Neural Network Regression Based on Falsity Input

Pawalai Kraipeerapun and Somkid Amornsamankul

***Abstract*— In general, only the truth input is used to train neural network. This paper applies both truth and falsity input, which is the complement of the truth input, to train neural network to solve regression problems. Four neural networks are created. The first two networks are trained using the truth input to predict the truth and falsity outputs based on the truth and falsity targets, respectively. The last two are trained using the falsity input to predict the truth and falsity outputs as well. In order to add more diversity, ensemble of neural networks is applied. Each component in the ensemble contains four types of neural networks created based on our proposed techniques. Aggregation techniques are proposed to provide more accuracy results. Three classical benchmark data sets from the UCI machine learning repository are used in our experiments. These data sets are housing, concrete compressive strength, and computer hardware. It is found that the four proposed networks improve the prediction performance when compared to backpropagation neural network and complementary neural networks.**

***Keywords*— Feedforward Backpropagation Neural Network, Ensemble Neural Networks, Complementary Neural Networks, Regression Problems, Truth Neural Network, Falsity Neural Network**

I. Introduction

ONE of popular machines used to solve regression problems is neural network since it is found to give better accuracy results than statistical methods in various problem areas [1], [2], [3], [4], [5]. It is also found to provide better performance when compared to support vector regression (SVR) in many applications such as artificial nose regression problem [6], stock price prediction [7], water demand prediction [8], and approximation of the function with gaussian function and morlet wavelet RBF function [9].

Over the past year, neural networks have been used to solve several regression problems such as typhoon losses [10], technical target setting in QFD for Web service systems [11], wheat stripe rust [12], macroscopic water distribution system modeling [13], travel behavior analysis [14], autumn flood season in Danjiangkou reservoir basin [15], sheep growth prediction [16], software development effort estimation [17], and calibration of near-infrared spectra [18].

Several techniques have been used to improve the performance of neural network. For example, neural network was integrated with marginalized output weights to provide probabilistic predictions and to improve on the performance of sparse gaussian processes at the same computational cost as the traditional neural networks [19].

P. Kraipeerapun is with Department of Computer Science, Faculty of Science, Ramkhamhaeng University, Bangkok, Thailand (email: pawalai@yahoo.com)

S. Amornsamankul (corresponding author) is with Department of Mathematics, Faculty of Science, Mahidol University, and is with Centre of Excellence in Mathematics, CHE, Bangkok, Thailand (email: scsam@mahidol.ac.th)

In [20], neural network was designed to handle small training sets of high dimension by using a statistically based methodology. In [21], the number of nodes in one-hidden layer feedforward neural network were chosen based on the adaptive stochastic optimization and a linear regression method.

Hou and Han [9] proposed a constructive approximation in which a single hidden layer decay RBF neural network with n+1 hidden neurons can interpolate n+1 multivariate samples and can approximate any multivariate functions without training. Their experiments demonstrated the faster convergence and better performance than support vector machines.

Vizitiu et al. [22] proposed the use of full-genetic approach to train RBF neural networks. It assured optimization both connectivity and neural weights of neural networks.

In [23], two implication rules in the truth table were applied to neural networks in order to increase performances of the prediction results. It was found to provide better performance when compared to backpropagation neural network and support vector regression with linear, polynomial, and radial basis function kernels. This technique was named complementary neural networks (CMTNN).

In this paper, our proposed techniques will be created based on the concept of CMTNN and the ensemble technique. The ensemble of neural networks was proof to provide better results than using a single neural network in various application areas [24]. There are several ways to design an ensemble. One of the most popular and easiest ways is the use of bagging technique [25]. Therefore, we will apply bagging to CMTNN. The aim of this paper is to apply falsity input to CMTNN in order to enhance the prediction performance of neural network. The rest of this paper is organized as follows. Section II explains the basic concept of complementary neural network. Section III describes the concept of applying falsity input to complementary neural networks to solve single output regression problems. Section IV describes data sets and results of our experiments. Conclusions and future works are presented in Section V.

II. Complementary Neural Networks

Complementary neural networks (CMTNN) consist of a pair of neural networks in which both networks have the same parameter values and they are trained using the same truth input. However, one network is trained using the falsity target value instead of the truth target value that is used to train another network. The falsity target value is the complement of the truth target value such as 0.2 and 0.8 for the falsity and the truth target values, respectively. Both networks are created based on the truth table for implication as shown in Table I.

Let A and B be the input and the target of neural network, respectively. The training process of CMTNN are considered as the implication "$A \rightarrow B$". The first two implication rules

TABLE I
LOGICAL IMPLICATION

Premise A	Conclusion B	Inference $A \rightarrow B$
True	True	True
True	False	False
False	True	True
False	False	True

shown in Table I are applied to CMTNN. The first implication rule is that if both A and B are true then the inference is true. This rule is applied to the first network. It means that if the network is trained using the truth input and the truth target then we get the truth output. The second implication rule is applied to the second network in which if A is true but B is false then the inference is false. This means that if the network is trained using the truth input and the falsity target then we get the falsity output. Let T_{target} and F_{target} be the truth and falsity target values. The falsity target value is considered as the complement of the truth target value. It can be computed as $1 - T_{target}$. From these two neural networks, the falsity output should be complement to the truth output.

