

Finding the Capacity of Fuzzy Neural Networks (FNNs) via Its Equivalent Fully Connected Neural Networks (FFNNs)

Jing Wang
Faculty of Science and Technology
University of Macau
MSAR, People Republic of China
wj_adr@163.com

Chi-Hsu Wang, FIEEE
Department of Electrical Engineering
National Chiao Tung University
Hsinchu, Taiwan
cwang@cn.nctu.edu.tw

C. L. Philip Chen, FIEEE
Faculty of Science and Technology
University of Macau
MSAR, People Republic of China
philip.chen@ieee.org

Abstract—The capacity of Fuzzy Neural Network (FNN) is explored in this paper. The FNN is first transformed into an equivalent fully connected three layer neural network, or FFNN, via a new approach proposed in this paper. Then the lower and upper bounds of FNN can be found. To check the validity of the theoretical bounds, an example is illustrated with the trainings to yield excellent capacity matching with the theoretical bounds. This new finding has its emerging values in all engineering applications using FNN, such as intelligent adaptive control, pattern recognition, and signal processing, ..., etc.

Keywords—Neural Networks, Fuzzy Logic, Fuzzy Neural Networks, Back Propagations, Universal Approximation Theorem, Capacity of Neural Networks

I. INTRODUCTION

During the past decade, fuzzy neural networks (FNNs) have been proposed and successfully applied to solve problems in various fields, such as intelligent adaptive control, system identification, pattern classification and image processing, ..., etc [1-4]. FNN can utilize human experience and knowledge by incorporating linguistic fuzzy information. FNN consists of a set of if-then fuzzy rules. The rules are built up by human experts as linguistic knowledge. To obtain better performance, some training algorithms such as back-propagation (BP) algorithm is adopted to tune the parameters of fuzzy system and the weights of neural network. Training algorithm attempts to minimize the training error, which is the difference between the desired and actual outputs [5]. If FNN is properly constructed, it follows Universal Approximate Theorem (UAT), i.e., the properly constructed FNN can approximate any functions [6-7]. However, the UAT does not point out how to properly construct the fuzzy system and

neural network. This is to say that the FNN designed for certain applications by human experts cannot be ideal, which will lead to the discussion of the constraints of FNN, i.e., the capacity of FNN, so that engineering applications via FNN can be achieved. Similar to the discussions of the capacity of multilayer neural networks [8], the capacity of FNN is defined as the maximum number of desired input vectors that can be mapped to desired output vectors without error. This is also to say that the number of training samples has significant impact on training process. Excessive samples size may lead training process to be diverged. The capacity of associative memory and multilayer perceptron (MLP) has been conducted in the past decade, assuming a fully connected neural network [8-15]. However, the capacity of FNN has never been found in the literatures. This is due to fact that the FNN is not a fully connected directed graph and the intermediate layer is defined as a product layer, instead of a summing layer in a NN.

To apply the techniques in finding the capacity of three layers NN, a new approach is proposed to transform the four layers FNN into a fully connected directed graph with all summing nodes. This is achieved in this paper by also combining a systematic fuzzy rule generation for FNN [16] so as to obtain a fully indexed equivalent three layer NN for FNN, or FFNN. By using this new FFNN, the lower and upper bounds of the capacity of FNN can be found in this paper. An example is illustrated this paper to validate the theoretical bounds by back propagation trainings. Excellent matching is achieved between the theoretical bounds and training simulation.

