International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:12, 2007

RANFIS: Rough Adaptive Neuro-Fuzzy Inference System

Sandeep Chandana, and Rene V. Mayorga

***Abstract*—**The paper presents a new hybridization methodology involving Neural, Fuzzy and Rough Computing. A Rough Sets based approximation technique has been proposed based on a certain Neuro – Fuzzy architecture. A New Rough Neuron composition consisting of a combination of a Lower Bound neuron and a Boundary neuron has also been described. The conventional convergence of error in back propagation has been given away for a new framework based on 'Output Excitation Factor' and an inverse input transfer function. The paper also presents a brief comparison of performances, of the existing Rough Neural Networks and ANFIS architecture against the proposed methodology. It can be observed that the rough approximation based neuro-fuzzy architecture is superior to its counterparts.

***Keywords*—**Boundary Neuron, Neuro – Fuzzy, Output Excitation Factor, RANFIS, Rough Approximation, Rough Neural Computing.

I. Introduction

SOFT Computing, may be regarded as a science aimed at emulating human ability to deal with uncertainty and imprecision in real time. Taking it away from the conventional AI techniques, soft computing has evolved not as one technique but as a synergistic collection of more than one technique, Evolutionary, Neural and Fuzzy Computing being the prime methodologies. This paper deals with a similar hybridization involving Neural and Fuzzy Computing. Another dimension has been added to the combination, by the introduction of Rough Sets based classification principles.

As Pawlak [1] stated, '*knowledge is deep seated in the classificatory abilities of living organisms*'. Abstract level classification forms the basis for reasoning, learning and decision making. Thus knowledge has a definitive relationship with classification patterns which may be used in a way to derive secondary knowledge from primary knowledge / information. Classification in principle aims to segregate the given objects into classes in an effort to allocate a known solution to the elements belonging to the respective classes.

Manuscript received on December 30, 2005.

Sandeep Chandana is a graduate student and affiliated to the Wise & Intelligent Systems & Entities Laboratory in the Faculty of Engineering (Industrial Systems Engineering) at the University of Regina, SK – S4S 0A2; Canada (phone: 306 585-6664; e-mail: chandans@uregina.ca).

Rene V. Mayorga heads the Wise & Intelligent Systems & Entities Laboratory in the Faculty of Engineering (Industrial Systems Engineering) at the University of Regina, SK – S4S 0A2; Canada (corresponding author to provide phone: 306 585-4726; fax: 306 585-4855; e-mail: Rene.Mayorga@uregina.ca).

Such a methodology results, in principle, in pruning of the search space (based on certain parameters).

[2] - [5] have successfully adopted rough regions into conceiving, better and efficient performance. The aim of the proposed architecture is to adopt such a pruning mechanism based on rough classification for continuous learning during and after training. The paper also outlines a novel method for fast error convergence in back propagation.

A Rough Neuron, introduced for the first time by Lingras [5], comprised of one upper and another lower neuron, each reflecting the respective rough approximation regions. Such pairs were connected through excitatory and inhibitory connections depending upon the behavior of the outputs. The connections would then be dynamically configured during the learning processes i.e. the system would configure itself to an excitatory connection if an increase in the input signal would in some proportion increase the output. If vice versa, the system would revert to an inhibitory connection. Such signal propagation, though superior to the conventional neural networks, doesn't take into account individual effects of the lower and upper approximation neurons. Further, it should be noted that, a similar rough pattern may be generated in the error back propagation as well.

This article, in principle details a new composition for the new rough neuron proposed by Chandana-Mayorga [8], comprising of a boundary neuron and a lower bound neuron. A new weighted (inter neuron) link system, based on what was called; the Output Excitation Factor (OEF) has been proposed. A new error convergence methodology based on a certain input inverse functions and the OEF has been detailed.