In the testing phase, let $T(x_i)$ and $F(x_i)$ be the truth and falsity output values obtained from the first and the second networks, respectively. Both values are predicted from the input pattern $x_i; i = 1, 2, 3, ..., n$ where n is the total number of input patterns in the testing phase. The combined output $O_1(x_i)$ can be computed as follows.

$$O_1(x_i) = \frac{T(x_i) + (1 - F(x_i))}{2} \quad (1)$$

In this paper, this technique is called CMTNN v.1. It was found that the combination of the truth output and the non-falsity output obtained from both network provides better result when compared to backpropagation neural network (BPNN), and support vector regression (SVR) with linear, polynomial, and radial basis function kernels [23].

III. APPLYING FALSITY INPUT TO COMPLEMENTARY NEURAL NETWORKS

The third implication rule shown in Table I is applied to the proposed neural network. If A is false but B is true then the inference is true. This means that if the network is trained using the falsity input and the truth target then we get the truth output. Two novel techniques are proposed and described below.

- CMTNN v.2
  Instead of considering the truth and falsity target, only the truth target is used in this technique. However, we consider the truth and falsity input instead. Figure 1 shows CMTNN v.2 in the training phase. Two neural networks having the same architecture and parameter values are trained to predict degree of truth values. The first neural network is trained using the truth input whereas the second neural network is trained using the falsity input. This technique conforms to the first and the third implication rules shown in Table I.
  Figure 2 shows CMTNN v.2 in the testing phase. Let $T_t(x_i)$ and $T_f(y_i)$ be the truth outputs obtained from the first and the second neural networks, respectively. $T_t(x_i)$ is the output obtained from the input pattern $x_i; i = 1, 2, 3, ..., n$ where n is the total number of input patterns in the testing phase. $T_f(y_i)$ is the output obtained from the input pattern y_i where $y_i = 1 - x_i$. In this case, y_i is considered as another format of x_i. The final output can be computed as follows.

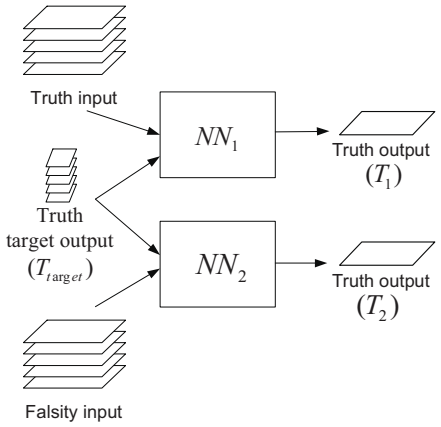

Fig. 1. Complementary Neural Networks v.2 (Training Phase)

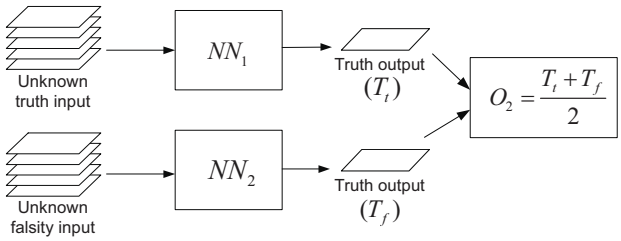

Fig. 2. Complementary Neural Networks v.2 (Testing Phase)

$$O_2(x_i) = \frac{T_t(x_i) + T_f(y_i)}{2} \quad (2)$$

- CMTNN v.3
  Similar to the previous technique, both truth and falsity inputs are applied. In order to improve the performance, the truth input is applied to a pair of neural networks that are trained to predict the truth and falsity outputs. On the other hand, the falsity input is applied to another pair of neural networks that are also trained to predict the truth and falsity output. It can be seen that the first pair of network is CMTNN v.1. All four neural networks have the same architecture and parameter values. Figure 3 shows CMTNN v.3 in the training phase.
  Figure 4 shows CMTNN v.3 in the testing phase. Let $T_{tt}(x_i)$ and $F_{tf}(x_i)$ be the truth and falsity outputs obtained from the first and the second neural networks, respectively. Both outputs are obtained from the input pattern $x_i; i = 1, 2, 3, ..., n$ where n is the total number of input patterns in the testing phase. Let $T_{ft}(y_i)$ and $F_{ff}(y_i)$ be the truth and falsity outputs obtained from the third and the fourth neural networks, respectively. Both outputs are obtained from the input pattern y_i where

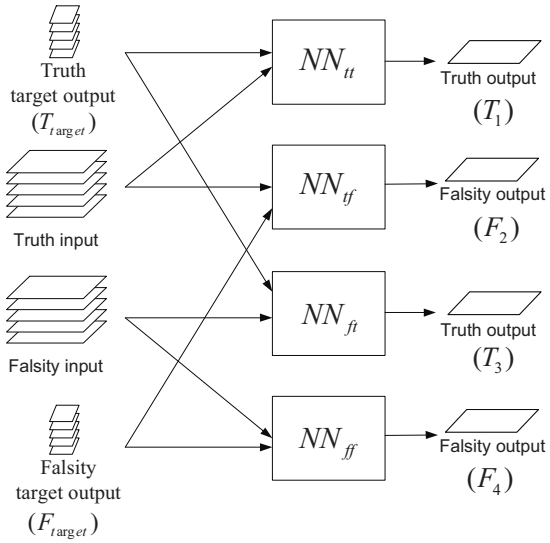

Fig. 3. Complementary Neural Networks v.3 (Training Phase)

