4th WSEAS Int. Conf. on COMPUTATIONAL INTELLIGENCE, MAN-MACHINE SYSTEMS and CYBERNETICS Miami, Florida, USA, November 17-19, 2005 (pp204-209)

A fault tolerant Neuro-Fuzzy Inference system: using Coarse-coded Distributed Representations

SRIRAM .G. SANJEEVI, DR. PUSHPAK BHATTACHARYYA

Sriram. G. Sanjeevi, Asst.Professor,
Dept. of Comp. Science & Engg.,
National Institute of Technology
Warangal 506004, INDIA

Dr. Pushpak Bhattacharyya, Professor,
Dept. of Comp. Science & Engg.,
Indian Institute of Technology, Bombay,
Mumbai 400076, INDIA

Abstract: - In this paper, we describe a fault-tolerant Neuro-Fuzzy inference system for performing fuzzy reasoning using *coarse-coded* distributed representations. The system implements the fuzzy membership functions in a novel way using coarse-coded distributed representations for the inputs and outputs of neural networks. Distributed representations are known to give advantages of fault tolerance, generalization and graceful degradation of performance under noise conditions. Performance of the Neuro-Fuzzy inference system with regard to its ability to exhibit fault tolerance under noise conditions is studied. The system offered very good results of fault tolerance under noise conditions. It has also exhibited good generalization ability on unseen test inputs.

Key-Words: - Coarse-coding, Fuzzy rules, Neural, Reasoning, Fault tolerance, Membership function.

1 Introduction

Traditionally reasoning systems using predicate logic have been implemented using symbolic methods of artificial intelligence. Connectionist or neural network methods of implementation of reasoning systems describe an alternative paradigm. Among the connectionist reasoning systems they use two types of representational schemes. They are 1) localist and 2) distributed representational schemes.

Localist representational schemes represent each concept with an individual unit or neuron. In the distributed representational schemes [5] each unit or neuron is used in representation of multiple concepts and multiple units or neurons are used to represent a single concept. In the literature, some localist methods for predicate logic reasoning using connectionist networks have been described. The connectionist inference system *SHRUTI* [1], [2], [3] described a localist method where temporal synchrony was used to create bindings between variables and entities they represent. A variable *x* of the predicate *give(x, y, z)* is getting bound to an entity *d* if the nodes representing them fire during the same phase of time *p1* during the predicate *p* activation period *T*. The time period *T* is divided into three phases *p1, p2* and *p3* during which synchronous firing of variables *x, y* and *z* and entity nodes they bound respectively takes place. This method has used temporal synchrony as a mechanism to create localist representations. *CONSYDERR* [4,9] described a localist method for variable binding and forward reasoning. It used an assembly or a set of interconnected nodes to represent each predicate $p(x_1 \ldots . x_k)$. Each assembly contains one C node for storing the confidence value of the predicate *p* and *k X* nodes to store the binding values for *k* variables of the predicate *p*. A separate node is allocated for each variable of a predicate. Since, these systems used localist representations, advantages of distributed representations were not obtained

by them. In our previous works [6,7,8] we proposed and described predicate logic reasoning systems using neural networks which used coarse-coded distributed representations to represent instantiated predicates. In these publications we have described the advantages of using the coarse-coded representations along with neural networks for performing predicate logic reasoning. However, the predicate logic reasoning is nonfuzzy and crisp reasoning and hence not suitable for fuzzy reasoning. Here we propose to examine the use of the coarse-coded distributed representations along with the neural networks in the implementation of a Fuzzy-rule-based system. Our motivation is to make available the advantages of distributed coarse-coded representations to the neuro-fuzzy reasoning systems. It is investigated here in this work to examine the advantages gained by the *Neuro-Fuzzy reasoning system* by using distributed *coarse-coded representations.*

2 Fuzzy Rule Base

We propose to implement, as an example, the following TSK (Takagi-Sugeno) fuzzy rules in our system.

1. If x is small and y is small, then
$z = f_1 = -x + y/4 + 1$
2. If x is small and y is large, then
$z = f_2 = 0.75y + 3$
3. If x is large and y is small, then
$z = f_3 = x + 2$
4. If x is large and y is large, then
$z = f_4 = x + y + 1$
5. If x is medium and y is large, then
$z = f_5 = x/2 + y + 2$
6. If x is small and y is medium, then
$z = f_6 = -x + y/2 + 1$