Section 2 deals with a preliminary introduction to the involved concepts. The Rough approximation based Neuro – Fuzzy Inference architecture has been described in section 3. Section 4 describes a novel method of accommodating rough patterns in the error back propagation along with the use of an input inverse function. Section 5 presents some preliminary results. Virtues of the proposed methodology have been compared with a few existing techniques in this section.

II. Preliminaries

Considering the fact that Rough Sets Theory and various models of Neuro – Fuzzy systems have been in existence for a few years, only a brief introduction to the said concepts has been provided here. Basic introduction to Rough Sets Theory and some preliminary information about one of the Neuro-Fuzzy Architectures are has been presented in the subsections.

A. Rough Sets Theory (RST)

RST was first developed by Zdzislaw Pawlak in Poland in the early 1980s'. It deals predominantly with the classificatory analysis of imprecise, uncertain or incomplete information. Rough Set Theory can be defined as follows [1], [2],

Let Universe (*U*) be a finite, non-empty set with, *I,* being an equivalence relation called the *indiscernibility relation* on *U*. *I(x)* would then be described as an equivalence class of the relation *I* containing the element *x*. The concept of an *indiscernibility relation* brings about the fact that not all elements in the Universe can be discerned given the information available. Further, such an *indiscernibility relation* is used to determine the *lower, upper* and *boundary approximations* (which may be expressed as),

$$I_*(X) = \{x \in U : I(x) \subseteq X\},$$

$$I^*(X) = \{x \in U : I(x) \cap X \neq 0\},$$

$$BN_I(X) = I^*(X) - I_*(X).$$

The rough membership function is defined as follows,

$$\mu_X^I(x) = \frac{card(X \cap I(x))}{card(I(x))},$$

where $\mu_X^I(x) \in [0,1]$

Based on such an expression of the membership function, the concepts of *lower, upper* and *boundary* regions can be defined as [1],

$$I_*(X) = \{x \in U : \mu_X^I(x) = 1\},$$

$$I^*(X) = \{x \in U : \mu_X^I(x) > 0\},$$

$$BN_I(X) = \{x \in U : 0 < \mu_X^I(x) < 1\},$$

Given the above functions, RST can be categorized as a methodology complimentary to the Fuzzy Set Theory.

B. ANFIS

The ANFIS architecture proposed by Jang [6] can be described as, one made up of adaptive networks which are functionally similar to Fuzzy Inference Systems. ANFIS has an edge over other prevalent hybrid architectures due to its mathematical framework devised to decompose the parameter set (of the adaptive network nodes). Such decomposition further helps in implementing a Hybrid Learning algorithm composed of the Gradient Descent Method and the Least Squares Estimator. ANFIS successfully attempts to represent the Sugeno Fuzzy models with an advantage that the learning can be interpreted from perspectives of both neural and fuzzy systems. And more importantly such a system enables viewing the problem solution in a linguistic fashion.

Architecture: The said architecture [6] is comprised of a five layer multi neuron neural network. The objective functions of all nodes in Layer 1 are designed to assign a relevant fuzzy membership function to the applied inputs. In Layer 2, the outputs of the individual preceding layers are multiplied to retrieve the first set of parameters called the premise parameters i.e. the weights are generated. Layer 3 neurons calculate the normalized weights (also called the normalized firing strengths) for the nodes. Layer 4, which is adaptive in nature, computes output based on the normalized weights and the output firing function. Computation in this layer generates a second set of parameters called the consequent parameters. Layer 5 computes the overall output of the five layer network by summing all of the incoming signals into an output. The training or the learning process is aided by the use of a hybrid learning algorithm.

III. Rough Approximation based Neuro – Fuzzy Inference System (RANFIS)

In this section, we describe a Rough Neuron, which is a working pair of neurons, consisting of one lower bound neuron and one *boundary neuron*. As the name suggests the lower neuron is representative of the lower bound approximation and the *boundary neuron* is representative of the disjunct between the upper and lower bound approximations. The main intent in introducing the concept of a *boundary neuron* is to accommodate the random and unpredictable effect of the boundary signal on the rough neuron output. A general analysis of the output of the lower bound neuron will yield a conclusion that, the behavior of the lower bound neuron output is skewed. This statement can be made in the sense that, the output of lower neuron relies on a rather less complex and predictable function. Further, it should be noted that the behavior of the lower bound neuron is thoroughly devoid of any uncertainty unlike its boundary based counterpart.