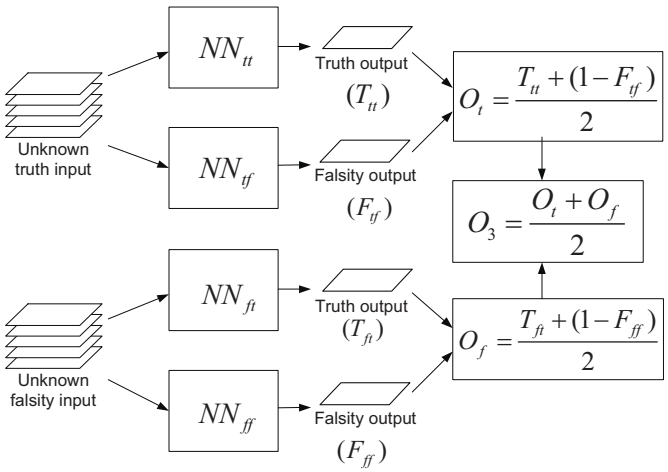

Fig. 4. Complementary Neural Networks v.3 (Testing Phase)

$y_i = 1 - x_i$. The combined output can be computed as follows.

$$O_t(x_i) = \frac{T_{tt}(x_i) + (1 - F_{tf}(x_i))}{2} \tag{3}$$

$$O_f(y_i) = \frac{T_{ft}(y_i) + (1 - F_{ff}(y_i))}{2} \tag{4}$$

$$O_3(x_i) = \frac{O_t(x_i) + O_f(y_i)}{2} \tag{5}$$

In our experiments in the next section, it is found that CMTNN v.3 provides the best results for all three data sets used in the experiments. Therefore, we decide to apply an ensemble technique to CMTNN v.3.

- Ensemble CMTNN v.3
  Bagging technique is used to create an ensemble. Therefore, bootstrap resampling is applied to the original training input in order to generate multiple training sets or bags. Each bag has the same size as the original size; however, some patterns may be repeated or some may not be included in the bag. Figure 5 shows an ensemble of complementary neural network v.3 in the training phase. Several bags are created based on the same original training input. For each bag, the truth input and the falsity input are created based on that bag. The truth input is the same set as the bag. The falsity input is the complement of the truth input. Let T^j_{train} be the truth training input of the j-bag. Let F^j_{train} be the falsity training input of the j-bag. F^j_{train} is created based on T^j_{train} as follows.

$$F^j_{train} = 1 - T^j_{train} \tag{6}$$

Similar to CMTNN v.3, each bag is trained to predict four outputs which are two truth outputs and two falsity outputs. In the testing phase, Let $T^j_{tt}(x_i)$ and $F^j_{tf}(x_i)$ be the truth and falsity outputs obtained from the first and the second neural networks based on the truth input pattern x_i of the j-component. Let $T^j_{ft}(y_i)$ and $F^j_{ff}(y_i)$ be the truth and falsity outputs obtained from the third and the fourth neural networks based on the falsity input pattern y_i of the j-component where $y_i = 1 - x_i$. Figure 6 shows the testing model of our proposed ensemble CMTNN v.3. The output obtained from each component can be computed as follows.

$$O^j_t(x_i) = \frac{T^j_{tt}(x_i) + (1 - F^j_{tf}(x_i))}{2} \tag{7}$$

$$O^j_f(y_i) = \frac{T^j_{ft}(y_i) + (1 - F^j_{ff}(y_i))}{2} \tag{8}$$

$$O^j_{tf}(x_i) = \frac{O^j_t(x_i) + O^j_f(y_i)}{2} \tag{9}$$

Instead of using all components in the ensemble, only the appropriate ensemble members are chosen. That is only components containing low uncertainty values should be selected. In this research, we consider uncertainty in each component in three aspects. First, the difference between the gap of the truth output and the falsity output based on the truth and falsity inputs is considered. Let U^j_1 be this type of uncertainty of the j-component. U^j_1 can be computed as follows.

$$U^j_1 = \frac{\sum_{i=1}^{n}((T^j_{tt}(x_i) - F^j_{tf}(x_i)) - (T^j_{ft}(y_i) - F^j_{ff}(y_i)))}{n} \tag{10}$$

Second, the difference between the truth outputs based on the truth and falsity inputs is considered. Let U^j_2 be this type of uncertainty of the j-component. U^j_2 can be computed as follows.

$$U^j_2 = \frac{\sum_{i=1}^{n}(T^j_{tt}(x_i) - T^j_{ft}(y_i))}{n} \tag{11}$$

Third, the difference between the falsity outputs based on the truth and falsity inputs is considered. Let U^j_3 be

