Cancer Classification using Adaptive Neuro Fuzzy Inference System with Runge Kutta Learning

C. Loganathan, Ph.D
Department of Mathematics
Maharaja Arts and Science College
Tamilnadu, India – 641407

K.V.Girija
Department of Mathematics
Hindusthan College of Engineering & Technology
Coimbatore, Tamilnadu, India – 641432

ABSTRACT
Cancer research is one of the major research areas in the medical field. Adaptive Neuro Fuzzy Interference System is used for the classification of Cancer. This algorithm compared with proposed algorithm of Adaptive Neuro Fuzzy Interference system with Runge Kutta learning method for the best classification of cancer. It is one of the better techniques for the classification of the cancer. The Adaptive Network-based Fuzzy Inference System is one of the well-known neural fuzzy controllers with fuzzy inference capability. For the cancer classification inputs are collected from the dataset of Lymphoma dataset and Leukemia dataset. In this paper, focused in classification of cancer by using ANFIS with RKLM.

Keywords
ANFIS, RKLM, ANFIS with RKLM.

1. INTRODUCTION
To introduce ANFIS (Adaptive-Network-Based Fuzzy Inference Systems) as a diagnosis and classification system, ANFIS has proven to be an excellent function approximation tools. Here it implements a first order Sugeno-style fuzzy system. Chief idea behind these Neuro-adaptive learning techniques is very simple. This type of techniques provide membership function parameters that best allow the associated fuzzy inference system track the given input/output data. he

In [6] proposed the Gene Selection for Cancer Classification using Support Vector Machines. In this paper, their problems are recorded in the DNA micro-arrays that are the selection of a small subset of genes from broad patterns of gene expression data [9, 8]

Neuro fuzzy systems are fuzzy systems which use ANNs theory in order to determine their properties (fuzzy sets and fuzzy rules) by processing data samples. In Neuro fuzzy development a specific approach is called adaptive Neuro fuzzy inference system (ANFIS), which has shown significant results in modeling nonlinear functions. Learning features for ANFIS in the data set adjusts the system parameters according to a given error criterion [8]. Successful implementations of ANFIS in biomedical engineering have been reported, for classification [9] and data analysis [10, 11]

In [12] successfully classified lymphoma data set with only 48 genes by using a statistical method called nearest shrunken centroids with an accuracy of 100%. For the SRBCT data, [12] classified all 20 testing samples with 96 genes. They used a two-layered linear neural network.

In [13] applied nearest shrunken centroids to the SRBCT data set and then they obtained 100% accuracy with 43 genes. In [14] they reduced number of genes required to correctly classify the four cancer sub-types in the SRBCT data set to 12 genes. In [13] obtained 100% accuracy in this data set with a SVM classifier and the separability based gene importance ranking. They used at least 20 genes to obtain this result. At the same time, three principal components (PCs) were generated from the 20 top genes. Their SVM also obtained 100% accuracy in the space defined by these three principal components.

1.1. Adaptive Neuro Fuzzy Inference Systems (ANFIS)
Adaptive Neuro Fuzzy Inference System (ANFIS) was first proposed by Jang. ANFIS can be easily implemented for a given input/output task and hence it is attractive for many application purposes. It has been successfully applied in different areas. The first kind of NFS we apply for our data classification problem is the so called Adaptive Neuro-Fuzzy Inference System (ANFIS) model, which hybridizes an ANN and a FIS into a kind of NFS with a homogeneous structure. That is, the ANFIS model integrates the ANN and FIS tools into a 'compound', meaning that there are no boundaries to differentiate the respective features of ANN and FIS. [5].

1.1.1. ANFIS Architecture
In order to describe the architecture of an ANFIS, we briefly introduce the first order Sugeno-style FIS at first. A first-order Sugeno-style FIS model is a system that manages the process of mapping from a given crisp input to a crisp output, using fuzzy set theory

$$\text{If } x_1 \text{is } A_1, x_2 \text{ is } A_2, \ldots . x_n \text{ is } A_n$$
$$\text{then } y = k_0 + k_1 x_1 + k_2 x_2 + \cdots + k_n x_n$$

Where $x_1, x_2, \ldots . , x_n$ are input variables; $A_1, A_2, \ldots . , A_n$ are fuzzy sets; and y is the output variable. Find that in such type of fuzzy rule. The output variable is a first-order polynomial on input variables.