II. THE FNN WITH SYSTEMATIC RULE GENERATION

Given the training input data vectors $\{x_q^*, q = 1, 2, 3, \dots, n\}$, and the desired output vectors $\{y_p^*, p = 1, 2, \dots, m\}$, the inference rules of simplified fuzzy reasoning can be established by the expert. The l^{th} control rule has the following form:

Rule l : IF x_1 is $A_1^{r_1}$ and and x_n is $A_n^{r_n}$ THEN y_1 is w_{11} and and y_m is w_{lm}

where l is a rule number, the $A_q^{r_i}$ are membership functions of antecedent part, and w_{lp} are real numbers of the consequent part. When the inputs are given, the truth value μ_l of the premise of the l^{th} rule is defined by

$$\mu_l = A_1^{r_1}(x_1) \times \dots \times A_n^{r_n}(x_n); \{l = 0, 1, 2, \dots, L; r_i = 0, 1, 2, \dots, R_{i-1}\} \quad (1)$$

where L is total number of fuzzy rules and R_i is the number of MFs for fuzzy variable x_i , and r_i is the ordinal index of MFs for fuzzy variable x_i . Among the commonly used defuzzification strategies, the simplified fuzzy reasoning yields a superior result. The output y_p of the fuzzy reasoning can be derived from the following equation:

$$y_p = \frac{\sum_i \mu_i w_{ip}}{\sum_i \mu_i}, (p = 1, 2, \dots, m) \quad (2)$$

Where μ_l is the truth value of the premise of the l^{th} rule, equations (1) and (2) describe the simplified fuzzy reasoning methods, through which the membership functions of the antecedent part are chosen. Then the fuzzy neural network (FNN) can be constructed and is shown in Fig. 1. [1]

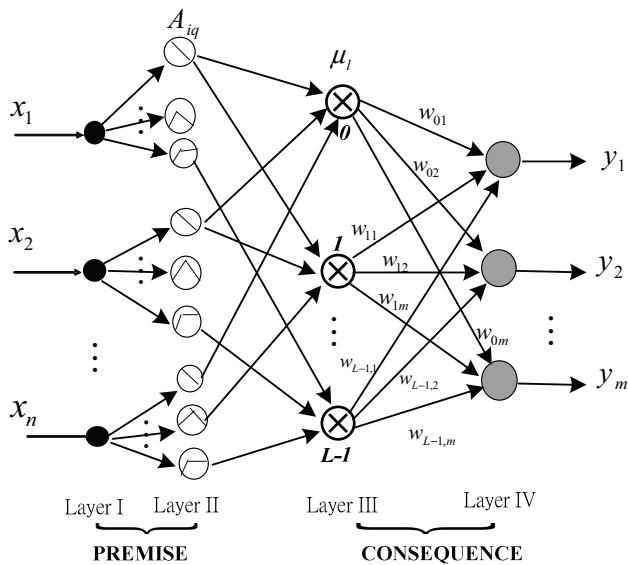


Figure. 1 The configuration of FNN

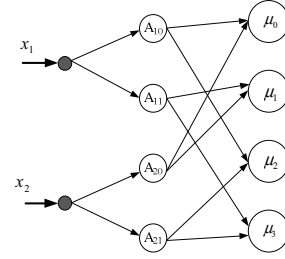


Figure. 2. Simple Illustration of Layers 1, 2 and 3 in FNN

TABLE I. SYSTEMATIC GENERATION OF RULES FROM MFs

r_1	r_2	$l(= r_1 + 2r_2)$	μ_l
0	0	0	$A_{10}(x_1)A_{20}(x_2)$
1	0	1	$A_{11}(x_1)A_{20}(x_2)$
0	1	2	$A_{10}(x_1)A_{21}(x_2)$
1	1	3	$A_{11}(x_1)A_{21}(x_2)$