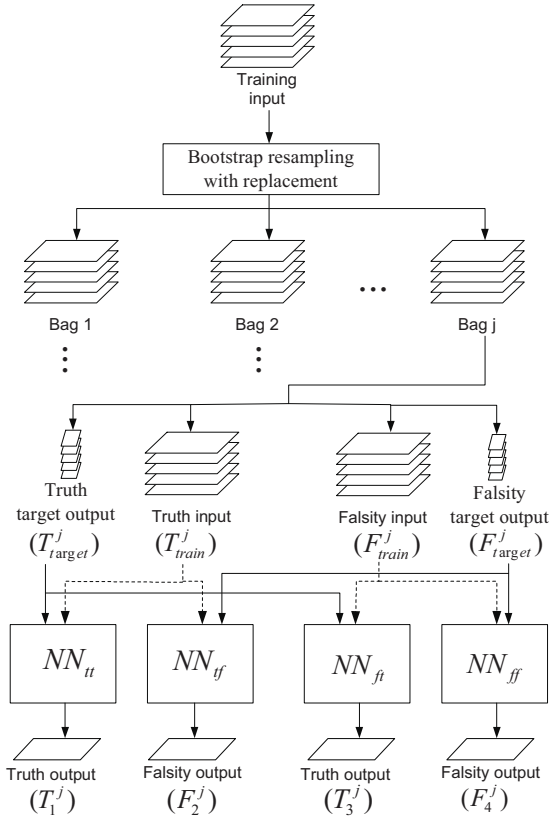

Fig. 5. Ensemble of Complementary Neural Networks v.3 (Training Phase)

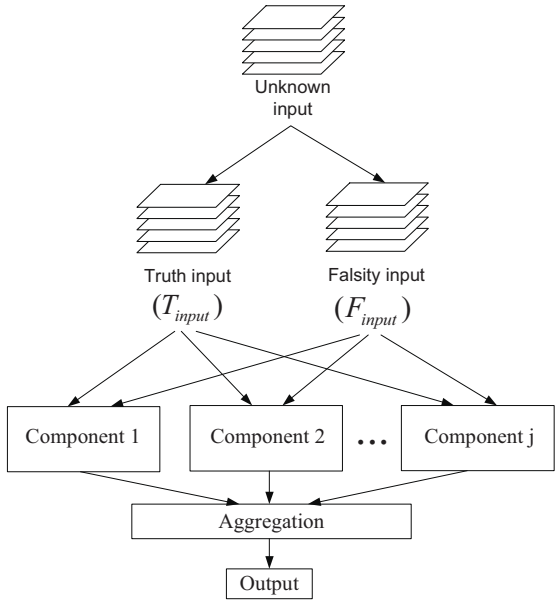

Fig. 6. Ensemble of Complementary Neural Networks v.3 (Testing Phase)

TABLE II
UCI DATA SETS USED IN THIS STUDY

Name	Feature type	No. of features	No. of samples
Housing	numeric	13	506
Concrete	numeric	8	1030
Hardware	numeric	6	209

this type of uncertainty of the j-component. U_3^j can be computed as follows.

$$U_3^j = \frac{\sum_{i=1}^{n}(F_{tf}^j(x_i) - F_{ff}^j(y_i))}{n} \tag{12}$$

The average of these three uncertainty values can be used as uncertainty indicator of each component in the ensemble. Let U^j be the average uncertainty value of the j-component. U^j can be computed as follows.

$$U^j = \frac{U_1^j + U_2^j + U_3^j}{3} \tag{13}$$

Only components having low uncertainty values will be selected. If m components are selected based on low uncertainty, all outputs obtained from those m components can be aggregated as follows.

$$O_4(x_i) = \frac{\sum_{j=1}^{m} O_{tf}^j(x_i)}{m} \tag{14}$$

IV. EXPERIMENTS

A. Data Sets

In the experiment, we apply the same benchmarking UCI data sets [26] used in [23] and [27]. The experimental results are also compared to results obtained from those two papers. These data sets are housing, concrete compressive strength, and computer hardware. The characteristics of these data sets are shown in the following table.

B. Experimental Methodology and Results

We separate the experiment into two parts. The first part deals with CMTNN v.2 and v.3. The second part copes with ensemble CMTNN v.3. In the first part, ten-fold cross validation are applied to each data sets. Four types of feed-forward backpropagation neural networks (BPNN) are created for each fold. The first neural network is trained using the truth input and the truth target to predict the truth output. This network is actually a traditional BPNN. The second neural network is trained using the truth input and the falsity target to predict the falsity output. The third and the fourth neural networks are trained using the falsity input; however, the third network is trained using the truth target where as the fourth network is trained using the falsity target to predict the truth output and falsity output, respectively.

For each data set, all neural networks are created based on the same architecture and parameter values. The number of input-nodes for each network is equal to the number of input features for each training set. Each network has one hidden layer constituting of $2m$ neurons where m is the number of input features.

For each fold, the first and the second neural networks are applied to CMTNN v.1. The first and the third neural networks are applied to CMTNN v.2. All four neural networks are applied to CMTNN v.3. Table III shows characteristics of these techniques. It can be seen that all techniques apply truth input and truth output. Also, they all apply single output

TABLE III

CHARACTERISTICS OF TECHNIQUES USED IN THIS PAPER

Characteristics	BPNN	CMTNN v.1	proposed CMTNN v.2	proposed CMTNN v.3
Truth input	x	x	x	x
Truth output	x	x	x	x
Falsity input			x	x
Falsity output		x		x
Single output NN	x	x	x	x

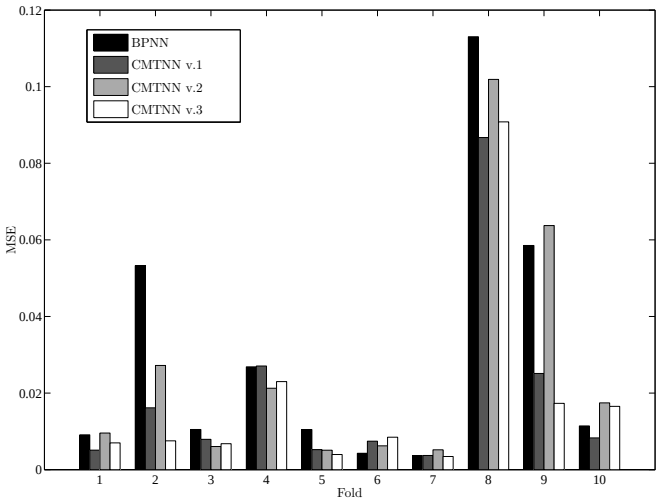

Fig. 7. Mean square error obtained from each fold of housing data set

neural network. Falsity output is applied to CMTNN v.1 and v.3. Falsity input is applied to CMTNN v.2 and v.3.

