Applying Falsity Input to Neural Networks to Solve Single Output Regression Problems

SOMKID AMORNSAMANKUL
Mahidol University
Faculty of Science
Department of Mathematics
Bangkok, THAILAND
and Centre of Excellence in Mathematics
CHE, Sriayudhaya Rd., Bangkok, Thailand
scsam@mahidol.ac.th

PAWALAI KRAIPEERAPUN
Ramkhamhaeng University
Faculty of Science
Department of Computer Science
Bangkok, THAILAND
pawalai@yahoo.com

Abstract: In this paper, both truth and falsity inputs are used to trained neural networks. Falsity input is the complement of the truth input. Two pairs of neural networks are created. The first pair of neural networks are trained using the truth input whereas the second pair of neural networks are trained using the falsity input. Each pair of neural networks are trained to predict degree of truth and degree of falsity outputs based on the truth and falsity targets, respectively. Two novel techniques are proposed based on these two pairs of neural network. We experiment our proposed techniques to three classical benchmark data sets, which are housing, concrete compressive strength, and computer hardware from the UCI machine learning repository. It is found that our proposed techniques improve the prediction performance when compared to backpropagation neural network and complementary neural networks.

Key–Words: Feedforward Backpropagation Neural Network, Complementary Neural Networks, Ensemble Neural Networks, Regression Problems, Truth Neural Network, Falsity Neural Network, Support Vector Regression

1 Introduction

Neural network is one of the most popular methods used to solve regression problems. Over the past year, neural networks have been used to solve several regression problems such as typhoon losses [13], technical target setting in QFD for Web service systems [17], wheat stripe rust [10], macroscopic water distribution system modeling [15], travel behavior analysis [4], autumn flood season in Danjiangkou reservoir basin [9], and calibration of near-infrared spectra [14].

Neural network is popular since it is found to provide better accuracy results than statistical methods in various problem areas [10, 15, 4, 9, 14, 3, 2]. It is also found to provide better performance when compared to support vector regression (SVR) in many applications such as artificial nose regression problem [12], stock price prediction [6], and water demand prediction [11].

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 [8].

In [16], neural network was designed to handle small training sets of high dimension by using a statistically based methodology. In [5], the number of nodes in one-hidden layer feedforward neural network were chosen based on the adaptive stochastic optimization and a linear regression method.

In [7], 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. The rest of this paper is organized as follows. Section 2 explains the basic concept of complementary neural network. Section 3 describes the concept of applying falsity input to complementary neural networks to solve single output regression problems. Section 4 describes data sets and results of our experiments. Conclusions and future works are presented in Section 5.

2 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 1.

Table 1: Logical implication

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

Let A and B be the input and the target of neural network, respectively. The training process of CMTNN are considered as the implication "$A \to B$". The first two implication rules shown in Table 1 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.

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

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 [7].

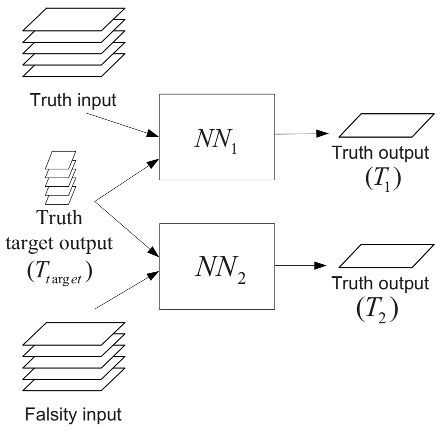

Figure 1: Complementary Neural Networks v.2 (Training Phase)

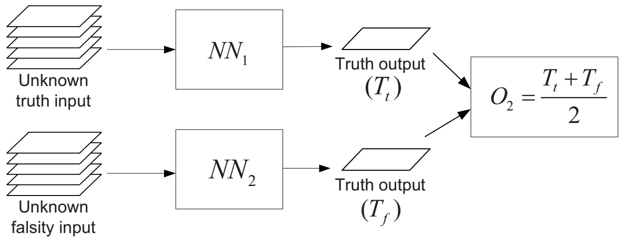

Figure 2: Complementary Neural Networks v.2 (Testing Phase)

3 Applying Falsity Input to Complementary Neural Networks

The third implication rule shown in Table 1 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

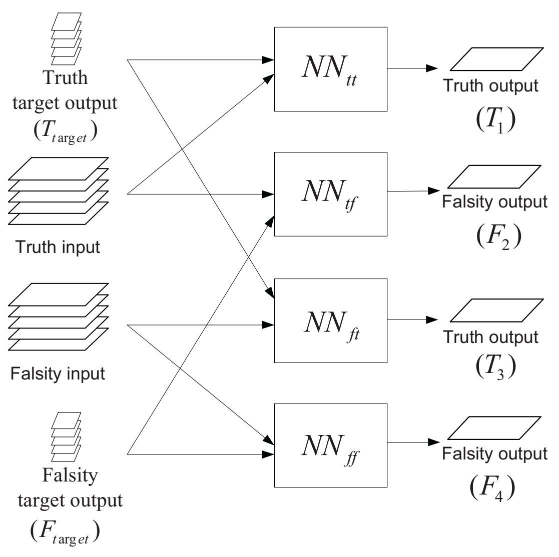

Figure 3: Complementary Neural Networks v.3 (Training Phase)

the second neural network is trained using the falsity input. This technique conforms to the first and the third implication rules shown in Table 1.

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.

