OPTIMUM NUMBERS OF SINGLE NETWORK FOR COMBINATION IN MULTIPLE NEURAL NETWORKS MODELING APPROACH FOR MODELING NONLINEAR SYSTEM

RABIATUL ADAWIYAH M.N. AND ZAINAL A.

Faculty of Chemical Engineering, Universiti Sains Malaysia Engineering Campus, Seri Ampangan 14300 Nibong Tebal, Seberang Perai Selatan, Pulau Pinang, Malaysia.

chzahmad@eng.usm.my

ABSTRACT: This paper is focused on finding the optimum number of single networks in multiple neural networks combination to improve neural network model robustness for nonlinear process modeling and control. In order to improve the generalization capability of single neural network based models, combining multiple neural networks is proposed in this paper. By studying the optimum number of network that can be combined in multiple network combination, the researcher can estimate the complexity of the proposed model then obtained the exact number of networks for combination. Simple averaging combination approach is implemented in this paper which is applied to nonlinear process models. It is shown that the optimum number of networks for combination can be obtained hence enhancing the performance of the proposed model.

ABSTRAK: Karya ini memberi tumpuan kepada pencarian angka optimum untuk rangkaian tunggal dalam gabungan rangkaian neural pelbagai untuk meningkatkan kemantapan model kawalan dan model proses tidak linear. Untuk meningkatkan keupayaan generalisasi model rangkaian tunggal, penggabungan rangkaian neural pelbagai telah dicadangkan dalam kertas kerja ini. Kajian bilangan optimum rangkaian tungal yang boleh digabungkan dalam gabungan rangkaian pelbagai, penyelidik boleh menganggarkan kerumitan model yang dicadangkan. Bilangan rangkaian yang tepat untuk gabungan juga boleh di kira. Pendekatan gabungan purata yang mudah berdasarkan model proses tidak linear telah di kaji dalam kertas ini. Bilangan optimum rangkaian untuk kombinasi yang

KEYWORDS: *neural network; multiple neural networks combination; nonlinear process; conic water tank*

1. INTRODUCTION

Artificial neural networks have been shown to be able to approximate any continuous non-linear functions and have been used to build data base empirical models for non-linear processes[1]. Hence what is a neural network? According to Haykin [2]:"A neural network is a massive parallel-distributed processor that has a natural capability for storing experiential knowledge and making it available for use. It resembles the brain in two respects knowledge is acquired by the networks through a learning process. Interneuron connection strengths known as synaptic weights are used to store the knowledge."

Furthermore, the main advantage of neural network based process models is that they are easy to build. This feature is particularly useful when modeling complicated processes where detailed mechanistic models are difficult to develop. However, a critical shortcoming of neural networks is that they often lack robustness unless a proper network training and validation procedure is used. Robustness of the model can be defined as one of the baseline to judge the performance of the neural network models and it is really related to the learning or training classes as what Bishop [3] described:"The importance of neural networks in this context is that they offer very powerful and very general framework for representing non-linear mappings from several input variables to several output variables, where the form of the mapping is governed by a number of adjustable parameters."

There are a lot of factors that contributed to the successful research on neural networks and among them the two main factors are as follows. The first one is that neural networks are very powerful modeling tool capable of modeling extremely complex functions [1, 4, 5]. In particular, neural networks are non-linear models, which are very useful in modeling nonlinear systems that cannot be successfully modeled by linear models.

The second main factor is that neural networks are easy to use and develop and they basically learn by examples. The neural network users gather representative data, and then invoke a training algorithm to automatically learn the structure of the data [6, 7, 8]. Because of the advantages or the tremendous capability of neural networks, currently there are a lot of applications of neural networks in industry and business where neural networks are applied in signal processing, control, pattern recognition, medicine, speech processing and in business.

In order to improve the robustness of neural networks a number of techniques have been developed lately like regularization [9] and the early stopping method[10]. Ohbayashi[11] implemented the universal learning rule and second order derivatives to increase the robustness in neural network models. Robustness is enhanced by minimizing the change in the values of criterion function caused by the small changes around nominal values of system parameters. Lack of the robustness in individual neural networks is basically due to the over fitting of the models [12]. Therefore combination of neural networks has come up and researchers concentrate on how over fitting can be avoided by improving the learning algorithm or by combining the neural networks.