Fig. 7, 8, and 9 shows the comparison of mean square error obtained from CMTNN v.2 and v.3 compared to results obtained from existing techniques: BPNN and CMTNN v.1 presented in [23] for each fold of housing, concrete, and hardware data sets, respectively. Table IV shows average accuracy results obtained from each technique based on each data set. Fig 10 portrays the comparison among average accuracy results for each data set. From the average accuracy results, it can be noted that all types of CMTNN provide better results than BPNN. CMTNN v.3 provides the best results for all data sets. Table V shows the percent improvement of the proposed CMTNN v.3 compared to other techniques.

In the second part, each data set is split into a training set containing 80% of the data and a testing set containing 20% of the data. Bagging technique is applied to each data set. In this experiment, all parameters assigned to neural networks

TABLE IV

THE AVERAGE OF MEAN SQUARE ERROR, MSE, (TEN FOLDS) OBTAINED FROM THE TEST DATA SETS

Method	Housing	Concrete	Hardware
BPNN	0.030113	0.021595	0.005601
CMTNN v.1	0.019275	0.017384	0.004377
CMTNN v.2	0.026364	0.017133	0.004194
CMTNN v.3	0.018480	0.015225	0.003121

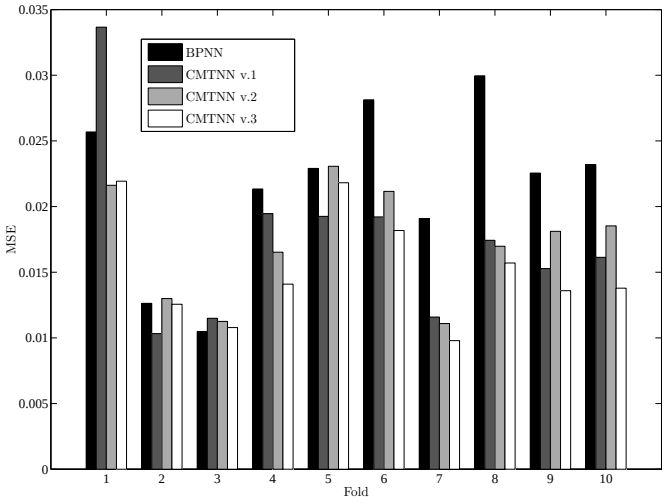

Fig. 8. Mean square error obtained from each fold of concrete data set

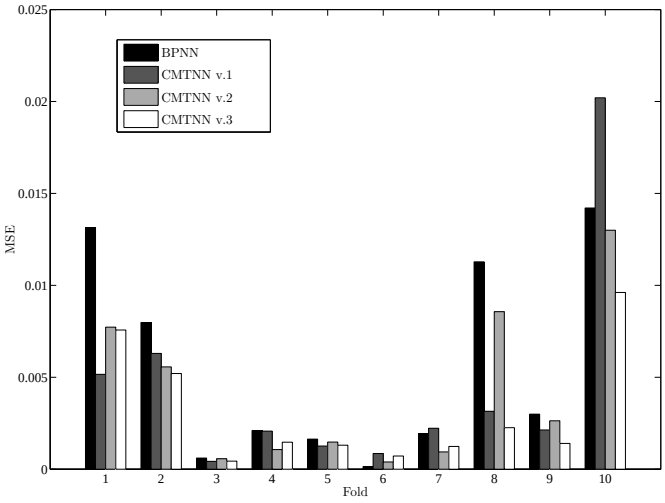

Fig. 9. Mean square error obtained from each fold of hardware data set

are set similar to the experiment in paper [27] for the reason of comparison. Thirty bags are created and thirty components of ensemble CMTNN v.3 are built. Each component contains four neural networks. The first and second networks apply the truth input whereas the third and fourth networks apply the falsity input. On the other hand, the first and the third networks predict the truth output whereas the second and the fourth networks predict the falsity output. From these components, only ten components having low uncertainty values computed using (13) are selected. Results obtained from these ten components are aggregated using (14). It can be seen that each network in our proposed technique predict a single output: either truth or falsity output. The result obtained from our proposed ensemble technique will be compared to other techniques, which are ensemble of BPNN, ensemble of CMTNN v.1, and ensemble of DONN [27].

TABLE V

THE PERCENT IMPROVEMENT OF THE CMTNN V.3 COMPARED TO OTHER TECHNIQUES.