For given crisp input values for x and y the inferred output *z* is calculated by

$$Z=(\mu_1 f_1+\mu_2 f_2+\mu_3 f_3+\mu_4 f_4+\mu_5 f_5+\mu_6 f_6)/(\mu_1+\mu_2+\mu_3+\mu_4+\mu_5+\mu_6) \quad \text{.....(1)}$$

where

$\mu_1 = \mu_{small}(x) * \mu_{small}(y);$
$\mu_2 = \mu_{small}(x) * \mu_{large}(y);$
$\mu_3 = \mu_{large}(x) * \mu_{small}(y);$
$\mu_4 = \mu_{large}(x) * \mu_{large}(y);$
$\mu_5 = \mu_{medium}(x) * \mu_{large}(y);$
$\mu_6 = \mu_{small}(x) * \mu_{medium}(y)$

where $\mu_{small}(x)$ denotes the membership function for linguistic value *small* of the linguistic variable x, $\mu_{medium}(x)$ denotes the membership function for linguistic value *medium* of the linguistic variable x, $\mu_{large}(x)$ denotes the membership function for linguistic value *large* of the linguistic variable x, $\mu_{small}(y)$ denotes the membership function for linguistic value *small* of the linguistic variable y, $\mu_{medium}(y)$ denotes the membership function for linguistic value *medium* of the linguistic variable y, $\mu_{large}(y)$ denotes the membership function for linguistic value *large* of the linguistic variable y, and '*' denotes the 'Minimum' operation between the indicated fuzzy membership functions. Our task is to start with the above fuzzy rule base and obtain the results of fuzzy inference correctly by our system.

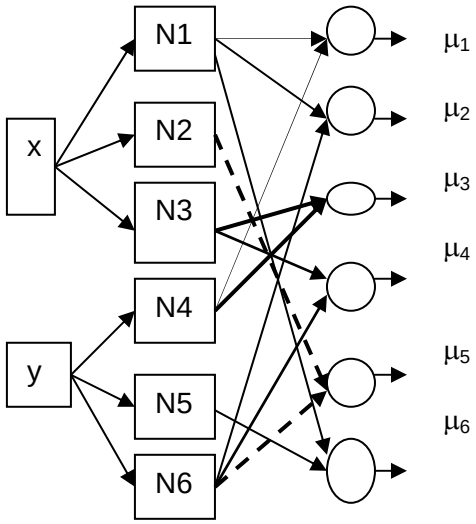

Fig.1. Neural Networks N1 to N6 implement fuzzy membership functions

The architecture for implementing the fuzzy membership functions with neural networks N1 to N6 which use coarse-coded representations at their inputs and outputs is shown in the figure 1 above. Neural networks N1 to N6 implement $\mu_{small}(x)$, $\mu_{medium}(x)$, $\mu_{large}(x)$, $\mu_{small}(y)$, $\mu_{medium}(y)$ and $\mu_{large}(y)$ respectively. *x* and *y* values are coarse-coded before giving them as inputs to the neural networks N1 to N6. Neural networks N1 to N6 generate on their outputs respective membership function values in coarse-coded form. These membership function values after decoding will be in the interval [0, 1]. Each of the six circles at the right side of figure 1 are used to denote the 'minimum' operation performed on its inputs on the decoded

membership function values. They generate on their outputs μ_1 to μ_6 respectively. From these the value of output *z* is computed using formula in equation (1) for a given *x, y.*

3 Description of data used by fuzzy membership functions

We show here in the following tables 1 to 6 the samples of the data needed by the fuzzy membership functions $\mu_{small}(x)$, $\mu_{medium}(x)$, $\mu_{large}(x)$, $\mu_{small}(y)$, $\mu_{medium}(y)$ and $\mu_{large}(y)$ respectively. We assume the data shown in tables 1 to 6 is obtained from the expert knowledge for the efficient performance of our fuzzy system.