Figure. 1 shows that the system has a total of four layers. Nodes at Layer I are input nodes, which represent input linguistic variables. Nodes at Layer II are term nodes, which represent the terms of the respective linguistic variables. Each node at Layer III is a fuzzy rule. All the layer-three nodes form the fuzzy rule base. Layer IV is the output layer. The links between Layer III and IV are fully connected by the weighting factors w_{lp} . Further a systematic generation of the fuzzy rule in Layer III has been derived in [16] which can reduce the programming effort with less execution time in real-time applications. In [16], a systematic generation of fuzzy rules in Layer III of the FNN in Figure 1 has been presented. In particular, the ordinal index l of the l^{th} fuzzy rule is:

$$l = r_1 + \sum_{i=2}^n \left(\prod_{j=1}^{i-1} R_j \right) r_i \quad (3)$$

where R_i is the total number of MFs for fuzzy variable x_i , and r_i ($0 \leq r_i \leq R_i - 1$) is the ordinal index for the r_i^{th} MF for fuzzy variable x_i . Equation (3) can produce all the l 's in the range of $0 \leq l \leq \left(\prod_{i=1}^n R_i \right) - 1$ with a one-to-one mapping from MFs to fuzzy rules. This will provide us with a complete indexing of the equivalent fully connected fuzzy neural network for FNN in the following sections. For two fuzzy variables $\{x_1, x_2\}$ with two MFs for $\{x_1, x_2\}$, i.e., $\{A_{10}(x_1), A_{11}(x_1)\}$ for x_1 , $\{A_{20}(x_2), A_{21}(x_2)\}$ for x_2 , with $\{R_1 = R_2 = 2\}$, as shown in Fig. 2. From (3), the ordinal index of four fuzzy rules can be generated using $\{l = r_1 + 2r_2, l = 0, 1, 2, 3\}$. Table I shows the result.

III. THE EQUIVALENT FULLY CONNECTED NEURAL NETWORK FOR FNN

As can be seen from the above discussion of FNN, the FNN is not a fully connected graph between Layer II and Layer III. This will hinder the discussion of the capacity of FNN. Therefore, in order to find the capacity of FNN, we need to convert the FNN into a fully connected neural network. There are two obstacles to be resolved. The first obstacle is that the links between Layer II and Layer III of FNN are not fully connected. The second obstacle is that computation unit in Layer III of FNN is a product unit rather than a summing unit. From Fig. 1, we can see the transitions from Layer II to Layer III can be simplified by arranging the links between Layer I to Layer III using equivalent weights. This can be more easily observed from Fig. 2 that there are links from each input node (x_1 or x_2) to every fuzzy rule, all through one of the MFs of every input node. Figures 3-1 & 3-2 show these cases for x_1 and x_2 with proper link weights. This is to say that we can suppress Layer II (MF layer) in the original FNN and replace new links between Layer I and Layer III, so that we can have a fully connected FNN (FFNN) as shown in Fig. 3-3.

This can be generalized to a general case of fully connected FNN, or FFNN as shown in Figure 4. This can resolve the first obstacle of not-fully connected property of the original FNN shown in Figure 1. The $\{V_{ij} (i = 1, 2, \dots, n; j = 1, 2, \dots, L)\}$ in Fig. 4 is defined as the link weight between the i^{th} node of Input Layer and the j^{th} node of Hidden Layer, or the fuzzy rule layer.

