Bull. Astr. Soc. India (1998) **26**, 667-684

Mapping H-R diagram onto a neural network

K. Ravindra Shetty[1], Ashok Rao[2] and K. Gopala[3]

[1] *Dept. of Electronics, S.D.M. College, K.M. Puram, Mysore 570004, India*

[2] *Dept. of E and C Eng., S.J.C.E. Mysore 570006 and CART, NIE, Mysore 570008, India*

[3] *Dept. of Physics, University of Mysore, Mysore 570006, India*

Received 10 February 1998; accepted 30 July 1998

Abstract. A study has been undertaken to classify stars in an H-R diagram using a neural network by setting the input feature in max-min-max Sub arrays. The input features we have considered are B-V color index, absolute magnitude and temperature in all five classes namely hot subdwarfs, white dwarfs, main sequence, normal giants and red giants. For classifying the output we have taken twenty eight stars which cover all above said five classes and have achieved an accuracy of 86.6%. Further we have evaluated the dependencies of the neural network output on different confidence interval widths. In addition we have studied the normalized system error on the number of hidden layers and the minimum number of hidden layers and the minimum number of input features that are required for output prediction. We would like to call this algorithm as max-min-max Generalized Delta Rule.

Key words : neural networks, star classification, H-R diagram

1. Introduction

It is known that the human brain is built of cells called neurons. A collection of neurons linked together in such a way that individual neurons can perform separate functions simultaneously is called a neural network. The computational model neural network called Artificial Neural Network (ANN) is characterized by the network topology, the connection strength between pairs of neurons (weights), node properties and the status-updating rules (Simon Haykin 1995). The ability to learn is a fundamental trait of intelligence. Learning in neural networks whether Supervised (training with teacher), Unsupervised (training without teacher), or Hybrid (combines supervised and unsupervised learning) is accomplished by adjusting weights between connections in response to new inputs or training patterns (Yi Shang et al. 1996). Work on ANN models has a long history. Development of detailed mathematical models began more than 43 years ago. Perceptron rule LMS algorithm, Stein bucties learning matrix, madaline Rule 1, mode seeking technique (first Competitive learning), Widrows reinforcement learning algorithm, Grossberg's Adaptive Resonance theory (ART), Outer product rules and equivalent approach of Hopfield and others, Adaptive Bi-directional Associative Memory (BAM), Generalized Delta Rule (GDR), etc., are the sequence of land marks in the neural network evolution (Bernard Widrow 1990).

2. ANN overview

An artificial neuron Fig. 1a has a set of inputs. X_1, X_2, X_3, ..., X_n collectively denoted by the vector X. Each input signal is multiplied by a weight W_1, W_2, W_3,...., W_n respectively and

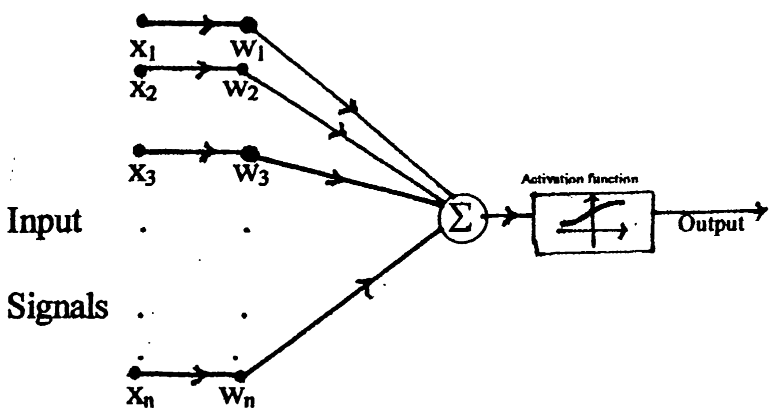

Figure 1a. Nonlinear model of a Neuron.

applied to an adder for summing the input signals, weighted by the respective synapses of the neuron : the operations described here constitute a linear combiner. The output called (NETX) is

$$\text{NETX} = \sum_{j=1}^{n} W_{ij}\, Xj$$

The NET signal is processed by any nonlinear activation (squashing) function f (Commonly used activation functions are threshold function, piecewise linear sigmoid and Gaussian) are shown in Fig. 1b, to produce the neurons output OUT.

$$\text{Output is} = f_s \sum_{j=1}^{n} W_{ij}\, Xj = f\ s\ \text{NET X} \tag{2.1}$$

where f_s is a Sigmoid function

There is a crude analogy here to a biological neuron : wires and interconnections model axons and dendrites, connection weights represent synapses, and the threshold function approximates the activity in Soma.