Table 1. Shows a sample of data used by membership function $\mu_{small}(x)$

S.No.	X	$\mu_{small}(x)$
2	6.0	0.98
10	30.0	0.08

Table 2. Shows a sample of data used by membership function $\mu_{medium}(x)$

S.No.	x	$\mu_{medium}(x)$
8	24.0	0.98
14	42.0	0.1

Table 3. Shows a sample of data used by membership function $\mu_{large}(x)$

S.No.	x	$\mu_{large}(x)$
12	36.0	0.3
16	48.0	0.96

Table 4. Shows a sample of data used by membership function $\mu_{small}(y)$

S.No.	y	$\mu_{small}(y)$
3	6.0	0.95
5	10.0	0.6

Table 5. Shows a sample of data used by membership function $\mu_{medium}(y)$

S.No.	y	$\mu_{medium}(y)$
6	12.0	0.3
12	24.0	0.4

Table 6. Shows a sample of data used by membership function $\mu_{large}(y)$

S.No.	y	$\mu_{large}(y)$
11	22.0	0.45
14	28.0	0.95

3.1 Obtaining Coarse-coded Distributed Representations

We describe here how to obtain coarse-coded representations of the data shown in the tables *1* to *6.* If the data items to be coarse-coded lie in the interval *[R1, R2*] where *R1* and *R2* are real numbers then we divide the interval *[a, b]* into *N* sub-intervals, where *a = floor(R1)* and *b = ceiling(R2).* Suppose we need to encode the real number *R* present in the *k*th sub-interval we proceed as follows. If *R* is in lower half of the sub-interval *k* we represent *R* with a localist pattern having *(N+1)* bits, with $(k)^{th}$ bit *1* and other bits '*zero'*. If *R* is in upper half of the sub-interval *k* we represent *R* with a localist pattern having *N+1* bits, with $(k+1)^{th}$ bit *1* and other bits *zero*. We illustrate the coarse-coding process through an example. For coarse-coding output values of membership functions we chose *N = 40*. Consider the *2nd* row of table *1*. The value of $\mu_{small}(x)$ is *0.08.* Since this value belongs to lower half of *4th* sub-interval we encode it with a localist pattern of *41* bits with *4th* bit *1* and all other bits '*zero'.*

'00000000000 0000000000 0000000000 0000001000' ……..(2)

We are yet to convert this localist pattern into the coarse-coded pattern. We add three leading zeroes to left side of the above pattern for the reason described below to get the following pattern

'00000000000000 0000000000 0000000000 0000001000'.

We illustrate the process of converting the above localist pattern into the coarse-coded pattern. We view the above pattern as being kept in overlapping coarse zones of length of *4* consecutive bits and encode the zone as *1* if there is at least one *1* bit in that zone or else as *0*. We then consider next coarse zone and encode it as *1* or *0* following above method. We do this process left to right starting from the left most bit. We do this encoding process for above localist pattern to get the following coarse-coded pattern

'00000000000000 0000000000 0000000000 0001111000'.

The reason we have added three leading zeros to the localist pattern in (2) is to enable us to coarse-code the pattern even if the 1 bit occurred in the 41st bit (from the right side) of the pattern.

Coarse-coding can be applied when the number of *1*'s in the original string is sufficiently sparse. If the number of *1*'s in the original string is not sufficiently sparse then coarse-coded string when decoded will not yield the original string. Coarse-coding increases the information capacity [5] by increasing the number of units active at a time compared to localist codes which have sparsely populated *1*'s. The amount of information conveyed by a unit that has a probability *p* of being '*1*' is

$$-p\log(p) - (1-p)\log(1-p).$$

The coarse-coding process involves making an approximation and hence an error. To decrease the error to the desired limits, we choose an appropriate value of '*N*'. '*x*' and '*y*' values used by our fuzzy system are real numbers in the range *0* to *60.0*, *0* to *40.0* respectively. The *N* values for coarse-coding them were chosen to be *60* and *40* respectively. Using the technique described above we obtain the coarse-coded representations of patterns for all the fuzzy membership function inputs and outputs in our system. We show here a sample of coarse-coded representations of the patterns for the various membership functions in tables 7 to 12.

Table 7. Shows a sample of coarse-coded data used by membership function $\mu_{small}(x)$

S.No.	X	$\mu_{small}(x)$
6	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Table 8. Shows a sample of coarse-coded data used by membership function $\mu_{medium}(x)$

S.No.	X	$\mu_{medium}(x)$
8	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Table 9. Shows a sample of coarse-coded data used by membership function $\mu_{large}(x)$

S.No.	X	$\mu_{large}(x)$
9	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0

Table 10. Shows a sample of coarse-coded data used by membership function $\mu_{small}(y)$

S.No.	Y	$\mu_{small}(y)$
3	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0	0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Table 11. Shows a sample of coarse-coded data used by membership function $\mu_{medium}(y)$

S.No.	y	$\mu_{medium}(y)$
6	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Table 12. Shows a sample of coarse-coded data used by membership function $\mu_{large}(y)$

S.No.	y	$\mu_{large}(y)$
11	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

4 Testing

Following are the details of neural networks used in our work shown in table 13. The neural networks N1 to N6 in figure 1 are feed forward neural networks using back-propagation algorithm.

Table 13. Shows the details of neural networks used

Neural Network	No. of input units	No.of hidden units	No.of output units
N1	63	52	44
N2	63	52	44
N3	63	52	44
N4	43	32	44
N5	43	32	44
N6	43	32	44

The fuzzy reasoning task was successfully accomplished by using the system to give the expected results. The value of system output z obtained for two values of (x, y) are shown in table 14.