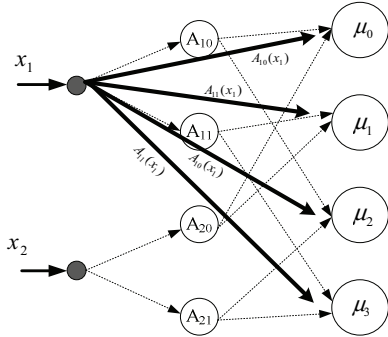


Figure 3-1. Suppressing the MF Layer for x_1

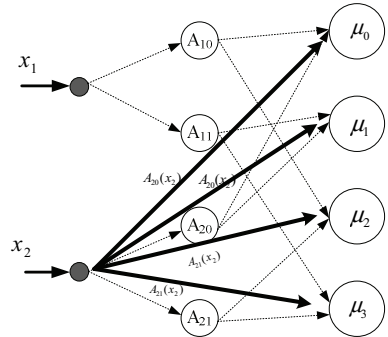


Figure 3-2. Suppressing the MF Layer for x_2

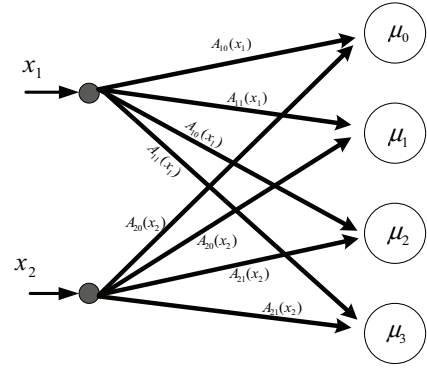


Figure 3-3. Suppressing the MF Layer with proper link weights

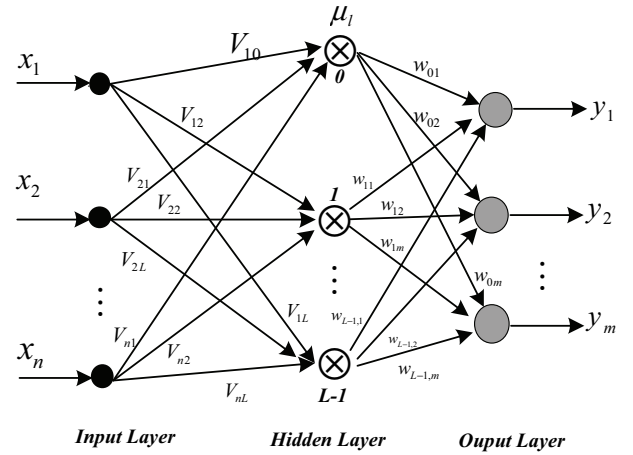


Figure 4. The equivalent fully connected neural network for FNN

However, the V_{ij} links between Input Layer and Hidden Layer in Figure 4 are still not well specified to yield μ_l in the Hidden Layer. We only know that each input node is only allowed to contribute one link weight form μ_l . Therefore μ_l is expressed in term of the product of n V_{ij} weights. Fortunately, a systematic way to generate the fuzzy rules in Hidden Layer has been proposed in [16]. This is to say that, given the fuzzy rule μ_l , we can determine its related links from all input nodes in a systematic way. From (3), we can reversely find all the r_i 's for a specific l (fuzzy rule number in Layer III) as follows:

$$\begin{aligned} r_1(l) &= l \% R_1; \\ r_i(l) &= (l / \prod_{k=1}^{i-1} R_k) \% R_i, i = 2, \dots, n \end{aligned} \quad (4)$$

where $A \% B$ represents the remainder of the division of A over B , A/B represents the quotient of the division of A over B . From (4) we can get n ordinal index $r_i(l) (i = 1, 2, \dots, n)$ of membership functions corresponding to rule l , then the $V_{ij} (i = 1, 2, \dots, n; j = 1, 2, \dots, L)$ can be set as:

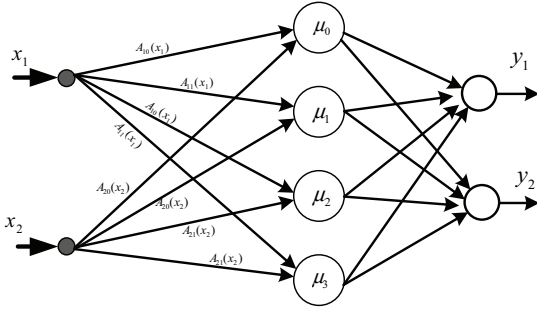


Figure 5. The FFNN with two input variables, two output variables

$$V_{ij} = A_{i_{r_j(j)}}(x_i), i = 1, 2, \dots, n; j = 1, 2, \dots, L \quad (5)$$

For two fuzzy variables $\{x_1, x_2\}$ with two MFs for each input variable i.e. $\{A_{10}(x_1), A_{11}(x_1)\}$ for x_1 , $\{A_{20}(x_1), A_{21}(x_1)\}$ for x_2 , then there are four hidden nodes (i.e., four rules), eight weights from input nodes to hidden nodes $V_{ij} (i = 1, 2; j = 1, 2, 3, 4)$. Table II shows the result. Fig. 5 shows the FFNN.