Description of the method
An ANFIS is in essence an ANN that is functionally Equivalent to a first order Sugeno-style FIS. Typically there are six layers in an ANFIS model; one input layer, four hidden layers and an output layer. Each layer performs task to forward the signals. Such an ANFIS model is shown in figure 1.

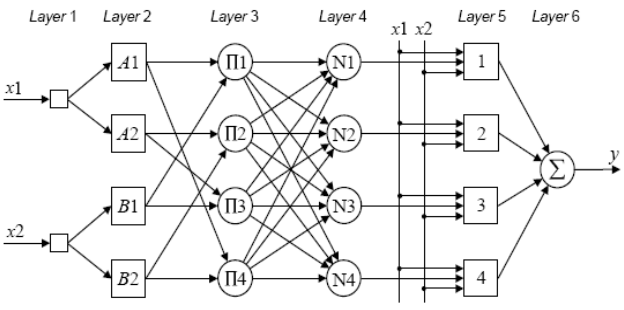

Figure 1 A typical ANFIS model with two inputs and one output.

The first layer, i.e. the input layer of the ANFIS model is the input layer. Neurons in this layer simply transmit the external input (crisp) signals to the next layer. Namely

$$x_i^1 = y_i^1 \quad (1)$$

Where x_i^1 the input is signal and y_i^1 is the output signal of neuron i in the first layer.

The second layer, i.e. the first hidden layer of the ANFIS model is fuzzification layer. Neurons in this layer represent antecedent fuzzy sets of fuzzy rules. A fuzzification neuron here receives an input signal and determines the degree to which this signal belongs to the neuron's fuzzy set. Let x_i^2 be the input and y_i^2 be the output signal of neuron i in the second layer, then have

$$y_i^2 = f(x_i^2) \quad (2)$$

Where f represents the activation function of neuron i and is set to a certain membership function.

The third layer, i.e. the second hidden layer is the fuzzy rule layer. Each neuron in this layer corresponds to a single first-order Sugeno fuzzy rule - a rule neuron receives signals only from the fuzzification neurons which are involved in the antecedents of the fuzzy rule it represents and computes the truth value of the rule. In an ANFIS, the 'product' operator is used to evaluate the conjunction of the antecedents. Therefore

$$y_i^3 = \prod_c^m x^3 c_i \quad (3)$$

Where $x^3 c_i$ is the signal from fuzzification neuron c in the second layer i in the third layer. y^3 is the output signal of neuron i in this layer, and m is the number of antecedents of the fuzzy rule neuron i represents.

The fourth layer, i.e. the third hidden layer is the normalization layer. Each neuron in this layer receives signals from all rule neurons in the third layer and calculates the so-called normalized firing strength of a given rule. This strength value represents the contribution of a given rule to the final result and is obtained as

$$y_i^4 = \frac{x_{d_i}^4}{\sum x_{d_i}^4} \quad (4)$$

Where $x_{d_i}^4$ the signal from neuron d in the third is layer to neuron i in the fourth layer, y_i^4 is the output signal of neuron i in this layer, and n is the number of rule neurons in the third layer.

The fifth layer, i.e. the fourth hidden layer is the defuzzification layer. Each neuron in this layer is connected to the respective normalization neuron in the fourth layer and also receives initial input signals $x_1, x_2, \ldots\ldots, x_n$. A defuzzification neuron computed the 'weighted consequent value' of a given rule as:

$$y_i^5 = x_i^5(k_{i0} + k_{i1}x_1 + k_{2i}x_2 + \ldots\ldots + k_{in}x_n) \quad (5)$$

Where x_i^5 is the input and y_i^5 is the output signal of neuron i in the fifth layer; and $k_{i0}, k_{i1}, k_{i2}, \ldots\ldots, k_{in}$ is a set of consequent parameters of rule .