Overfitting basically refers to the poor generalization of the networks due to fitting the noise in the data [13]. Furthermore, the trained network might not minimize the error on the training data set because it has uncontrolled excess dynamics capability or because the training data itself is corrupted with noise. The representation capability of a neural network is determined by its size (number of neurons). If networks are too large they can find many solutions which fit the training set data exactly, but which contain high frequency dynamics is not present in the underlying function. When the data is corrupted with noise a second form of over fitting occurs. Here the data itself contain high frequencies not present in the underlying function, with the result that minimizing the error on the data set will result in the networks fitting the noise.

Neural networks are related to the basic principle of brain [14] and try to mimic how brain works. They have been developed since 1940 after World War 2 when industrialization was growing rapidly. Neural networks are generally structured in layers of which all the neurons are connected between the adjacent layers. As mentioned by

Willis et al. [15], more accurate representation of the processes are required to ensure good process control performance especially in Advance Process Control. Therefore neural network models must be robust or stable when they are applied to new (unseen) data. Even though neural network models are very powerful non-linear modeling tools, noises in the input data sometimes cause the model over-fitting. Over-fitting and under fitting is the main problem in developing neural network models.

In over-fitting, the error on the training data set is driven to a very small value, but when applied to unseen data, the network error is large and the generalization capability of the neural network is poor. While under fitting is due to that the neural network itself cannot cope with or fails to capture the relationship within the complex data [16].

Therefore a lot of techniques have been introduced to improve the generalization capability of neural network models like regularization techniques [12, 13, 17], Bayesian learning [8, 18] and also by using the parsimonious networks structure [19]. The most exceptional model for this approach is network pruning techniques and sequential orthogonal training techniques. A sequential orthogonal training technique gradually builds up a neural network model and avoids unnecessarily large networks structure [20]. Among those approaches for improving neural network generalization, the combination of multiple neural networks seems to be very effective. Therefore the multiple neural networks and combination of multiple neural networks is proposed in this paper with the aim of enhancing the single neural network robustness.

The paper is organized as follows. Section 2 presents the concept multiple neural networks and how its can be combined. Section 3 presents the nonlinear process modeling for a nonlinear process. Applications of the proposed technique to two case studies are given in Section 4. Finally, the last section concludes this paper.

2. MULTIPLE NEURAL NETWORK

The idea of multiple neural networks came up from Wolpert [21] where he described about stacked generalization which is a technique for combining different representations to improve the overall prediction performance. It can also be described as an architecture of network consisting of several sub-models and a mechanism which combines the outputs of these sub-models [22]. There are several types of multiple neural networks but the underlying ideas are basically similar and the main difference is on how to create the sub-models as shown in Fig. 1. Two major types of multiple neural networks are described here.

The first category is multiple model neural networks [23, 24]. The training data are totally different in building the individual networks which can be built using different inputs in different regions of operation. The idea of this approach is to adapt different information by using different inputs, and by combining this information a better prediction can be obtained [22, 25]. The learning algorithm in each network can also be different and can be supervised or unsupervised methods. Other multi model approach are introduced by Jacobs [26] by using the ‘mixture of local expert’. Then, Jordan and Jacobs[27] came up with the hierarchical mixture of neural networks. In this case they basically discuss about the supervised learning algorithm and how the divide and conquer method works.

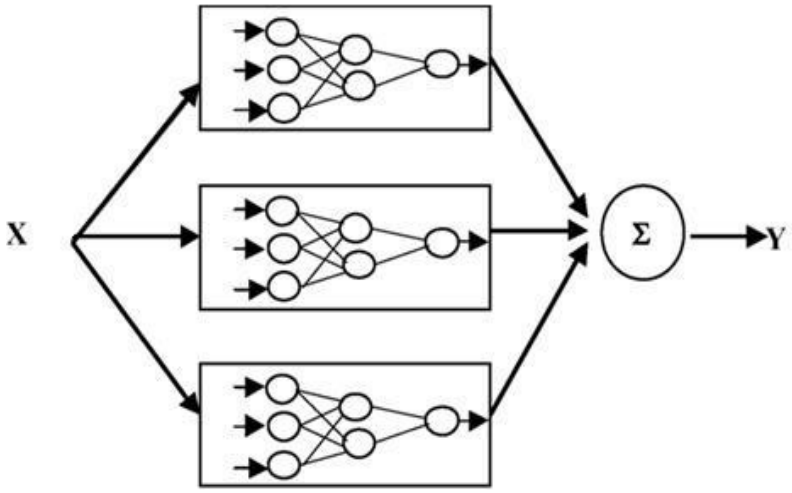


Fig. 1: A multiple neural networks.