Therefore we have resolved the issue of not-fully connected MF layer in the original FNN, as shown in Figure 4. The next task will be to transform the product unit in Layer III (fuzzy rule layer) of FNN, or Layer II in Fig. 4 into an equivalent summing unit. To do so, we need to change the V_{ij} defined in (5) into the following new weight:

$$V_{ij} = \text{Ln}(A_{i_{r_j(j)}}(x_i)) \quad (6)$$

The output of hidden Layer is:

$$\mu_j = \sigma\left(\sum_{i=1}^n \text{Ln}(A_{i_{r_j(j)}}(x_i))\right) = \sigma\left(\text{Ln}\left(\prod_{i=1}^n A_{i_{r_j(j)}}(x_i)\right)\right) \quad (7)$$

where σ is any activation function, Ln is natural logarithm. If we take exponential function as activation function, we have

$$\mu_j = \text{Exp}\left(\text{Ln}\left(\prod_{i=1}^n A_{i_{r_j(j)}}(x_i)\right)\right) = \prod_{i=1}^n A_{i_{r_j(j)}}(x_i) \quad (8)$$

Fig. 6 illustrates this transformation via (6) to (8).

It is important to note that the exponential activation function in Fig. 6 has limit zero when the input argument is a negative number, or the exponential activation function has limit at one infinite direction. This does match the requirement for finding the capacity of the three layers NN [15]. Figure 7 shows the complete fully connected FNN, or FFNN. It is obvious that the FFNN shown in Figure 7 is equivalent to the original FNN shown in Fig. 1. However, the FFNN is a fully connected three layer directed graph, which is the basis of finding the capacity of FFNN, or FNN. This will be discussed in the following sections.

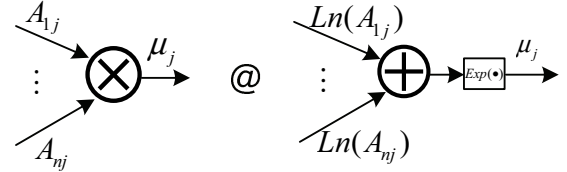


Figure 6. Transforming the product unit into summing unit in FNN

TABLE II FFNN WEIGHTS GENERATED FROM MFs AND RULE NUMBER

$l(=j-1)$	$r_1(l)$	$r_2(l)$	V_{1j}	V_{2j}
0	0	0	$A_{10}(x_1)$	$A_{20}(x_2)$
1	1	0	$A_{11}(x_1)$	$A_{20}(x_2)$
2	0	1	$A_{10}(x_1)$	$A_{21}(x_2)$
3	1	1	$A_{11}(x_1)$	$A_{21}(x_2)$

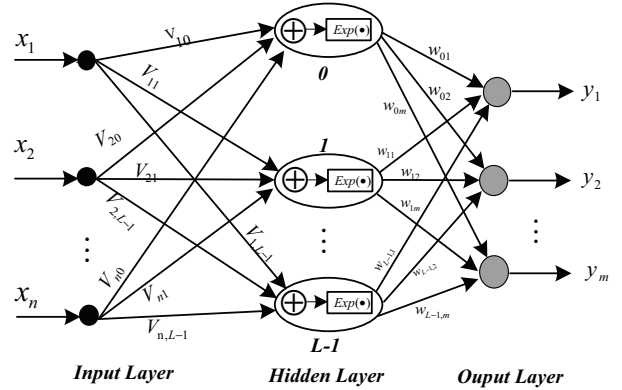


Figure 7. The complete fully connected FNN, or FFNN

IV. THE CAPACITY OF FNN OR FFNN

As mentioned in section 3, the original FNN has the same capacity as that of the equivalent FFNN with three layers. In this section, we will discuss the capacity of the three layers FFNN with exponential activation functions in the hidden layer. The following Theorem 1 shows lower and upper bounds of the FNN, or the three layer FFNN.