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

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

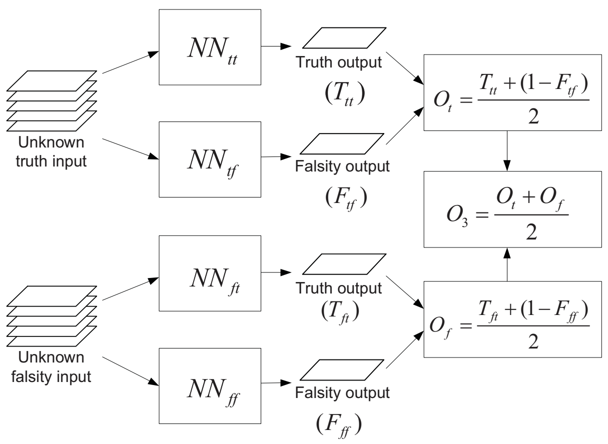

Figure 4: Complementary Neural Networks v.3 (Testing Phase)

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

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

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

4 Experiments

4.1 Data Sets

In the experiment, we apply three benchmarking UCI data sets [1] to test our proposed technique. These data sets are housing, concrete compressive strength, and computer hardware. Each data set is randomly split into a training set containing 80% of the data and a testing set containing 20% of the data. The characteristics of these data sets are shown in the following table.

Table 2: 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

Table 3: 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

4.2 Experimental Methodology and Results

We apply ten-fold cross validation to each data sets. Four types of feed-forward backpropagation neural networks 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 3 shows results obtained from the test set of housing, concrete, and hardware data. 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 4 shows the percent improvement of the proposed CMTNN v.3 compared to other techniques.

5 Conclusion

Instead of applying only the first two implication rules shown in Table 1 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 ensemble of those four neural networks is found to provide better performance when compared to other techniques. In the future, we will apply our approach to the classification problems.

Table 4: 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

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

References:

[1] A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.

[2] Wun-Hua Chen and Jen-Ying Shih. Comparison of support-vector machines and back propagation neural networks in forecasting the six major Asian stock markets. *International Journal of Electronic Finance*, 1(1):49–67, 2006.

[3] Sven F. Crone, Stefan Lessmann, and Swantje 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*, pages 3159–3166, Canada, July 2006.

[4] Zhao Dan, Shao Chunfu, Zhu Nuo, and Liu Yinhong. Application of BP netework for travel be-

havior analyis: complexity recognition of trip chaining. In *Proceedings of 2010 International Conference on Intelligent Computation Technology and Automation (ICICTA)*, pages 738–741, Changsha, May 2010.

[5] Boris Igelnik, Yoh-Han Pao, Steven R. LeClair, and Chang Yun Shen. The Ensemble Approach to Neural-Network Learning and Generalization. *IEEE Transactions on Neural Networks*, 10(1):19–30, January 1999.

[6] Huseyin Ince and Theodore B. Trafalis. Kernel Principal Component Analysis and Support Vector Machines for Stock Price Prediction. In *Proceedings of IEEE International Joint Conference on Neural Networks*, volume 3, pages 2053–2058, July 2004.

[7] Pawalai Kraipeerapun, Sathit Nakkrasae, Chun Che Fung, and Somkid Amornsamankul. Solving regression problem with complementary neural networks and an adjusted averaging technique. *Memetic Computing*, 2(4):249–257, 2010.

[8] Miguel Lázaro-Gredilla and Aníbal R Figueiras-Vidal. Marginalized neural network mixtures for large-scale regression. *IEEE transactions on neural networks*, 21(8):1345–1351, August 2010.

[9] Yong Liu, Yuanfang Chen, Jian Hu, Qin Huang, and Yintang 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)*, pages 1717–1720, Shandong, China, August 2010.

[10] Lihong Mo. Prediction of Wheat Stripe Rust using Neural Network. In *Proceedings of 2010 IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS)*, pages 475–479, China, October 2010.

[11] Ishmael S. Msiza, Fulufhelo V. Nelwamondo, and Tshilidzi Marwala. Water Demand Prediction using Artificial Neural Networks and Support Vector Regression. *Journal of Computers*, 3(11):1–8, November 2008.

[12] Stanislaw Osowski, Krzysztof Siwek, and Tomasz Markiewicz. MLP and SVM Networks – a Comparative Study. In *Proceedings of the 6th Nordic Signal Processing Symposium*, pages 37–40, June 2004.

[13] Wenbin Sun, Shigang Shan, Cuicui Zhang, Peipei Ge, and Liangliang 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*, pages 252–256, China, October 2010.

[14] Abhisek Ukil, Jakob Bernasconi, Hubert Braendle, Henry Buijs, and Sacha Bonenfant. Improved calibratio of near-infrared spectra by using ensembles of neural network models. *IEEE Sensors Journal*, 10(3):578–584, March 2010.

[15] Hongxiang Wang and Wenxian Guo. ACO Optimizing Neural Network for Macroscopic Water Distribution System Modeling. In *Proceedings of 2010 International Conference on Intelligent Computing and Cognitive Informatics (ICICCI)*, pages 367–370, Kuala Lumpur, June 2010.

[16] Jen-Lun Yuan and Terrence L. Fine. Neural-Network Design for Small Training Sets of High Dimension. *IEEE Transactions on Neural Networks*, 9(2):266–280, March 1998.

[17] Lianzhang Zhu and Xiaoqing (Frank) Liu. Technical Target Setting in QFD for Web Service Systems Using an Artificial Neural Network. *IEEE Transactions on Services Computing*, 3(4):338–352, October-December 2010.