Table 14. Shows the details of inputs (x,y) and corresponding z values

Sr.no	1	2
x	16	33
y	11	20
μ1	0.6	0
μ2	0	0.05
μ3	0	0
μ4	0	0.05
μ5	0	0.2
μ6	0.05	0.05
z	-12.04	29.14

Table 15. Shows the details of tests for fault-tolerance

Neural Network	No. of Training Patterns	No. of Test Patterns	No. of Patterns Corrected	No. of Patterns not Corrected
N1	20	20	20	0
N2	20	20	19	1
N3	20	20	19	1
N4	20	20	20	0
N5	20	20	20	0
N6	20	20	20	0

Secondly, the performance of the above coarse-coded Neuro-Fuzzy reasoning system was tested for fault tolerance under noise conditions for each of the neural networks N1 to N6. In the tests, neural networks N1 to N6 are trained with 20 patterns each. These training patterns were made test patterns after introducing *1* bit error at a random location in each of these patterns. These artificially introduced errors are simulating noise conditions. Results are as shown in table 15. The coarse-coded Fuzzy reasoning system was found to be highly fault tolerant to errors as was indicated by the test results.

In the second set of tests conducted, the performance of Coarse-coded fuzzy reasoning system was checked for its generalization ability on unseen test patterns. The system has displayed good generalization ability on unseen test patterns. Results are shown in table 16.

Table 16. Shows the details of tests for generalization

Neural Network	No. of Training Patterns	No. of unseen Test Patterns	No. of Patterns Correctly generalized	No. of Patterns not Correctly generalized
N1	20	20	14	6
N2	20	20	16	4
N3	20	20	17	3
N4	20	20	16	4
N5	20	20	14	6
N6	20	20	13	7

5 Conclusions

We have designed and tested a Neuro-Fuzzy reasoning system which uses distributed coarse-coded representations with the neural networks.The system implements in a novel way the fuzzy membership functions for its linguistic terms using neural networks with coarse-coded distributed representations at their inputs and outputs. The system has successfully performed the fuzzy reasoning. The coarse-coded fuzzy reasoning system exhibited good generalization ability on unseen test patterns thereby displaying learning ability. The coarse-coded fuzzy reasoning system exhibited very good fault tolerance under simulated noise conditions as indicated by the tests.

References:

[1] L. Shastri, Advances in SHRUTI: a neurally motivated model of relational knowledge representation and rapid inferencing using temporal synchrony, *Applied Intelligence,* 11(1), 1999, pp. 79-108.

[2] C. Wendelken and L. Shastri, Multiple instantiation and rule mediation in SHRUTI, *Connection Science*, 16, 2004, pp. 211-217.

[3] L. Shastri, C. Wendelken, Seeking coherent explanations --- a fusion of structured connectionism, temporal synchrony and evidential reasoning. *Proceedings of Cognitive Science 2000,* Philadelphia, PA, August 2000

[4] R. Sun, On variable binding in connectionist networks. *Connection Science*, 4, 1992, pp. 93-124.

[5] G.E. Hinton, J.L. McClelland and D.E. Rumelhart, Distributed representations. In D.E.Rumelhart and J.L.McClelland, editors, *Parallel Distributed Processing,* Vol.1. Cambridge, MA. MIT Press, 1986

[6] Sriram.G.Sanjeevi, P. Bhattacharya. Connectionist Reasoning System using Coarse-coded Distributed Reprentations, *International conference on Systemics, Cybernetics and Informatics,(ICSCI 2005),* vol 1, pp723-728, Hyderabad, January 06-09, 2005.

[7] S.G.Sanjeevi and P. Bhattacharya., A fault tolerant Connectionist Model for Predicate Logic Reasoning, variable binding: using Coarse-coded Distributed Representations, *WSEAS Transactions on Systems,* Issue 4, Volume 4, April 2005, pp. 331-336.

[8] Sriram.G.Sanjeevi, P. Bhattacharya. A Connectionist Model for Predicate Logic Reasoning using Coarse-coded Distributed Representations, 9th International conference on Knowledge-based & Intelligent Information and Engineering Systems(KES 2005) *, LNCS Proceedings of KES 2005, Springer-Verlag,* Melbourne, Australia, September 14th-16th, 2005.

[9] A.Browne, R. Sun, Connectionist inference models, *Neural Networks* 14, 2001, pp.1331-1355.

[10] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE transactions on systems, Man and Cybernetics*, 15:116-132, 1985.

[11] S. Haykins, *Neural Networks, a comprehensive foundation,* Second edition, New Jersey: Prentice hall, 1999