Such a composition of the rough neuron effectively results in drawing a line between the certain and the uncertain behaviors i.e. an implicit region for certainty and uncertainty can be identified within the solution space. Such a demarcation should (and has,) result in an improved overall approximation.

A. The New Rough Neuron

The New Rough Neuron is made up of a combination of two individual neurons working in a pseudo deterministic fashion (explained in the succeeding sections). One of the constituent neurons is called the Lower Bound Neuron, and is responsible for the output approximation (of the overall Rough Neuron) based on the lower bound of the applied input signal i.e. the lower bound neuron deals only with the definite or certain part of the signal and produces its share of the total output. The second and the more prominent half of the Rough Neuron, what the authors call the *Boundary Neuron*, is designed in such a way that it processes only the boundary signal i.e. the boundary neuron deals only with the random part of the signal. The randomness in the applied signal may be categorized either as the noise present in the data or data content that the neurons have not yet been introduced to. This interpretation of randomness is applicable only to the learning / training stage of the neural network. The aforementioned details about the purpose and structure of the Rough Neuron constituents may be understood better with the help of the following illustration;

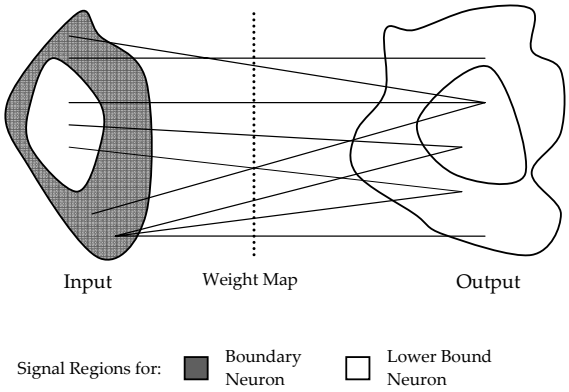

Fig. 1 Rough Signal Mapping for the constituent Neurons

The proposed architecture has been designed in such a way that the overall boundary signal generated as output; reduces as the signal moves forward from one layer to the next. This mechanism can be interpreted in simpler terms, by saying that the randomness in the output signal consistently decreases in the forward runs.

Network Design: A basic Rough Neural Network Design has been described in this subsection. Let's first consider the linkages between 4 rough neurons in a network; (Fig. 2). As mentioned earlier, the boundary and lower bound parts of the M_i neuron, partition the applied input signal into random and predictable sections.

Then they respectively process their parts of the input signal and produce outputs which are collated (as per details in the next subsection). The thus produced output signals from the neurons of layer M (which are now input signals into layer N,) are once again partitioned into the two classes. This process repeats until the penultimate layer (or last hidden layer) further to which, the outputs are simply passed through a summation operator. The deviation of the obtained output compared to the desired output (after each complete iteration) dictates all further training and modification of the linkages.

With reference to Fig. 2; please take note that (B) depicts the implicit connections between the four constituent neurons of the two connected Rough Neurons. Despite of not being physically connected to an array of other neurons, the presented model establishes a qualitative linkage between every pair of non-homogenous neurons; a pair of constituent (Boundary and Lower Bound) neurons, belonging to the same Rough Neuron, forms a homogenous set.