Method	CMTNN v.3 (%improvement)		
	Housing	Concrete	Hardware
BPNN	38.63	29.50	44.28
CMTNN v.1	4.12	12.42	28.69
CMTNN v.2	29.90	11.14	25.58

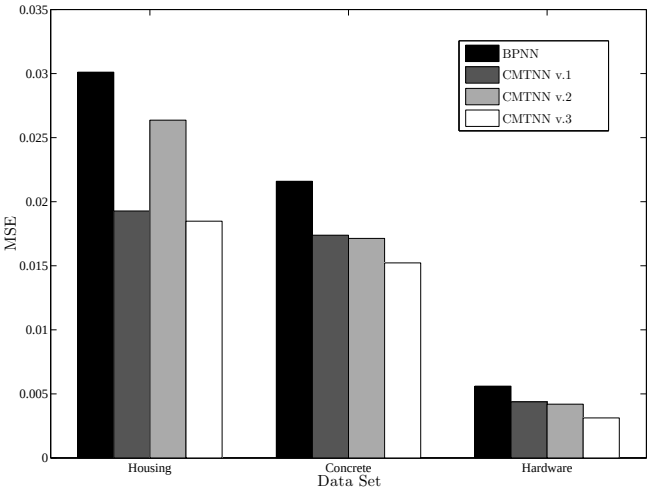

Fig. 10. Average of mean square error obtained from housing, concrete, and hardware data sets

DONN is a duo output neural network that predicts both truth and falsity output at the same time. This technique was created based on CMTNN. Each component in the ensemble of DONN contains two neural networks. The first network is trained to predict the truth and falsity outputs whereas the second neural network is trained to predict the falsity and truth outputs which are arranged in the opposite order of the first one. The result is computed from the combination of outputs obtained from both networks. This technique applies multiple outputs neural network to predict a single output regression problem. Table VI shows characteristics of ensemble of BPNN, CMTNN v.1, DONN, and CMTNN v.3.

Table VII shows average of mean square error (MSE) obtained from the test data set of housing, concrete, and hardware based on ensemble techniques. Table VIII shows the percent improvement of the proposed ensemble CMTNN v.3 compared to other techniques. Fig. 11, 12, and 13 portray the comparison among averages of mean square error obtained from the proposed ensemble technique and other techniques for housing, concrete, and hardware data sets, respectively.

It can be seen that ensemble CMTNN v.3 provides better results when compared to BPNN, ensemble of BPNN, CMTNN v.1-3, and ensemble of CMTNN v.1. Ensemble of DONN provides just a little bit better performance in some cases when compared to our proposed ensemble technique. However, the advantage of our proposed ensemble technique is that it has less complexity than the ensemble of DONN. For DONN, much effort has been dedicated to the experiment

TABLE VI

CHARACTERISTICS OF ENSEMBLE TECHNIQUES USED IN THIS PAPER

Characteristics	Ensemble of			
	BPNN	CMTNN v.1	DONN	proposed CMTNN v.3
Truth input	x	x	x	x
Truth output	x	x	x	x
Falsity input				x
Falsity output		x	x	x
Single output NN	x	x		x
Multiple outputs NN			x	

TABLE VII

THE MEAN SQUARE ERROR, MSE, BASED ON ENSEMBLE TECHNIQUES

Data set	Ensemble of			
	BPNN	CMTNN v.1	DONN	proposed CMTNN v.3
Housing	0.008398	0.008087	0.007663	0.007609
Concrete	0.019303	0.012462	0.010717	0.010837
Hardware	0.004419	0.001912	0.001575	0.001732

to deal with multiple outputs neural network. For CMTNN, only simple single output neural network is used. Also, only the complement technique is applied to the input and target values.

V. CONCLUSION

Instead of applying only the first two implication rules shown in Table I to the process of neural network training, the third implication rule is also applied in this paper. Four neural networks are trained. They conform to those three implication rules in which the first pair of neural networks conform to the first two implication rules whereas the third neural network is created to conform to the third implication rule. However, in order to improve the performance of the third neural network, the fourth neural network is created using the concept of traditional CMTNN to increase performance of the third neural network. Therefore, we have two pairs of neural networks. These two pairs can be considered as two CMTNNs in which the first one is the original CMTNN and the second one is the CMTNN created based on the falsity input. These neural networks contain different combination among the truth and falsity input and output. Hence, diversity in the prediction is increased. Therefore, the result of the aggregation of those four neural networks is found to provide better performance when

TABLE VIII

THE PERCENT IMPROVEMENT OF AN ENSEMBLE OF CMTNN V.3 (AVERAGING) COMPARED TO OTHER TECHNIQUES

Method	Ensemble of CMTNN v.3		
	Housing	Concrete	Hardware
BPNN	74.73	49.82	69.08
CMTNN v.1	60.52	37.66	60.43
CMTNN v.2	71.14	36.75	58.70
CMTNN v.3	58.83	28.82	44.50
Ensemble of BPNN	9.40	43.86	60.81
Ensemble of CMTNN v.1	5.91	13.04	9.41
Ensemble of DONN	0.70	-1.12	-9.97