The sixth layer, i.e. the output layer is the summation layer. There is only one neuron in the layers, which calculated the sum of outputs of all defuzzification neurons in the fifth layer and consequently produce the overall ANFIS output y as follows

$$y = \sum_{i=1}^{n} x_i \quad (6)$$

Where x_i is the signal from defuzzification neuron i in the fifth layer to this summation neuron; and n is the number of defuzzification neurons, namely the number of fuzzy rules in the ANFIS model.

1.1.2. Training ANFIS Model

ANFIS can be trained to learn from given data. As we can observe from the ANFIS architecture, in order to configure an ANFIS model for a specific problem, we need to specify the fuzzy rules and the activation functions (i.e. membership functions) of fuzzification neurons. For the fuzzy rules, we can specify the antecedent fuzzy sets once we know the specific problem domain; while for the consequents of the fuzzy rules, the parameters (e.g. $k_{i0}, k_{i1}, k_{i2}, \ldots\ldots, k_{in}$) are formed and adjusted by certain learning algorithm in the training process. On the other hand, the shapes of activation functions can also be formed and adjusted in the training process.

For ANFIS models, the most commonly used activation function is the so-called bell-shaped function, described as:

$$y = \frac{1}{1 + \left[\left(x - \frac{s}{r}\right)^2\right]^t} \quad (7)$$

Where r, s and t are parameters that respectively control the slope, center and width of the bell-shaped function. And in the training process, these parameters can be specified and adjusted by the learning algorithm.

Specifically, an ANFIS uses a 'hybrid' learning (training) algorithm. This learning algorithm combines the so-called least-squares estimator and the gradient descent method finally with Runge-Kutta learning method. In the beginning, initial bell-shaped functions with certain parameters are assigned to each fuzzification neuron. The function centre of the neurons connected to input xi are set so that the domain of xi is divided uniformly, and the function widths and slopes are set to allow sufficient overlapping(s) of the respective functions.

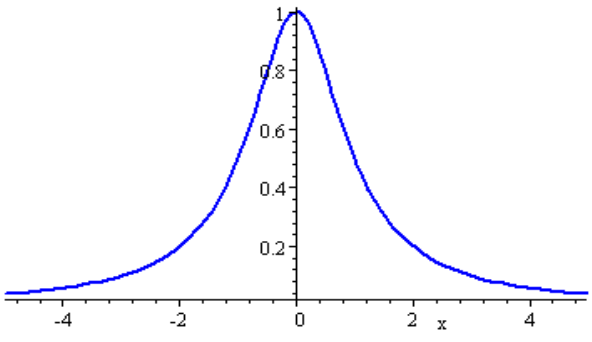

Figure 2 shows a bell-shaped function with r = t = 1 and s = 0.

During the training process, the training dataset is presented to the ANFIS cyclically. Each cycle through all the training examples is called an epoch. In the ANFIS learning algorithm, each epoch comprises of a forward pass and a backward pass. The purpose of the forward pass is to form and adjust the consequent parameters, while that of the backward pass is to adjust the parameters of the activation functions. [11]

1.1.3. ANFIS with RKLM

RKLM is a powerful way of solving the behavior of a dynamic system if the system is characterized by ordinary differential equations [17]. This method is examined for the successful estimation of the system states while long run. It should be emphasized that the neural network architecture realizes the changing rates of the system states instead of the

$[x(k), \tau(\square k)\square] \rightarrow \square[x(k+1)]$ mapping. Thus the RKNN approach can be used to alleviate the difficulties caused due to the discretization methods. The first order discretization method brings large approximation errors. Here h denotes the Runge-Kutta integration step size.