Node Functions & Signal Propagation: The Lower Bound Neuron and the Boundary Neuron have both been designed with a sigmoid transfer function. A generalized sigmoid function (Equation 1.), has been chosen in order to accommodate any non-linearity encountered during the modeling process.

$$F(x) = \alpha / \left(1 + e^{\beta \cdot x + \varsigma}\right) \tag{1}$$

The network structure is based upon the nodal transfer functions and the behavior of the nodal outputs. The Nodal

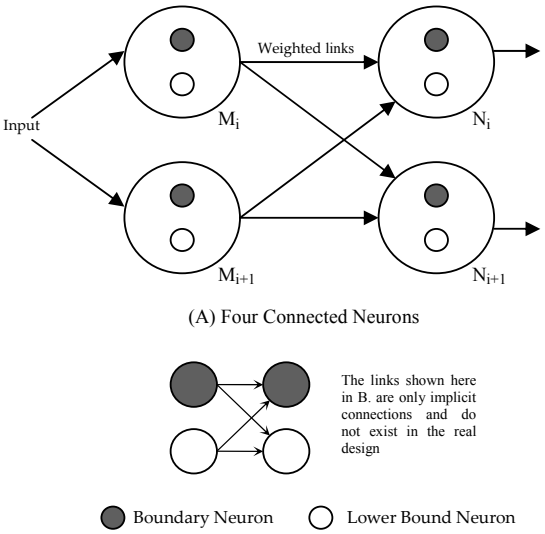

Fig. 2 Design of Linkages among the Rough Neurons

outputs in turn are dependant upon the *Output Excitation Factor* (OEF; ε).

Output Excitation Factor (ε) may be defined as a ratio of change in the (resultant) output magnitude to the change in (applied) input magnitude. Determination of such a behavior trait is critical in implementing randomness reduction in the signal. Following are the relevant output functions (in order for an applied signal ***x***); which would further explain the methodology;

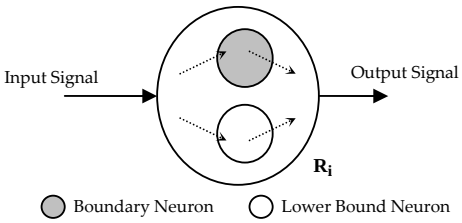

Fig. 3 One New Rough Neuron

$$\text{Output (of } \mathbf{R_i}) = \Sigma \text{ output of constituent neurons} \tag{2}$$

Output of Boundary Neuron is given as;

$$o_{\overline{R}-\underline{R}} = \frac{transfer\ (x_{\underline{R}})}{\varepsilon_{\underline{R}}} + \frac{transfer\ (x_{\overline{R}-\underline{R}})}{\varepsilon_{\overline{R}-\underline{R}}} \tag{3}$$

And the output of the Lower Bound Neuron is given as;

$$o_{\underline{R}} = \min\left(transfer\ (x_{\underline{R}}), transfer\left(x_{\overline{R}-\underline{R}}\right)\right) \tag{4}$$

The Output Excitation Factors may be given as;

$$\varepsilon_{\overline{R}-\underline{R}} = -\left[\frac{\Delta(o_{i,j})}{\Delta(x_{i,j})}\right] \ \&\ \varepsilon_{\underline{R}} = \left[\frac{\Delta(o_{i,j})}{\Delta(x_{i,j})}\right] \tag{5}$$

One should pay attention to the fact that the excitation factor, in reality has a negating effect on the output of the boundary neuron (refer to Eq.3 & 5). And with reference to Equation 2, such an effect is in turn propagated to the overall rough neuron output. Since the Boundary Neuron processes the random part of the applied input signal, over a few layers, we see a considerable reduction in the randomness. This is one half way of reducing uncertainty; the second half will be taken up for discussion in the (next) subsection, while describing the learning algorithm.

As mentioned earlier (refer to Fig.1); the lower bound neuron also tends to produce an output with some randomness in it. But a negating operator is not applied to its output excitation factor, since the generated random signal would be dealt with a boundary neuron in the succeeding layer. Such a mechanism applied across the rough neuron $\boldsymbol{R_i}$ (refer to Fig.3) would boost the lower bound signal and simultaneously inhibit the boundary signal.