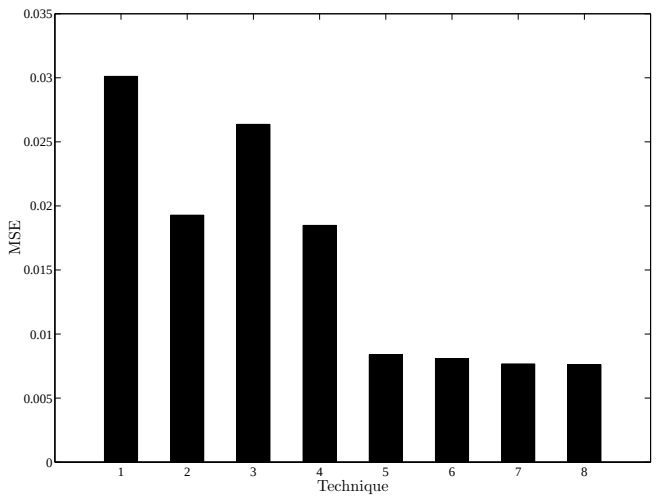

Fig. 11. Averages of mean square error for housing data set obtained from (1) BPNN (2) CMTNN v.1 (3) CMTNN v.2 (4) CMTNN v.3 (5) ensemble of BPNN (6) ensemble of CMTNN v.1 (7) ensemble of DONN (8) ensemble of CMTNN v.3

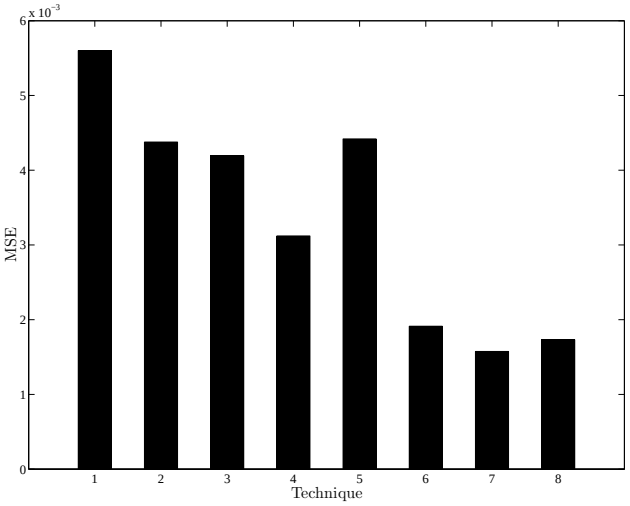

Fig. 13. Averages of mean square error for hardware data set obtained from (1) BPNN (2) CMTNN v.1 (3) CMTNN v.2 (4) CMTNN v.3 (5) ensemble of BPNN (6) ensemble of CMTNN v.1 (7) ensemble of DONN (8) ensemble of CMTNN v.3

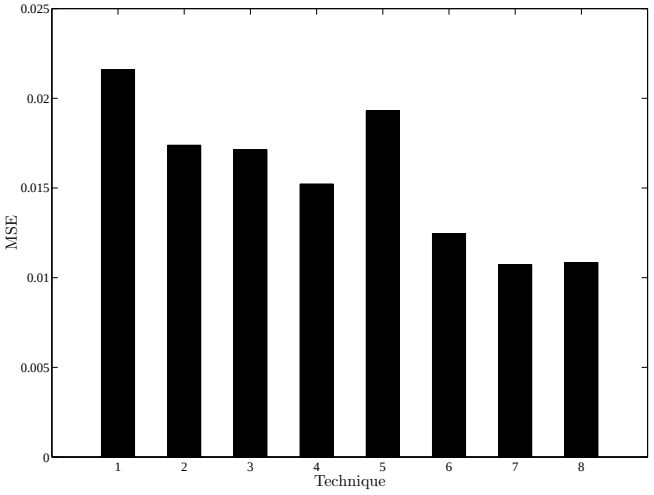

Fig. 12. Averages of mean square error for concrete data set obtained from (1) BPNN (2) CMTNN v.1 (3) CMTNN v.2 (4) CMTNN v.3 (5) ensemble of BPNN (6) ensemble of CMTNN v.1 (7) ensemble of DONN (8) ensemble of CMTNN v.3

compared to other techniques. Furthermore, bagging technique is also applied to increase diversity in neural network ensemble. The results show that our proposed ensemble techniques outperform other techniques. In the future, we will apply our approach to the classification problems.

ACKNOWLEDGMENT

This research project is supported by Faculty of Science, Mahidol University.

REFERENCES

[1] S. F. Crone, S. Lessmann, and S. Pietsch, "Forecasting with Computational Intelligence - An Evaluation of Support Vector Regression and Artificial Neural Networks for Time Series Prediction," in *Proceedings of International Joint Conference on Neural Networks*, Canada, July 2006, pp. 3159–3166.

[2] W.-H. Chen and J.-Y. Shih, "Comparison of support-vector machines and back propagation neural networks in forecasting the six major Asian stock markets," *International Journal of Electronic Finance*, vol. 1, no. 1, pp. 49–67, 2006.

[3] Y.-M. Wang, S. Traore, and T. Kerh, "Monitoring Event-Based Suspended Sediment Concentration by Artificial Neural Network Models," *WSEAS Transactions on Computers*, vol. 7, no. 5, pp. 559–568, May 2008.

[4] C. Botoca, R. Bardan, M. Botoca, and F. alexa, "Prostate Cancer Prognosis Evaluaion Assisted by Neural Networks," *WSEAS Transactions on Computers*, vol. 9, no. 2, pp. 164–173, February 2010.

[5] P. Hajek and V. Olej, "Municipal Revenue Prediction by Ensembles of Neural Networks and Support Vector Machines," *WSEAS Transactions on Computers*, vol. 9, no. 11, pp. 1255–1264, November 2010.