Theorem 1. For the FNN defined in Figure 1, where there are n inputs, m outputs, and L neurons in fuzzy layer (or hidden layer), each fuzzy variable x_i has R_i MFs, each MFs has N_{cp} control parameters, then the capacity P of the FNN can be bounded by the following inequality:

$$L \leq P \leq \left(\sum_{i=1}^n R_i N_{cp} + Lm\right) / m \quad (9)$$

Proof:

In [15], it has been shown that standard single-hidden layer feed-forward networks (SLFN) with at most L hidden neurons and with any bounded nonlinear activation function which has a limit at one infinity can learn L distinct samples with zero

error. This implies that the SLFN with L hidden units can memorize at least L distinct patterns. For the FFNN, the exponential activation function in the hidden layer satisfies the condition of activation function in [15], so the lower bound on the capacity P is the number of hidden nodes L , i.e.,

$$P \geq L \quad (10)$$

Further, it has been shown in [8] that the upper bound for the capacity of feed-forward networks with arbitrary hidden unit activation functions can be found. For a feed-forward network (without bias) with n inputs, m outputs, L hidden units and arbitrary hidden activation functions, the number of patterns P that can be memorized with no error is less than or equal to $L(n+m)$, divided by the number of outputs m . Then the upper bound for the capacity P of SLFN is

$$P \leq L(n+m) / m \quad (11)$$

In the right hand side of (11), nL represents the number of links between input layer and hidden layer and mL represents the number of links between hidden layer and output layer. The nL and mL links should have the weights which are independent of input variables. However the link weights between layer I and layer II of FFNN depends on input variables, we should find another kind of control parameters between layer I and layer II to find the upper bound of the capacity of the FFNN. Assume that each MF in FNN has identical N_{cp} control parameters, such as the mean and variance in Gaussian MFs. In this popular case, $N_{cp} = 2$. So the total number of independent control parameters between layer I and layer II is $\sum_{i=1}^n R_i N_{cp}$. Replacing nL in (11)

with $\sum_{i=1}^n R_i N_{cp}$, we have the following upper bound for the capacity P of FFNN:

$$P \leq (\sum_{i=1}^n R_i N_{cp} + Lm) / m \quad (12)$$

From (10) and (12), we can conclude that:

$$L \leq P \leq (\sum_{i=1}^n R_i N_{cp} + Lm) / m \quad \text{Q.E.D.}$$

For Gaussian or any triangular membership functions, the number of control parameters N_{cp} is 2. From (1) we

have $L = \prod_{i=1}^n R_i$, so if $n \geq 2$ and $R_i \geq 2$, then we have

$$nL = n \prod_{i=1}^n R_i > N_{cp} \sum_{i=1}^n R_i. \text{ Therefore, from (11) and (12),}$$

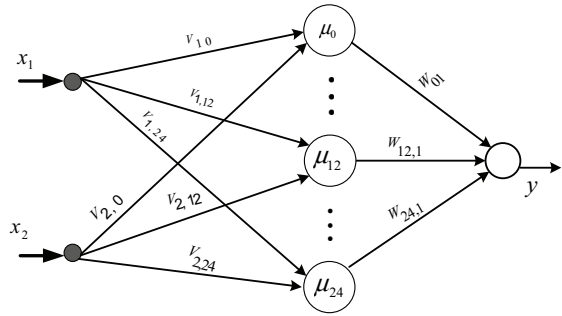


Figure. 8. FFNN with two input variables, each variable with five MFs

we find that the upper bound for capacity of FNN is less than that of MLP with the same configuration for larger n .