The Transfer Function may be given as;

$$F(x) = \alpha / \left(1 + e^{\beta \cdot x + \varsigma}\right) \quad (1)$$

where the parameters α, β, ζ should be optimally fixed in order to generalize the neural network for performance.Thus, (with reference to Fig. 3) the output of the full Rough Neuron may be given as;

$$o_{R_i} = o_{\overline{R-R}} + o_{\underline{R}} \quad (2.a)$$

Further, (with reference to Fig.2A) input to a succeeding layer (***N***) can be given as a linear combination of all outputs of neurons belonging to the preceding layer (***M***). The linear combination is based on the weighted connections between the respective neurons. Finally, the output layer of the rough neural network should consist of either a singular rough neuron or a conventional neuron with a summation equivalent transfer function.

B. Network Architecture

The proposed architecture (in essence) consists of a five layer neural network, but such is the network's design that it can be made up of *2n+3* layers, where *n* = (*1, 2* …..).

A simple network design has been depicted in Figs. 4.

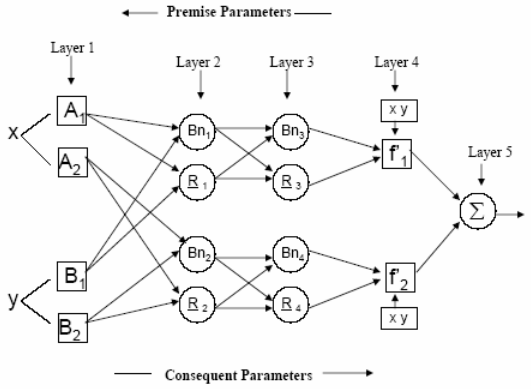

Fig. 4 Rough Approximation based Neuro Fuzzy Inference System network architecture

Layer 1: The node functions assign a certain membership function to the applied input i.e. (*following the same notation as used in [6] we get*)

$$o_i^1 = \mu_{A_i}(x) \quad (6)$$

Layer 2: The input signals to this layer are subsequently broken down as per certain and uncertain behaviors i.e. the applied input signal propagates to a rough neuron. The non linear parameter set of the weights of the inter neuron links is generated in this layer i.e.

$$o_i^2 = \mu_{BN_i}(x) \times output_{BN_i} + \mu_{R_i}(x) \times output_{R_i} \quad (7)$$

Layer 3: The neurons in layer 3 perform in a fashion similar to those in layer 2, with a negligible difference that the inputs into layer 3 are rough weights from the preceding layer.

$$o_i^3 = \mu_{BN_i}(x) \times output_{BN_i} + \mu_{R_i}(x) \times output_{R_i} \quad (8)$$

Layer 3 helps in further refining the weights on the links between layers 2 & 3 by accommodating rough pattern into the computation as explained earlier. At the end of layers 3, the firing strengths of the nodes are normalized twice based on the output function of the rough neuron.

Layer 4: Neurons in this layer are functionally the same as those in layer 4 of the ANFIS architecture. The Sugeno model based fuzzy if-then rules are now multiplied with the normalized firing strengths. Such a computation results in the generation of the linear parameters.

The linear parameters are weighted parameters governing the fuzzy rules i.e.

$$o_i^4 = o_i^3 \times f_i \quad (9)$$

Layer 5: An individual neuron performs summation upon the input signal to produce the desired output i.e.

$$o_i^5 = \sum o_i^4 \quad (10)$$

C. Parameter Estimation

Hybrid learning is employed to perform parameter approximation. The two different sets of network parameters identified in the previous section can be estimated in two different run directions of the neural network. The linear weighted parameters (generated from the Sugeno fuzzy rules) are approximated in the forward run whereas the non-linear parameters (generated from the weights of the links between neurons) are approximated in the backward run i.e. during the error back propagation. Detailed methodology of the hybrid learning paradigm may be found in [7]. Certain modifications have been made to the conventional technique (discussed in the next section).