[6] S. Osowski, K. Siwek, and T. Markiewicz, "MLP and SVM Networks – a Comparative Study," in *Proceedings of the 6th Nordic Signal Processing Symposium*, June 2004, pp. 37–40.

[7] H. Ince and T. B. Trafalis, "Kernel Principal Component Analysis and Support Vector Machines for Stock Price Prediction," in *Proceedings of IEEE International Joint Conference on Neural Networks*, vol. 3, July 2004, pp. 2053–2058.

[8] I. S. Msiza, F. V. Nelwamondo, and T. Marwala, "Water Demand Prediction using Artificial Neural Networks and Support Vector Regression," *Journal of Computers*, vol. 3, no. 11, pp. 1–8, November 2008.

[9] M. Hou and Z. Han, "Constructive Approximation to Multivariate Function by Decay RBF Neural Network," *IEEE Transactions on Neural Networks*, vol. 21, no. 9, pp. 1517–1523, September 2010.

[10] W. Sun, S. Shan, C. Zhang, P. Ge, and L. Tao, "Prediction of Typhoon Losses in the South-East of China Based on B-P Network," in *Proceedings of 2010 International Conference on Artificial Intelligence and Computational Intelligence*, China, October 2010, pp. 252–256.

[11] L. Zhu and X. F. Liu, "Technical Target Setting in QFD for Web Service Systems Using an Artificial Neural Network," *IEEE Transactions on Services Computing*, vol. 3, no. 4, pp. 338–352, October-December 2010.

[12] L. Mo, "Prediction of Wheat Stripe Rust using Neural Network," in *Proceedings of 2010 IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS)*, China, October 2010, pp. 475–479.

[13] H. Wang and W. Guo, "ACO Optimizing Neural Network for Macroscopic Water Distribution System Modeling," in *Proceedings of 2010 International Conference on Intelligent Computing and Cognitive Informatics (ICICCI)*, Kuala Lumpur, June 2010, pp. 367–370.

[14] Z. Dan, S. Chunfu, Z. Nuo, and L. Yinhong, "Application of BP netework for travel behavior analyis: complexity recognition of trip chaining," in *Proceedings of 2010 International Conference on Intelligent Computation Technology and Automation (ICICTA)*, Changsha, May 2010, pp. 738–741.
[15] Y. Liu, Y. Chen, J. Hu, Q. Huang, and Y. Wang, "Long-term Prediction for Autumn Flood Season in Danjiangkou Reservoir Basin Based on OSR-BP Neural Network," in *Proceedings of 2010 Sixth International Conference on Natural Computation (ICNC 2010)*, Shandong, China, August 2010, pp. 1717–1720.
[16] M. R. B. Behzadi and A. A. Aslaminejad, "A Comparison of Neural Network and Nonlinear Regression Predictions of Sheep Growth," *Journal of Animal and Veterinary Advances*, vol. 9, no. 16, pp. 2128–2131, 2010.
[17] R. Bhatnagar, V. Bhattacharjee, and M. K. Ghose, "Software Development Effort Estimation - Neural Network Vs. Regression Modeling Approach," *International Journal of Engineering Science and Technology*, vol. 2, no. 7, pp. 2950–2956, 2010.
[18] A. Ukil, J. Bernasconi, H. Braendle, H. Buijs, and S. Bonenfant, "Improved calibratio of near-infrared spectra by using ensembles of neural network models," *IEEE Sensors Journal*, vol. 10, no. 3, pp. 578–584, March 2010.
[19] M. Lázaro-Gredilla and A. R. Figueiras-Vidal, "Marginalized neural network mixtures for large-scale regression," *IEEE transactions on neural networks*, vol. 21, no. 8, pp. 1345–1351, August 2010.
[20] J.-L. Yuan and T. L. Fine, "Neural-Network Design for Small Training Sets of High Dimension," *IEEE Transactions on Neural Networks*, vol. 9, no. 2, pp. 266–280, March 1998.
[21] B. Igelnik, Y.-H. Pao, S. R. LeClair, and C. Y. Shen, "The Ensemble Approach to Neural-Network Learning and Generalization," *IEEE Transactions on Neural Networks*, vol. 10, no. 1, pp. 19–30, January 1999.
[22] I.-C. Vizitiu, P. Ciotirnae, T. Oroian, A. Radu, F. Popescu, and C. Avram, "Training of RFB neural networks using a full-genetic approach," *WSEAS Transactions on Information Science and Applications*, vol. 7, no. 8, pp. 1015–1024, August 2010.
[23] P. Kraipeerapun, S. Nakkrasae, C. C. Fung, and S. Amornsamankul, "Solving regression problem with complementary neural networks and an adjusted averaging technique," *Memetic Computing*, vol. 2, no. 4, pp. 249–257, 2010.
[24] L. K. Hansen and P. Salamon, "Pattern Analysis and Machine Intelligence," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, October 1990, pp. 993–1001.
[25] L. Breiman, "Bagging Predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.
[26] A. Asuncion and D. Newman, "UCI machine learning repository," 2007. [Online]. Available: http://www.ics.uci.edu/∼mlearn/MLRepository.html
[27] S. Amornsamankul and P. Kraipeerapun, "Bagging of duo output neural networks for single output regression problem," in *Proceedings of 2010 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)*, vol. 7, Chengdu, China, July 2010, pp. 135–139.