the other two methods, which are 9 hidden neurons in single hidden layer MLP and (4, 4) hidden neurons in double hidden layer MLP. The existence of interaction in regression model provides better estimation ability for non-linear functions as compared to MLP when all three have the same number of around 28 weights.

As we can notice, the negative slope appears in the Figure 3. This indicates that double hidden layer MLP with (6, 6) neurons give the smaller FVU values throughout all five non-linear functions as compared to the single hidden layer MLP with 18 neurons. We realize that the single hidden layer MLP does not outperform the double hidden layer MLP

because the larger the FVU value, the greater the variation of the model.

We observe that polynomial regression without interaction (24 parameters except X_1X_2 term) gives the lowest FVU values for the five non-linear functions respectively except for complicated interaction function (g5) from the comparison in Figure 4. This polynomial regression fit is slightly better than polynomial regression model without interaction (25 parameters) because of the existence of the only interaction term (X_1X_2) in regression model with 24 parameters. The interaction feature possesses the better estimating ability than those models without interaction. For complicated interaction function (g5), its FVU value is the lowest when the approximation is done by using 8 hidden neurons MLP.

The comparison of FVU values under the similar number of hidden neurons is shown in Figure 5. As we can see from the figure above, 16 hidden neurons and (8, 8) hidden neurons are used in the comparison for single hidden layer MLP and double hidden layer MLP respectively. Both of these MLP models have 16 hidden neurons, but the total number of their weights is different. (8, 8) hidden neurons contribute more weights (1x8 + 8x8 + 2x8 = 88) than 16 hidden neurons (1x16 + 2x16 = 48). A larger number of weights usually give the better approximation result with smaller FVU value as we observe from Figure 5.

4 Conclusion

In general, when we do not consider the number of parameters or weights being used, approximation using polynomial regression generates lower FVU value as compared to approximation using neural networks with single hidden layer MLP and double hidden layer MLP. This is true for all training functions except for the complicated interaction function (g5) and this is consistent with the result from [11]. According to [11], neural networks are capable of exploring relationships among the data which are difficult to arrive at using traditional statistical methods. The optimal number of parameters in polynomial regression is 18 parameters (without interaction except for X_1X_2 term) and 28 parameters (with interaction term). The existence of interaction term in the regression model will produce higher R Squared and adjusted R Squared value. Polynomial fit without interaction does not give many significant terms and this situation contributes to less desirable approximation ability.

On the other hand, double hidden layer MLP provides better estimation results than single hidden layer MLP without considering the number of parameters being used. Apart from that, the number of hidden neurons also plays a crucial role in determining the effectiveness of an approximator. We found that the approximator with more weights or parameters will perform slightly better than those with less weights or parameters. The performance of the MLP can be improved by increasing the number of neurons to handle very complicated functions (provided no over fitting occurs) but increasing the weight of polynomial regression does not give any large influence to the model (see [12]). Furthermore, the increasing the number of parameters or weights in the polynomial regression model will make the model more complicated when doing the approximation

When the comparison of FVU values is carried out on the basis of similar number of parameters or weights, polynomial regression generally performs better than the MLP except for the complicated interaction function. In neural networks, double hidden layer MLP has lower FVU values when compared to the single hidden layer MLP. This is consistent with the result in which double hidden layer MLP is superior in extracting information [13].

If the comparison of FVU values is based on similar number of hidden neurons, double hidden layer MLP performs better than single hidden layer MLP. In fact, having similar number of neurons does not necessarily imply similar number of weights or parameters.

The need for function approximation arises in many branches of applied mathematics. In general, a function approximation problem requires us to select a function among a well defined class that closely matches or approximates a target function. In this study, we have compared the approximation ability of multi-layered perceptron (MLP) neural network with the polynomial regression using five nonlinear functions from [9].

A comparison can also be done using other types of neural networks like, for example, the radial basis function neural network and using a different appropriate set of nonlinear functions as training sets. Similarly, other multivariate statistical methods like the projection pursuit regression can be used for comparison.

References:

[1] Darpa. Neural Network Study. *AFCEA International Press*. 1998 p. 60.

[2] Sadovski , A.L., Tissot, P., Michaud, P. And Steidley, C. Statistical and Neural Network Modeling and Prediction of Tides in the Shallow Waters of the Gulf of Mexico. *Proceeding of the WSEAS International Conference on System Science, Applied Mathematics and Computer Science*. 2002 p. 2131-2136.

[3] Marteau, P.F. and Monbet V. Conditional Prediction of Markov Processes using Non Parametric Viterbi Algorithm – Comparison with MLP and GRNN Models. *Proceeding of the WSEAS Multi Conference:Neural Networks and Applications.* 2004.

[4] Sarle, W.S., ed. Neural Network FAQ, part 1 of 7: Introduction, *Periodic* Posting to the Usenet Newsgroup comp.ai.neural-nets, URL: ftp://ftp.sas.com/pub/neural/FAQ.html 1997. (Accessed on 02 January 2008)

[5] Raul, Rojas. Neural Networks – A Systematic Introduction. *Springer Verlag, Berlin, New-York.* 1996 chap. 7.

[6] Murfi, H. and Kusumoputra, B. A Computational Study of Incremental Projection Learning in Neural Networks . *Proceeding of the 3rd WSEAS International Conference on Neural Networks and Applications.* 2002. p. 3181-3187

[7] Armitage, P. & Berry G. Statistical Methods in Medical Research. 3rd edition. *Blackwell*. 1994.

[8] Kleinbaum, D.G., Kupper, L.L., Muller, K.E. and Nizam, AApplied Regression Analysis and Other Multivariable Methods. 3rd edition. *Duxbury Press, Pacific Grove California*. .1998.

[9] Hwang, J.N., Lay, S.R., Maechler, M., Martin, R.D. and Schimert, J. Regression Modeling in Back-Propagation and Projection Pursuit Learning. *IEEE Transactions on Neural Networks*. **5** (3). 1994.

[10] Ong, H.C., Lim, C.K. and Yong, Y.W. Non Linear Approximations using Multi-layered Perceptions and Polynomial Regressions. *Proceedings of the 2nd IMT-GT 2006 Regional Conference on Mathematics, Statistics and Application.* 2006. p.115-119.

[11] Dvir, D., Arie, B.D., Sadeh, A. and Shenhar. A.J. Critical Managerial Factors Affecting Defense Projects Success: A Comparison between Neural Network and Regression Analysis. *Engineering Applications of Artificial Intelligence*. **19** (5): 535-543. 2006.

[12] Michaelsen J. Environmental Data Analysis, University of California, Santa Barbara,USA.(http://www.geog.ucsb.edu/~joel/g210_w05/lecture_notes/lect18/oh05_18_1.html). 2005. (Accessed 9 December 2005)

[13] Min, H., Lei, C. and Hua, M. Application of Four-layer Neural Network on Information Extraction. *Neural Networks*. **16** (5-6): 547-553. 2003.