Some examples of multi model applications are in the field of pattern recognition where different models represent different image classification [28, 29]. Medical application of multi models is presented by Jerebko [30] where different classifications of polyps as single neural network models using different inputs are combined and better prediction rate is obtained. It has also been used in other medical fields like in diagnosis application and in detecting the lung cancer [31, 32]. Multiple models have also been applied in time series forecasting [24]. In this case, each model forecasts a difference time series prediction or prediction horizon and this reduces the recursive prediction promoted to reducing the recursive error occurred in the long range prediction. It also shows that the multi network model performs better than single networks.

The second category is to creating multiple models using the same training data but re-sampled or partitioned using particular algorithms [33, 34]. There are three main algorithms being used to re-sample or partition the training data which are bagging or bootstrap [34, 35] which is being used in this paper to create a multiple neural networks, adaboost [36, 37] and randomization [38]. The motivation of creating those different inputs or partitions is to create the effective network ensembles [39]. The bootstrap or bagging basically refers to replication of a training data set where the bootstrap algorithm re-samples the original training data set. Some of the data samples may occur several times, and other may not occur in the sample at all. The individual training sets are independent and the neural networks can be trained in parallel. In this paper the data were resample using bootstrapping method.

The development of computer capability also promoted the development of multiple neural networks. Application of multiple neural networks will grow rapidly and become an important component of future research. This is also due to the various used of the neural networks and combining neural networks is one of the methods that increase the performance of network models. Therefore, the objective of this paper is to study the optimum numbers of network that can be combined. The single network will be added one in a time to be combined until the optimum number of network is obtained based on the sum square error (SSE) which is used the simple averaging method as shown below:

2.1 Simples Averaging

This method is the most common method in combining several model outputs with the weights fixed as shown below:

$$\hat{Y} = w_1\hat{y}_1 + w_2\hat{y}_2 + ... + w_n\hat{y}_n \quad (1)$$

where $\hat{y}_i$ is the network prediction from the ith network, n is the number of networks to be combined, $\hat{Y}$ is the final prediction output, and $w_i = 1/n$ is the weight for combining the ith network. The main disadvantage of this approach is that all the networks have the same contribution to the final prediction output even though some of the networks might have better predictions then others; consequently it might deteriorate the model.

3. NONLINEAR PROCESS MODELING

In this case study, the individual networks were trained by the Levenberg-Marquardt optimization algorithm with regularization and "early stopping". The aggregated of single network are varies from 2 to 50 numbers of networks. If the number of networks is too small we might not get the optimum reduction of the SSE in the combination. To accommodate systems with lag elements, while re-sampling the training and testing data using bootstrap re-sampling techniques, the training and testing were already in discrete time function, therefore, by re-sampling discrete time function, it will not affect the sequence of input-output mapping of the prediction.

All weights and biases were randomly initialized in the range from –0.1 to 0.1. The individual networks are single hidden layer feed forward neural networks. Hidden neurons use the sigmoid activation function whereas output layer neurons use the linear activation function. To cope with different magnitudes in the input and output data, all the data were scaled to zero mean and unit standard deviation. The data for neural network model building were divided into: 1) Training data (for network training); 2) Testing data (for cross-validation based network structure selection and early stopping); and 3) Unseen validation data (for evaluation of the final selected model). In networks with fixed structure, network structures (numbers of hidden neurons) were determined through cross-validation. Networks with different numbers of hidden neurons were trained on the training data and tested on the testing data. The network with the lowest SSE on the testing data is selected. In assessing the developed models, SSE on the unseen validation data is used as the performance criterion.

3.1 Case Study: Water Tank Level Prediction

Figure 2 shows the model of conic water tank level apparatus. There is an inlet stream to the tank and an outlet stream from the tank. Manipulating the inlet water flow rates will regulates the water tank level.