In the preceding discussed the Euler method; a fairly simple iterative algorithm for determining the solution of an initial value problem.

$$\frac{dx}{dt} = F(t,x) \quad , \quad x(t_0) = x_0 . \tag{8}$$

The key idea was to interpret the F(x, t) as the slope m of the best straight line fit to the graph of a solution at the point (t, x). Knowing the slope of the solution curve at (t_0, x_0) we could get to another (approximate) point on the solution curve by following the best straigh t-line-fit to a point $(t_1, x_1) = (t_0 + \Delta t, x_0 + m_0\Delta t)$,where $m_0 = F(t_0, x_0)$. And then we could repeat this process to a third point $(t_2, x_2) = (t_1 + \Delta t, x_1 + m_1\Delta t)$, and so on. By iterating this process n times gives us a set of n + 1 values $x_i = x(t_i)$ for an approximate solution on the interval $[t_0, t_0 + n\Delta t]$.

Now recall from our discussion of the numerical procedures for calculating derivatives that the formal definition

$$dx/dt = \lim_{h\to 0} \frac{x(t+h)-x(t)}{h} \tag{9}$$

does not actually provide the most accurate numerical procedure for computing derivatives. For while

$$\frac{dx}{dt} = \frac{x(t+h)-x(t)}{h} + 0(h) \tag{10}$$

A more accurate formula would be

$$dx/dt = 4/3h\,(x(t+h/2) - x(t-h/2)) - 1/6h\,(x(t+h) - x(t)) + 0(h\hat{}4) \tag{11}$$

Let us begin with the Taylor series for x (t + h):

$$x(t+h) = x(t) + hx'(t) + \frac{h^2}{2}x''(t) + \frac{h^3}{2}x'''(t) + 0(h^4) \tag{12}$$

From the differential equation we have,

$$x'(t) = F \tag{13}$$

$$x''(t) = F_t + F_xF \tag{14}$$

$$x'''(t) = F_{tt} + F_{tx}F + (F_{xt} + F_{xt})F + (F_t + F_xF)F_x \tag{15}$$

And so the Taylor series for x(t + h) can be written

$$x(t+h) = x(t) + hF(t,x) + \frac{h^2}{2}F_t(t,x) + F_x(t,x)) + 0(h^3)$$

$$= x(t) + \frac{1}{2}hF(t,x) + \frac{1}{2}h\,(F(t,x) + hF_t(t,x) + hF(t,x)F_x(t,h)) + 0(h^3) \tag{16}$$

Now, regarding F (t + h, x + hF(t, x)) as a function of h, we have the following Taylor expansion,

$$F(t+h, x+hF(t,x)) = F(t,x) + hF_t(t,x) + F_x(t,h)(hF(t,h)) + 0(h^2) \tag{17}$$

$$x(t+h) = x(t) + \frac{h}{2}F(t,x) + \frac{h}{2}F(t+h, \tag{18}$$

$$x + hF(t,x)) + 0(h^3)$$

$$x(t+h) = x(t) + \frac{1}{2}(F_1 + F_2) \tag{19}$$

where $\mathbf{F_1 = hF(t,x)}$

$$\mathbf{F_2 = hF\,(t+h, x+F_1)} \tag{20}$$

Partition the solution interval [a,b] into n subintervals:

$$\mathbf{\Delta t = \frac{b-a}{n}} \tag{21}$$

$$\mathbf{t_k = a + k\Delta t} \tag{22}$$

Set x_0 equal to x (a) and then for k from 0 to n-1 calculate

$$\mathbf{F_{1,k} = \Delta tF(t_k, x_k)} \tag{23}$$

$$\mathbf{F_{2,k} = x_k + 1/2\,(F_{1,k} + F_{2,k}))} \tag{24}$$

This method is the second order Runge-Kutta method.

Higher order Runge-Kutta methods are also possible.Here is the formula for the classical fourth-order Runge-Kutta method:

$$\mathbf{x\,(t+h) = x(t) + \frac{1}{6}\,(F_1 + 2F_2 + 2F_3 + F_4)} \tag{25}$$

Where

$$\mathbf{F_1 = hF\,(t,x)}$$

$$\mathbf{F_2 = hF\,(t + \frac{1}{2}h, x + \frac{1}{2}F_1)}$$

$$\mathbf{F_3 = hF\,(t + \frac{1}{2}h, x + \frac{1}{2}F_2)}$$

$$\mathbf{F_4 = hF\,(t + h, x + F_3)}$$

2. ANFIS with Back propagation

Based on approach of error correction learning, Back propagation is a systematic method for training, provides a computationally efficient method for changing the synaptic weights in the neural network, with differentiable activation function units. The back propagation algorithm uses method of