V. ILLUSTRATED EXAMPLE

In this section, Theorem 1 will be adopted to decide the theoretical upper and lower bounds of a particular FNN. In order to check the validity of the theoretical results, detailed BP trainings of the FNN are also conducted. The FFNN shown in Fig. 8 has two input nodes ($n = 2$) and one output node ($m = 1$), each linguistic variable has five items, so there are 25 nodes in hidden layer ($L = 25$). According to Theorem 1, the capacity P of the FNN should be bounded by

$L \leq P \leq (\sum_{i=1}^n R_i N_{cp} + Lm) / m$, which is $25 \leq P \leq 45$. The membership functions are Gaussian functions ($= \text{Exp}(\frac{-(x-\mu)^2}{2\sigma^2})$), the output node is a linear unit. The

training data is generated by $y = \sin(x_1\pi) \cos(x_2\pi)$, x_1 and x_2 are both in the range of $[-1, 1]$ with 0.1 as the step-size. The FNN can be trained by the following BP algorithm as described in [5]:

$$\left\{ \begin{array}{l} m_i(t+1) = m_i(t) - k_m \frac{\partial e_p}{\partial m_i}, i = 1, \dots, K \\ \sigma_i(t+1) = \sigma_i(t) - k_\sigma \frac{\partial e_p}{\partial \sigma_i}, i = 1, \dots, K \\ w_{jp}(t+1) = w_{jp}(t) - k_w \frac{\partial e_p}{\partial w_{jp}}, j = 1, \dots, L, p = 1, \dots, m \end{array} \right. \quad (13)$$

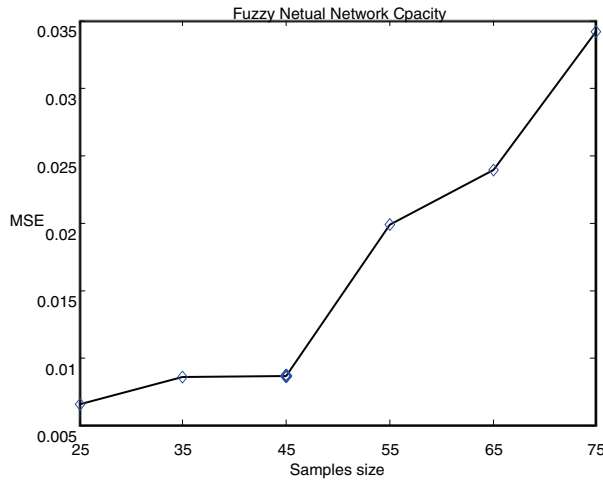


Figure 9. Relationship between training error and patterns number of FNN

Where $e_p = \frac{1}{2}(y_p - y_p^*)^2$, $K = \sum_{i=1}^n R_i$ is the number of MFs.

From Fig. 9, we can see that if the number of training patterns is near the lower bound (=25), the MSE can be made as close as possible to zero. When the number of training patterns reach upper bound (=45), the MSE values start to deviate from zero rapidly, which will be very difficult for the training process to converge. Therefore the theoretical bounds are in excellent agreement with the real BP training results

VI. CONCLUSION

This paper proposes a new approach to find the lower and upper bounds of the capacity of FNN. By transforming the conventional four layers FNN into an equivalent fully connected three layer feed-forward NN, or FFNN, we can obtain the lower and upper bounds of FNN. An illustrated example is proposed to demonstrate the validity of theoretical bound using real BP trainings. Excellent agreement has been obtained between the theoretical bounds and real BP trainings. The impact of this paper is that all engineering application via FNN should have the capacity of their imperfect FNN in mind to complete the successful implementations. This will be our future research directions.

VII. ACKNOWLEDGEMENT

This research is supported by the research fund of the National Grand Fundamental Research 973 Program of China under Grant 2011CB302801.