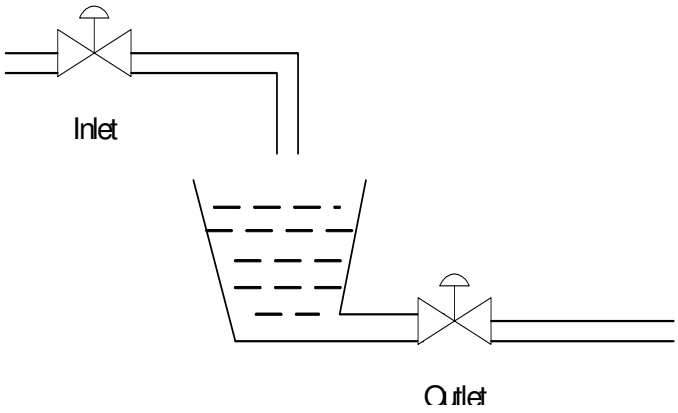


Fig. 2: Conic water tank.

$$\frac{dV}{dt} = Q_1 - Q_o \tag{2}$$

where, V is represents as volume of water in the tank (cm^3), Q1 is inlet water flows rates (cm^3/s) and Q_o is outlet water flows rates (cm^3/s). The outlet water flow rate, Q_o, is related to the tank level, *h*, by the following equation:

$$Q_o = k\sqrt{h} \tag{3}$$

where *k* is constant for a fixed valve opening.The volume of water in the tank is related to the tank level by the following equation:

$$V = \pi h\left[r^2 + \frac{hr}{\tan\theta} + \frac{h^2}{3(\tan\theta)^2}\right] \tag{4}$$

where *r* is the tank bottom radius and θ is the angle between the tank boundary and the horizontal plane.

Combining equations above, the following dynamics model for the tank level is obtained:

$$\frac{dh}{dt} = \frac{Q1 - k\sqrt{h}}{n\left[r^2 + \frac{2rh}{\tan\theta} + \frac{h^2}{(\tan\theta)^2}\right]} \tag{5}$$

Based on the above model, a simulation file is developed to simulate the process. The parameters used in the simulation are *r* = 10 cm, *k* = 34.77 cm 2.5/s and θ = 60. The sampling time is 10 second. The above equation indicates that the relationship between the inlet water flow rate and the water level in the tank is quite non-linear. The outlet valve characteristic showed that the static gain increased with tank level. The time constant of the processes increases with the tank level because the tank is of a conical shape. Thus, both the static and dynamic characteristics of the process vary with the operating condition. All the building data are generated from the simulation program and noise with the distribution N (0, 0.7 cm) are added to simulated tank. The data was divided into three sections, which are training data, testing data and validation data respectively.

4. RESULTS AND DISCUSSION

In this case study, the input data to the neural network system will be divided into three sections (testing, training and validation). All these data have to be scaled before it can be use for analysis which is to normalize the data to zero mean and unity standard deviation. This is important since different input data have different magnitude or units. The dynamics model for tank level prediction is in the form of:

$$\hat{h}(t) = f\left[h(t-1), Q_1(t-1)\right] \tag{6}$$

where $\hat{h}$ represent the predicted tank level. The optimum number of hidden neuron has to be determined before the networks being used for training, testing and validation purpose. Based on the analysis, the number of hidden neurons that came up with the least SSE for the tank level model is used for other analysis in multiple neural networks. Table 1 shows the results of SSE for training and testing data for each trial while Fig. 3 shows the plot of the results.

From Table 1 and Fig. 3, the optimum number of nodes with minimum SSE is 3 nodes with the value of SSE 3.2668. Hence, 3 nodes will be used for the tank level prediction analysis.

Table 1: Sum Square Error (SSE) for training and testing data for different no. of nodes for tank level prediction

No. of Nodes in hidden layer	SSE Training Data	SSE Testing Data	SSE (Testing + Training)
1	2.1703	1.6045	3.7748
2	1.7278	1.9299	3.6577
3	**1.6273**	**1.6395**	**3.2668**
4	1.6305	1.9049	3.5354
5	1.6074	1.8927	3.5001
6	1.6195	1.9123	3.5318
7	1.6376	1.9746	3.6122
8	1.6762	2.1139	3.7901
9	1.6329	2.2068	3.8397
10	1.7344	2.2205	3.9549