IV. Learning and Error Convergence

Error propagation is opposite to the output propagation in direction i.e. error estimation (and convergence) is taken up in the backward runs of the neural network. Conventional neural networks estimate an error function for every neuron in a layer, the output of which is propagated further based on a certain inverse transfer function and weights on the links. The aspect of determining an appropriate error function at each

neuron can turn out to be a laborious task requiring numerous forward and backward runs until the overall system error stabilizes. We propose a relatively modular methodology for error estimation in the backward direction. A secondary neural network has been used to approximate the inverse function for error i.e.

$$error = f(desired - output, obtained - output)$$

$$output = g(input, transfer - function)$$

Thus,

$$error = f \circ g(input, transfer - function) \quad (11)$$

$$input = f \circ g^{-1}(error) \quad (12)$$

The neuro approximated inverse transfer function has been used in order to update the weights (which form the central core of the non linear parameters). The weight update expression can be given as,

$$\overset{+}{_i w_j} = \left[{}_i w_j \times \frac{\varepsilon_i}{\varepsilon_i + \varepsilon_j} \right] + IEF_j \quad (13)$$

where $IEF_j = error_j \times invtrans_j \times \frac{\varepsilon_j}{\varepsilon_i + \varepsilon_j}$ and the $invtrans_j$ is iteratively approximated using a secondary neural network.

The parameter update formula is given as,

$$\Delta\alpha = -\eta \cdot \frac{\partial E}{\partial \alpha} \quad (14)$$

where η is called the *learning rate* and is given as,

$$\eta = \frac{k}{\sqrt{\Sigma_\alpha \left(\frac{\partial E}{\partial \alpha} \right)^2}} \quad (15)$$

where k is the *step size* and governs the speed of convergence towards an optimal solution.

Uncertainty Reduction; The learning algorithm in the backward direction is applied in *two tiers* i.e. two parallel (and simultaneously operating) algorithms indulge in parameter approximation; one algorithm runs through the boundary neurons and the other runs through the lower bound neurons. This type of learning has been implemented in order to segregate the differential behaviors of the two tracks and thus induce a faster convergence rate. Moreover, the algorithm is implemented in its on-line learning form to achieve improved performance. All these characteristics of the learning algorithm have resulted in (the second half of) uncertainty reduction.

V. Testing, Results & Discussion

The proposed methodology was tested on the problem of approximating the landing parameters of a supersonic jet. Theoretical Aerodynamic Equations of various parameters for the landing control of a supersonic jet were approximated and the results compared with approximations obtained through other methods. Emphasis was laid on determining the effect of rough inclusion on the aspects of training time, training epochs, error of approximation and error propagation.

A. Aerodynamic Equations and Modeling

The case of a Supersonic aerodynamics has been chosen to test the proposed architecture taking into consideration the high non-linearity of the governing equations. In particular McDonnell Douglas F/A-18E/F [9] Super Hornets' aerodynamic equations relating to Deceleration Distance and Deceleration Time have been chosen. The equations [10] outline the phenomenon on which the proposed architecture has been tested and have been presented in the appendix to this paper due to column format.

B. Training and Error Efficiency

Training parameters encountered during the numerous test runs have been tabulated in *Table I, II, III and IV*. Two networks were trained and simulated in order to approximate the landing parameters. The parameters of Deceleration Distance and Time have been considered for testing and comparing the various available methodologies.

Deceleration Distance Approximation Network:

TABLE I
Training Parameters while Training

Model	Training err for epoch 1	Total no. of Epochs	Final Error
Proposed	**1.28E-3**	**12**	**5.43E-5**
RNN	**1. 65E-3**	**27**	**3.31E-4**
ANFIS	**218**	**97**	**1.61E-3**
RBFNN	**5732**	**~20,000**	**6.08E-2**

TABLE II
Error Approximation while Training

Model	Error Relative to a Unit output
Proposed	**2.77E-7**
RNN	**1.46E-6**
ANFIS	**8.43E-6**
RBFNN	**1.34E-4**

Deceleration Time Approximation Network:

TABLE III
Training Parameters while Training

Model	Training err for epoch 1	Total no. of Epochs	Final Error
Proposed	**4.12**	**40**	**2.31E-3**
RNN	**5.4**	**120**	**2.63E-3**
ANFIS	**21**	**2000**	**2.49E-3**
RBFNN	**103**	**10500**	**9.1E-3**

TABLE IV
ERROR APPROXIMATION WHILE TRAINING

Model	Error Relative to a Unit output
Proposed	**7.01E-5**
RNN	**1.32E-4**
ANFIS	**8.88E-5**
RBFNN	**8.57E-4**

where,
Proposed model: Rough Approximation based Neuro – Fuzzy Inference System proposed in this paper. **RNN**: [5] based Rough Neural Network developed by Lingras, P. **ANFIS**: [6] based Neuro – Fuzzy Inference System developed by Jang, J. S. R. **RBFNN**: Radial Basis Function based Neural Network.

C. Discussion

It can be clearly observed that the number of training epochs are considerably reduced for the case of the Rough Approximation based Neuro – Fuzzy Inference System (proposed in this paper). The total number of training epochs required to obtain a stable system has been reduced by 55% (in the case of Deceleration Distance) and by 67% (in the case of Deceleration Time).

Further more, it is quite evident from the training error at the end of first iteration for various models that, the Rough Sets based Neuro Fuzzy mechanism is the most effective in pruning the solution space. It should be noted that, a methodology should not only be efficient in the overall training error but also display superiority in aspects pertaining to the training and error convergence.

Lastly and most importantly, the overall stabilized errors (in output approximation) are the least for the proposed methodology. It may be noted that ANFIS has a (varied) better performance over the rough neural network presented in [5], but the proposed methodology has shown reasonable superiority over the other techniques.

These results have been presented only to provide the reader with a better estimate of the various methods. The presented results are by no means comprehensive, thus only preliminary implications about the quality of the techniques should be inferred from them.

VI. CONCLUSION

Rough Approximation based Neuro – Fuzzy Inference System employs various rough set concepts in a synergistic fashion within the fuzzified neural architecture. A new composition of the rough neurons has been adopted. Such a composition of a *boundary* and lower neuron has been implemented within a neuro – fuzzy framework, which is used to approximate continuous functions. A new technique for error back propagation has been devised and implemented. The overall performance of the proposed technique is superior to other comparative models. The presented results (though not extensive) provide a reasonable estimate about the quality of approximation. Pending further analysis, it can be stated that this architecture overcomes the limitation of brittle boundaries in rough approximation through the use of fuzziness.

REFERENCES

[1] Z. Pawlak, *Rough sets: theoretical aspects of reasoning about data*, Kluwer Academic Publishers, London, 1991.
[2] Z. Pawlak, "Rough sets, rough relations and rough functions", *Fundam. Inform*, vol. 27 (2/3), 1996, pp. 103 – 108.
[3] Z. Pawlak, "Rough classification", *Intl. J. of Human Computer Studies*, vol. 51 (2), 1999, pp. 369 – 383.
[4] P.Lingras, Y.Y. Yao, "Data mining using extensions of the rough set model", *J. of American Society for Information Science*, vol. 49 (5), 1998, pp. 415 – 422.
[5] P. Lingras, "Rough neural networks", *In Proc. of the 6th Intl. Conf. on Information Processing and Management of Uncertainty*, Universidad da Granada, Granada, 1996, pp. 1445 – 1450.
[6] J. S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system", *IEEE Trans. on Systems, Man and Cybernetics*, 1993.
[7] J. S. R. Jang, C.T. Sun, E. Mizutani, *Neuro-Fuzzy and soft computing: a computational approach to learning and machine intelligence*, Prentice Hall, New Jersey, 1997.
[8] S. Chandana, R. V. Mayorga, "Rough approximation based neuro-fuzzy inference systems," *In Proc. of IEEE Intl. Conf. on Hybrid Intelligent Systems*, IEEE, Rio, pp. 518 521,2005.
[9] Boeing Inc., Specifications of F/A-18E/F Super Hornet. Weblink. Viewed Dec 2004 (www.boeing.com/defense-space/military/fa18ef).
[10] A. Miele, *Flight mechanics: theory of flight paths*. Addison Wesley. 1962, pp. 107-374.