REFERENCES

- [1] Chi-Hsu Wang, Wei-Yen Wang, Tsu-Tian Lee, Pao -Shun Tseng, "Fuzzy B-spline Membership Function and Its Applications in Fuzzy-Neural Control," IEEE Trans. on Systems, Man, and Cybernetics , pp. 841-851, Vol. 25, No. 5, May, 1995.
- [2] Yih-Guang Leu, Tsu-Tian Lee and Wei-Yen Wang, "On-Line Tuning of Fuzzy-Neural Network for Adaptive Control of Nonlinear Dynamical Systems," IEEE Trans.on System Man and Cybernetics Part B, Vol. 27, No. 6, pp. 1034-1043, Dec. 1997.
- [3] Jeffrey T. Spooner and Kevin M. Passino, "Stable Adaptive Control Using Fuzzy Systems and Neural Networks," IEEE Trans. on Fuzzy System, Vol. 4, No. 3, pp. 339-359, Aug. 1996.

- [4] Wen Yu and Xiaou Li, "Fuzzy Identification Using Fuzzy Neural Networks With Stable Learning Algorithms," IEEE on Fuzzy System, Vol. 12, No. 3, pp. 339-359, Jun. 2004.
- [5] Kevin M. Passino and Stephen Yurkovich, Fuzzy Control, Addison Wesley Longman, Menlo Park, CA, 1998.
- [6] Liu, Puyin and Li, Hongxing, Fuzzy Neural Network Theory and Application, River Edge, NJ : World Scientific, 2004.
- [7] V. Kreinovich, H. T. Nguyen, and Y. Yam, "Fuzzy systems are universal approximators for a smooth function and its derivatives," International Journal of Intelligent Systems, Volume 15, Issue 6, pages 565-574, June 2000.
- [8] Pramod L. Narasimha, Michael T. Manry and Francisco Maldonado, "Upper Bound on Pattern Storage in Feedforward Networks," Proceedings of International Joint Conference on Neural Networks, Orlando, Florida, USA, August 12-17, 2007.
- [9] Robert J. McEliece, "The Capacity of the Hopfield Associative Memory," IEEE Trans. on Information Theory, Vol. IT-33, NO. 4, July, 1987.
- [10] Anthony Kuh and Bradley W. Dickson, "Information Capacity of Associative Memories" IEEE Trans. on Information Theory , Vol. 35, No. 1, Jan., 1989.
- [11] Thomas M. Cover, "Capacity Problems for Linear Machines," in Pattern Recognition, L. Kanal ed., Thompson Book Co., pp. 283-289. 1968.
- [12] Shin'ichi Tamura, "Capabilities of a three-layer feedforward neural network," in Proc. IJCNN91, pp. 2757-2762, Nov. 1991.
- [13] Shih-Chi Huang and Yih-Fang Huang, " Bounds on the Number of Hidden Neurons in Multilayer Perceptrons," IEEE Transaction on Neural Networks. Vol.2. No.1, pp. 47 -55, Jan. 1991.
- [14] M.A. Sartori and P.J. Antsaklis, "A simple method to derive bounds on the size and to train multilayer neural networks," IEEE Tran. on Neural Networks, vol. 2, No.4, pp. 467-471, July 1991.
- [15] Guang-Bin Huang and Haroon A. Babri, "Upper Bounds on the Number of Hidden Neurons in Feedforward Networks with Arbitrary Bounded Nonlinear Activation Function," IEEE Trans. on Neural Networks, vol. 9, no. 1, pp. 224-229, Jan. 1998.
- [16] Chi-Hsu Wang and Jung-Sheng Wen, "On the Equivalence of a Table Lookup (TL) Technique and Fuzzy Neural Network (FNN) With Block Pulse Membership Functions (BPMFs) and Its Application to Water Injection Control of an Automobile," IEEE Trans on Systems, Man, and Cybernetics- Part C: Applications and Reviews, Vol. 38, NO. 4, pp. 574-580, July 2008.