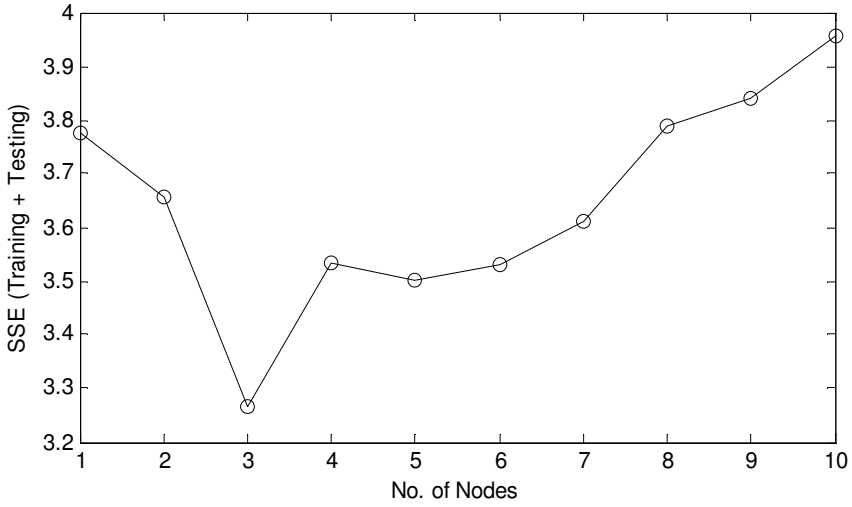


Fig. 3: Plot of total SSE for (training + testing) data of tank level prediction versus number of nodes.

After determined the number of nodes, the m-file for tank level prediction will be simulated to obtain the results of sum square error, SSE and also mean sum square error, mean SSE for neural networks system. The method of calculation of SSE and mean SSE in multiple neural networks are different. For SSE in multiple networks, the predicted output value, y_p for each network will be combined to obtain the mean predicted output value, $\overline{y_p}$.

$$\overline{y_p} = \frac{\sum_{i=1}^{n} y_{p,i}}{n} \tag{7}$$

where, n = number of network. This combined output value of $\overline{y_p}$ will be used to compare with the true output value, y to obtained the sum square error, SSE.

$$SSE = (y - \overline{y_p})^2 \tag{8}$$

For mean SSE in multiple neural networks, the predicted output value, yp for each network will be used to compare with the true output value, y. This means the sum square error is obtained for each network in the multiple networks system. The mean value of SSE for the networks will be determined and describe as mean SSE.

Mean SSE, $$\overline{SSE} = \frac{\sum_{i=1}^{n} SSE_{p,i}}{n} \tag{9}$$

The results of SSE and mean SSE for tank level prediction are tabulate in Table 2 and 3. While Fig. 4 and Fig. 5 show the plots of the results.

Table 2: Results of SSE for training, testing and validation data for tank level prediction.

No. of Networks	SSE Training Data	SSE Testing Data	SSE for (Training + Testing)	SSE Validation Data
1	1.6273	1.6395	3.2668	3.2311
5	1.6066	1.6948	3.3014	3.2922
10	1.5788	1.6224	3.2012	3.1671
15	1.5622	1.6219	3.1841	3.161
20	1.5642	1.6268	3.191	3.1554
25	1.5624	1.6123	3.1747	3.1387
30	1.5649	1.6062	3.1711	3.1363
35	1.5651	1.6019	3.167	3.1345
40	1.5626	1.5909	3.1535	3.1328
45	1.5629	1.5882	3.1511	3.1338
50	1.5627	1.5898	3.1525	3.1338

Table 3: Results of mean SSE for training, testing and validation data for tank level prediction.

No. of Networks combined	Mean SSE Training Data	Mean SSE Testing Data	Mean SSE (Training + Testing) Data	Mean SSE Validation Data
5	1.6967	1.7849	3.4816	3.5579
10	1.6826	1.716	3.3986	3.4317
15	1.6695	1.7153	3.3848	3.4075
20	1.6645	1.7126	3.3771	3.373
25	1.6615	1.6959	3.3574	3.3473
30	1.6595	1.6821	3.3416	3.3347
35	1.6586	1.6727	3.3313	3.308
40	1.6656	1.667	3.3326	3.349
45	1.6715	1.6687	3.3402	3.3615
50	1.6681	1.6659	3.334	3.354