Sandeep Chandana, obtained his B. Engineering in Mechanical Engineering (spl. Production) from Osmania University in 2001. He is currently working as a graduate student in the Wise & Intelligent Systems & Entities Lab in the Department of Industrial Systems Engineering at the University of Regina, where he is completing his M. A. Sc program. He has worked in projects involving hybridization of the Rough Sets Theory with Neural Networks and Fuzzy Systems. His research interests include Machine Learning, Soft Computing and application of AI to Engineering problems.

Dr. Rene V. Mayorga, is interested in the development of *Computational Sapience [Wisdom]* (as an extension of Computational Intelligence and Soft Computing) and *MetaBotics* (as a generalization of Robotics) as new disciplines. He is involved in the development of Paradigms for Intelligent and *Sapient [Wise]* Systems, *Sapient* Decision & Control, and *MetaBots*; and their application to Robotics (Robot and Multi-Robot Systems Motion Planning and Design), Automation, Manufacturing (Decision Making, Optimization), Human-Computer Interaction/Interface, and Software Engineering. Dr. Mayorga is the Editor of the journal of *Applied Bionics and Biomechanics*, and Associate Editor of the *Journal of Intelligent Service Robotics.* He has been in several recent occasions the General Chair of the *ANIROB International Symposium of Robotics and Automation* and Editor of the corresponding Proceedings.

APPENDIX – AERODYNAMIC EQUATIONS [10]

Deceleration Distance:

$$X = \frac{W}{2\rho g S C_{DO}} \left[\log\left(1 + \frac{V_i^4 \rho^2 S^2 C_{DO}}{4W^2 K^2}\right) - \log\left(1 + \frac{V_f^4 \rho^2 S^2 C_{DO}}{4W^2 K^2}\right) \right]$$

Deceleration Time:

$$T = \sqrt{\frac{W}{2g^2 \rho S \sqrt{K C_{DO}^3}}} \left[\left(\frac{1}{\sqrt{2}} \left(\left(\arctan\left(\frac{2V_i \sqrt{\rho SWK\sqrt{C_{DO}}}}{2WK - V_i^2 \rho S \sqrt{C_{DO}}} \right) \right) - \left(\frac{1}{2} \log\left(\frac{2WK + V_i^2 \rho S \sqrt{C_{DO}} + 2V_i \sqrt{\rho SWK\sqrt{C_{DO}}}}{2WK + V_i^2 \rho S \sqrt{C_{DO}} - 2V_i \sqrt{\rho SWK\sqrt{C_{DO}}}} \right) \right) \right) \right) - \ldots \right.$$

$$\left. \ldots \left(\frac{1}{\sqrt{2}} \left(\left(\arctan\left(\frac{2V_f \sqrt{\rho SWK\sqrt{C_{DO}}}}{2WK - V_f^2 \rho S \sqrt{C_{DO}}} \right) \right) - \left(\frac{1}{2} \log\left(\frac{2WK + V_f^2 \rho S \sqrt{C_{DO}} + 2V_f \sqrt{\rho SWK\sqrt{C_{DO}}}}{2WK + V_f^2 \rho S \sqrt{C_{DO}} - 2V_f \sqrt{\rho SWK\sqrt{C_{DO}}}} \right) \right) \right) \right) \right]$$

where,

V_i Initial Aircraft Flying Velocity; V_f Touchdown Aircraft Velocity
V Average velocity; ρ Air Density
S Wing Surface Area; K Induced Drag Coefficient
C_{DO} Zero Drag Lift Coefficient; W Aircraft Weight
C_L Lift Coefficient; g Acceleration due to Gravity

arctan Tan^{-1} (inverse Tangent)