supervised learning. We provide the algorithm with the recorded set of observations or training set [18] i.e. examples of the inputs and the desired outputs that we want the network to compute, and then the error (difference between actual and expected results) is computed. These differences in output are back propagated in the layers of the neural networks and the algorithm adjusts the synaptic weights in between the neurons of successive layers such that overall error energy of the network, E is minimized. The idea of the back propagation algorithm is to reduce this error, until the ANN learns the training data. Training of network i.e. error correction is stopped when the value of the error energy function has become sufficiently small and as desired in the required limits[15].Total error for $\mathbf{pth}$ observation of data set and $\mathbf{jth}$ neuron in the output layer can be computed as:

$$\mathbf{E_j = t_j - y_j} \quad (26)$$

Where $\mathbf{t_j}$ represents the desired target output and $\mathbf{y_j}$ represents the predicted from the system.

3. EXPERIMENT RESULTS

3.1. Classification on cancer data set

The proposed methodology was applied to the publicly available cancer datasets namely Lymphoma and Leukaemia cancer dataset and the experimented using MATLAB.

3.1.1. Lymphoma dataset

Lymphoma data set [9] contain 42 samples derived from diffuse large B-cell lymphoma (DLBCL) and 9 samples from follicular lymphoma (FL) after that 11 samples from chronic lymphocytic leukaemia (CLL). The entire data set contain 4026 genes. In this data set, a small part of data is missing.

Applied the combination of ANFIS to classify the lymphoma microarray data set. The first choose some important genes using a feature importance ranking scheme. This ranking Scheme are based on the t- score (TS) is a t-statistic between the centroid of a specific class and the overall centroid of all the classes. Another possible model for TS could be a t-statistic between the centroid of a specific class and the centroid of all the other classes.

First added the selected 50 genes one by one to the network according to their TS ranks starting with the gene ranked. At first used only a single gene that is ranked 1 as the input to the network. Then trained the network with the training data set, and subsequently tested the network with the test data set then repeat the process so on.

3.1.2. Leukemia dataset

The leukemia data set contains expression levels of 7129 genes taken over 72 samples. Labels indicate which of two variants of leukemia is present in the sample. This dataset is of the same type as the colon cancer dataset and can therefore be used for the same kind of experiments. Then classify the leukemia dataset by applying ANFIS with RKLM for their selected 50 genes one by one to the network according to their TS ranks starting with the gene ranked.

Table I Performance Analysis

	ANFIS			ANFIS with RKLM		
	Epochs	**Time (sec)**	**Accuracy (%)**	**Epochs**	**Time(sec)**	**Accuracy (%)**
Lymphoma dataset	80	12	89	35	7	93.55
Leukemia dataset	63	9	87	29	5	92.88

A modular classification algorithm based on the ANFIS with RKLM has been used for this application. Experiment results shown the comparison of ANFIS with RKLM and ANFIS. Two types are dataset are taken for the classification. Selected genes are taken from the datasets. In table I give the comparison between ANFIS and ANFIS with RKLM for the classification of cancer based on Epochs, Time in seconds, and Accuracy in percentage. From the table clearly noticed that the ANFIS with RKLM gives the better results, it have the less epoch time of 35 and their processing time is 7 for the lymphoma data set ANFIS with RKLM classifier and searched for gene combinations among the top 50 genes and achieved the 93.55% accuracy for the ANFIS 89% accuracy. For the leukemia dataset achieved their epoch's time of 29 and their classification time is 5 seconds accuracy of 92.89% for the ANFIS 87%. Finally concluded proposed method of ANFIS with RKLM provides better accuracy than ANFIS.

Thus the comparison shows that the ANFIS method has low classification performance than ANFIS with RKLM. Proposed method of ANFIS with RKLM gives the best results.