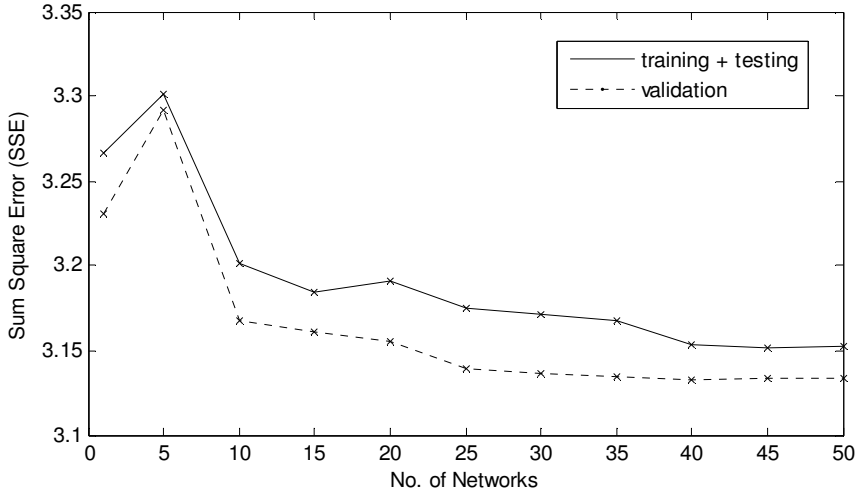


Fig. 4: Plot of SSE for (training + testing) data and validation data versus number of networks for tank level prediction.

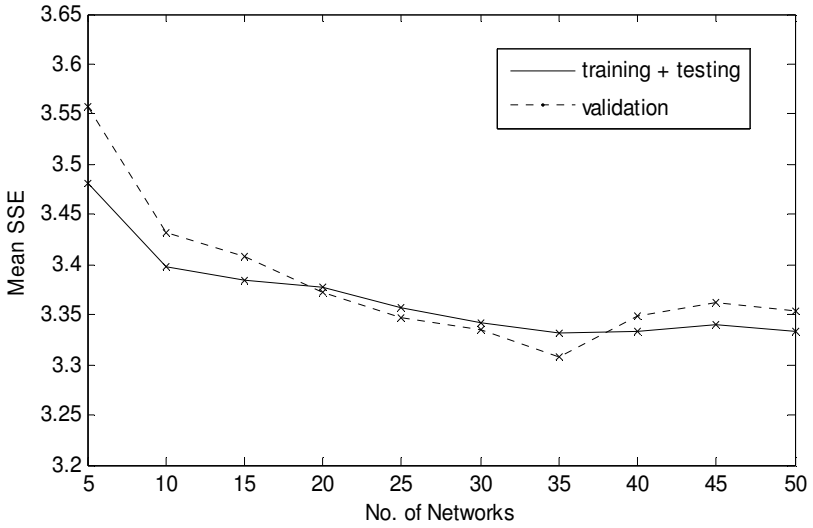
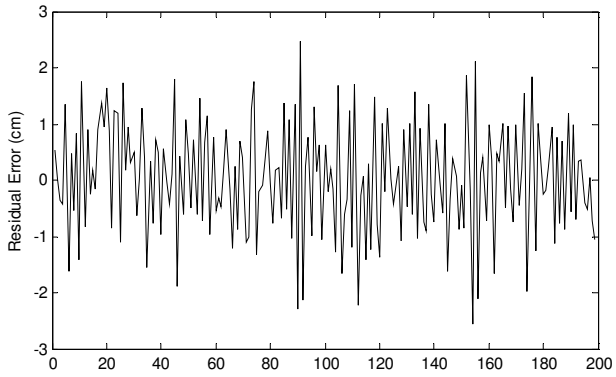


Fig. 5: Plot of mean SSE for (training + testing) data and validation data versus number of networks for tank level prediction.

From the results show in Fig. 4, the SSE for multiple neural networks are generally lower compare to single network. As compare to single network, the SSE value increase when 5 networks is used for multiple networks system. Beyond 10 networks, the SSE values slowly decrease until almost steady at 35 networks.

From Fig. 5, the mean SSE also decrease with the increase in number of networks. The optimum number of networks with minimum mean SSE for validation data achieve at 35 networks. The mean SSE value for validation increase slightly when more than 35 networks are used. The overall results show that multiple neural networks provide good performance with lower SSE compare to single network. The optimum number of network



achieves at 35 networks. For further analysis, the plot of residual error of validation data for multiple (35) networks will be conducted. The result of plot is shown in Fig. 6.

Fig. 6: Plot of residual error of validation data for multiple neural networks for tank level prediction.

The residual error is the difference between true output value, y and predicted output value, y_p:

$$\text{Residual error} = y - y_p \tag{10}$$

The true and predicted output values are rescaled to its original value (level, cm) before finding the residual error. From Fig. 6, the residuals appear randomly scattered around zero indicating that the model describes the data well. The range of error is from -2.5 to 2.5 cm. There are total 199 sets of validation data use to compare with the predicted output values. To prove how well the multiple neural networks predict the output data, we can plot a graph to relate between the true and predicted output values. Fig. 7 shows the results of the plot.