4. CONCLUSION

Analyzes the performance of classification of cancer using ANFIS with RKLM and ANFIS. In this assessment, the estimation performance, together with the training error rate is considered as the primary comparison measures. From the result it is seen that the performance of ANFIS with RKLM gives the best in estimation. The work, reported in this paper, indicates that ANFIS structure is a good candidate for identification purposes. Additionally, the elegant performance of the RKLM approach with on-line operation and with ordinary feed forward neural network. The work is in progress in the direction of improving the estimation performance through the use of ANFIS with RKLM approaches in combination.

5. REFERENCES

[1]. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, et al., "Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling," Nature, vol. 403, pp. 503–511, 2000.

[2]. S.Y. Belal, A.F.G. Taktak, A.J. Nevill, S.A. Spencer, D. Roden, S. Bevan, "Automatic detection of distorted plethysmogram pulses in neonates and paediatric patients using an adaptive-network-based fuzzy inference system," Artificial Intelligence in Medicine, vol. 24, pp. 149-165, 2002.

[3]. Chen Liao and Shutao Li, "A support vector machine ensemble for cancer classification using gene expression data", ISBRA'07: Proceedings of the 3rd international conference on Bioinformatics research and applications, 2007.

[4]. J. Deutsch, "Evolutionary algorithms for finding optimal gene sets in microarray prediction," Bioinformatics, vol. 19, pp. 45–52, 2003.

[5]. Dudoit, J. Fridlyand, and T. P. Speed, "Comparison of discrimination methods for the classification of tumors using gene expression data," Journal of the American Statistical Association, vol. 97, pp. 77–87, 2002.

[6]. Isabelle Guyon, Jason Weston, Stephen Barnhill, Vladimir Vapnik. Gene Selection for Cancer Classification using Support Vector Machines", Machine Learning, Vol. 46, No. *1*-3 pp. 389-422. 2002..

[7]. Jamal M. Nazzal, Ibrahim M. El-Emary and Salam A. Najim, "Multilayer Perceptron Neural Network (MLPs) For Analyzing the Properties of Jordan Oil Shale", World Applied Sciences Journal 5.

[8]. Jang.J.S.R and Sun.C.T, Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1997.

[9]. J.-S.R. Jang, "ANFIS: Adaptive-network-based fuzzy inference system," IEEE Transactions on Systems, Man, and Cybernetics, vol. 23(3), pp. 665-685, 1993.

[10]. J. M. Khan, J. S. Wei, M. Ringner, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson, et al., "Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks," Nature Medicine, vol. 7, pp. 673–679, 2001.

[11]. C..Loganathan and K.V.Girija "A New Hybrid Learning for Adaptive Neuro Fuzzy Inference System" International Journal of Computer Science And Technology www. ijcst . com IJCST Vol. 4, Issue 3, July - Sept 2013 .

[12]. Mingjun Song and Sanguthevar Rajasekaran, "A greedy algorithm for gene selection based on SVM and correlation" International journal of Bioinformatics Research and Applications, 2010.

[13]. R .Tibshirani, T. Hastie, B. Narasimhan, and G. Chu, "Class predicition by nearest shrunken centroids with applications to DNA microarrays," Statistical Science, vol. 18, pp. 104–117, 2003.

[14]. R. Tibshirani, T. Hastie, B. Narashiman, and G. Chu, "Diagnosis of multiple cancer types by shrunken centroids of gene expression," Proc. Natl. Acad. Sci. USA, vol. 99, pp. 6567–6572, 2002.

[15]. O. Troyanskaya, M. Cantor, G. Sherlock, et al., "Missing value stimation methods for DNA microarrays," Bioinformatics, vol. 17, pp. 520–525, 2001.

[16]. I. Virant-Klun, J. Virant, "Fuzzy logic alternative for analysis in the biomedical sciences," Computers and Biomedical Research, vol. 32, pp. 305-321, 1999.

[17]. Wang, Y-J., C-T. Lin, "Runge-Kutta Neural Network for Identification of Dynamical Systems in High Accuracy", IEEE Transactions on Neural Networks, Vol. 9., No. 2, pp. 294-307, March 1998.

[18]. http://en.wikipedia.org/wiki/Stochastic_gradient_descent,29 th January 2012.