The result of plots shows that the predicted output values were close to the true output values. The calculated correlation coefficient value R^2 for true and predicted output for Fig. 7 for multiple networks is **0.9893**. Correlation coefficient is a normalized measure of linear relationship strength between two variables. The perfect fit will obtain a correlation coefficient value of 1. This means the results for the tank level prediction case study is close to the perfect fit value and indicate the multiple networks predict a good result.

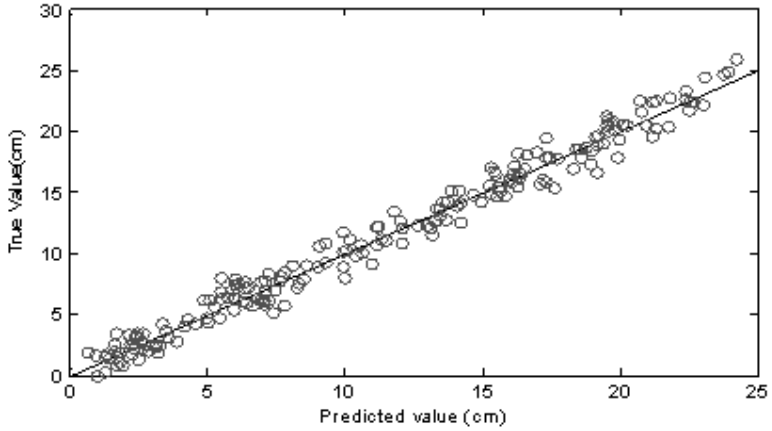


Fig. 7: Plot of true output values versus predicted output values for multiple neural networks for tank level prediction.

5. CONCLUSION

As mention in earlier part that more accurate representation of neural networks is very crucial as it contributes to the improvement of model performance especially when it is applied to unseen data. Therefore, major problem in modeling neural networks such as over fitting and under fitting can be avoided and generalization capability of the model can be enhanced. Driven by such convincing result, a step has been taken into reaching the goal of modeling neural networks. One of the methods that have been utilized is by studying the optimum number of neural networks that eligible to be combined.

By implementing the method, researcher can predict the intricacy of the model then finally obtained the best number of neural networks that should be combined. The overall result of this research has proposed 35 networks where the SSE values reach the almost steady condition. The combination of 35 networks also has been proved be reliable since the result shows that the predicted output values closed to the true output values.

ACKNOWLEDGEMENT

This research is supported by Universiti Sains Malaysia through RU grant 811106.

REFERENCES

[1] Hertz J, Krogh A and Palmer RG: Introduction to the Theory of Neural Computation. Addison-Wesley; 1991.
[2] Haykin S: Neural Networks. Macmillan College Publicating Company; 1994
[3] Bishop C: Neural Networks for Pattern Recognition.Clarendon Press; 1995
[4] Grossberg S: Neural Networks and Natural Intelligence. MIT Press; 1988
[5] Hinton G E: How Neural Networks Learn from Experience. Scientifc American 1992, 267:144-151.
[6] English E: MStacked Generalisation and Simulated Evolution. BioSystem 1996, 39: 3-18.
[7] Chen K, Xu L and Chi H: Improved Learning Algorithm for Mixture of Expert in Multiclass Classification. Neural Networks 1999, 12: 1229 – 1252.
[8] Crucianu M, Bone R and Asselin de Beauville JP: Bayesian Learning for Re-current Networks. Neurocomputing 2001, 36: 235-242.

[9] Girosi F, Jones M and Poggio T: Regularisation Theory and Neural Networks Architecture. Neural Computation 1995, 7:219-269.

[10] Morgan N and Bourlard H: Generalisation and Parameter Estimation in Feedforward Nets: Some Experiments. D. S. Touretzkey (Ed). In Advances in Neural Information Processing System 2,. San Mateo CA, 1990, :630-637.

[11] Ohbayashi M, Hirasawa K, Toshimitsu K, Murata J and Hu J: Robust Control for Non-linear System by Universal Learning Networks Considering Fuzzy Criterion and Second Order Derivatives. In the Proceeding IEEE World Congress on Computational Intelligence 1998.

[12] Caruana R, Lawrence S and Lee Giles C: Overfitting in Neural Networks: Backpropagation, Conjugate Gradient and Early Stopping. Neural Information Processing System 2000, 13:402-408.

[13] Mc Loone S and Irwin G: Improving Neural Networks Training Solution Using Regularisation. Neurocomnputing 2001, 37: 71-90.

[14] Patterson D W: Artificial Neural Networks Theory and Application. Prentice Hall; 1996.

[15] Willis M J, Di Massimo C, Montaque G A, Tham M T and Morris A J: Artificial Neural networks in Process Estimation and Control. Automatica 1992, 28 (6):1181-1187.

[16] Guyon X and Yao J: On the Underfitting and Overfitting Sets of Model Chosen by Order Selection Criteria. Journal of Multivariate Analysis 1999, 70:221-249.

[17] Hagiwara K and Kuno K: Regularisation Learning and Early Stopping in Linear Networks. International Joint Conference on Neural Networks 2000, :511-516.

[18] Radford M N: Bayesian Learning for Neural Networks. Statistics 1997, :118-183.

[19] Zhang J, Morris A J and Martin E B: Long-term Prediction Models Based on Mixed Order Locally Recurrent Neural Networks. Computers and Chemical Engineering .1998, 22 (7-8): 1051-1063.

[20] Zhang J, Morris A J, Martin E B and Kiparissides C: Prediction of Polymer Quality in Batch Polymerisation Reactors Using Robust Neural Networks. Chemical Engineering Journal 1998, 69:135-143

[21] Wolpert D H: Stacked Generalisation. Neural networks 1992, 5:241-259.

[22] Eikens B and Karim M N: Process Identification with Multiple Neural Networks Models. International Journal of Control 1999, 72(7/8):576-590.

[23] Chen L and Narendra K S: Intelligent Control using Multiple Neural Networks. International Journal of Adaptive Control and Signal Processing 2003, 17:417-430.

[24] Nguyen H H and Chan C W: Multiple Neural Networks for a Long term Time Series Forecast. Neural Computation and Application 2004, 13:90-98.

[25] Hashem S: Optimal Linear Combination. Neural Networks 1997, 10(4):599-614.

[26] Jacobs R A M I J, Nowlan S J and Hinton, G E: Adaptive Mixture of Local Expert. Neural Computation 1991, 3:79-87.

[27] Jordan M I and Jacobs R A: Hierarchical Mixtures of Expert and the EM Algorithm. Neural Computation 1994, 6:191-214.

[28] Giacinto G and Roli F: Design oo Effective Neural Network Ensembles for Image Classification Purposes. Image and Vision Computing 2001, 19:699-707.

[29] Cho S B and Lee J H: Learning Neural Network Ensemble for Practical Text Classification. J. Liu (Ed). In Lectures Notes in Computer Science. Berlin Heidelberg, Springer-Verlag 2003, :1032-1036.

[30] Jerebko A K, Malley J D, Franaszek M and Summers R M: Multiple Neural Networks Classification Scheme for Detection of Colonic Polyps in CT Colonography Data set. Academic Radiology 2003, 10(2):154-160.

[31] Hayashi Y and Setiono R: Combining Neural Network Prediction for Medical Diagnosis. Computers in Biology and Medicine 2002, 32:237-246.

[32] Zhou Z H, Y B Jiang and Chen S F: Lung Cancer Cell Identification Based on Artificial Neural Networks Ensembles. Artificial Inteligent in Medicine 2002, 24:26-36.

[33] Zhang J: Developing Robust Non-linear Models Through Bootstrap Aggregated Neural Networks. Neurocomputing 1999, 25:93-113.

[34] Cunningham P, Carney J and Jacob S: Stability Problems with Artificial Neural Networks and The Ensembles Solutions. Artificial Inteligent in Medicine 2000, 20:217-225.

[35] Wehrens R, Putter H and Buyden L M C: The bootstrap: a tutorial,Chemometrics and Intelligent Laboratory System 2000, 54:35-52.

[36] Schwenk H and Bengio Y: Boosting Neural Networks. Neural Computation 2000, 12:1869-1887.

[37] Freund Y and Schapire R E: Experiment with a new Boosting Algorithm. In the Proceeding 13th International Conference on Machine Learning 1996:148-156.

[38] Dietterich T G: An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. Machine Learning 2000, 40(2):139 - 157.

[39] Sharkey A J C (Eds): Multi Nets System. In Combining Artificial Neural Nets Ensemble and Modular. London: Springer Publication; 1999.