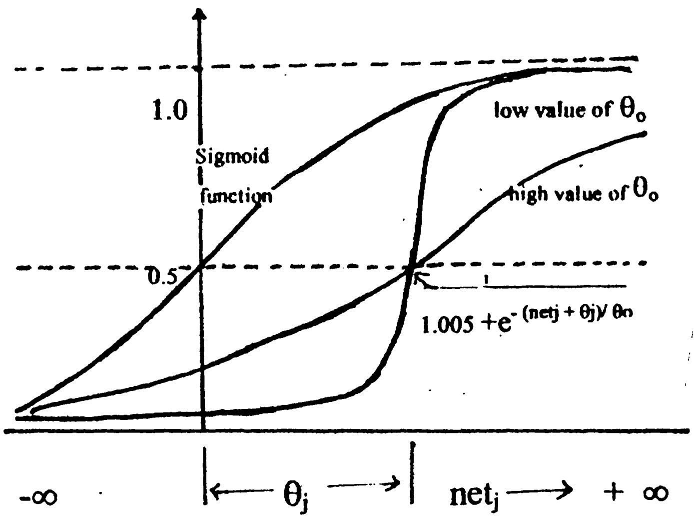

Figure 1b. The modified sigmoid function.

By connecting neurons into networks, their power can be greatly increased. Capabilities of the network can be further enhanced by cascading a group of single layer neural networks into a multilayer neural network. Output of one layer is the input to the next layer as shown in Fig. 1c. It is the presence of the nonlinear activation function which makes a multilayer ANN more powerful than a single layer ANN. This structure is known as a nonrecurrent or feedforward network. If feedback connections are present, the network is called recurrent. Non recurrent networks have no memory. Their output is solely determined by current inputs and weights. Recurrent networks exhibit properties similar to that of short term memory in humans (Jain et al. 1996).

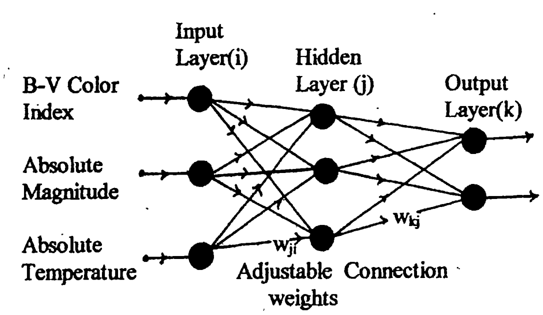

Figure 1c. Basic architecture of an ANN for star classification.

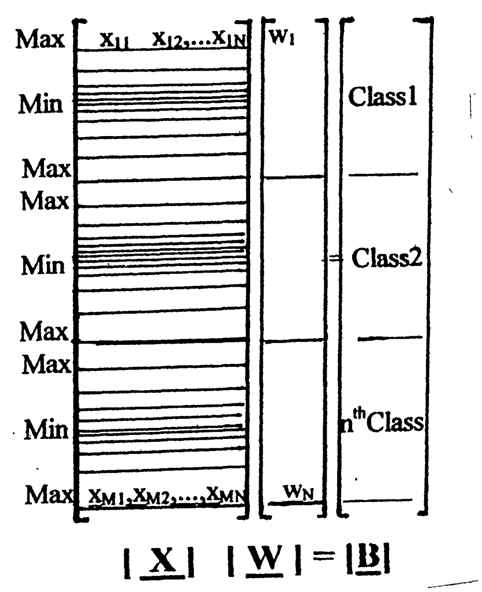

Figure 1d. Max-Min-Max Sub arrays of patterns in N dimension (Refer to Box 1).

3. Network architecture and methodology

Before this decade most people believed that a Neural Network model is crude, but at present the field of ANN is enjoying great popularity. We have used neural networks to classify stars in an H-R diagram, as there is no known mathematical models for the prediction or classification of such a problem. To increase the capability for the network we have used multilayer network. For better accuracy of the output we have used modified sigmoid function at the internal representation units and the output at node j Fig. 1c is f_j (x) and is given by,

$$f_j(x) = \frac{1}{1.005 + e^{-(\text{Net } X + \theta j)/\theta_0}} \tag{3.1}$$

(where NetX = $\sum_{j=1}^{n} W_{ij} X_j$, θ_j serves as a threshold or bias its function. It is to shift the activation function to the left along the horizontal axis, θ_o will modify the shape of the Sigmoid function as shown in Fig. 1b).

We have trained the NN for different numerical constants (1.5, 1.45,...., 0.50) at the denominator of equation (3.1), and achieved minimum error for 1.005. This function is highly nonlinear, continuous and continuously differentiable. The Back Propagation Algorithm (BPA) was originally introduced by Paul Werbos in 1974. The BPA is an iterative gradient algorithm designed to minimize the mean square error between the actual output of multilayer feedforward perceptron and the desires output. BPA is a hill climbing algorithm based on the steepest (or equivalent) descent principle. It adjusts parameter by an amount proportional to the component of the gradient of the error in the direction of the parameter. This algorithm has also been called "the Generalized Delta Rule (GDR)". The neurons in layers other than the input and output layers are called hidden units or hidden nodes, as their outputs do not directly interact with the environment. With BPA, the weights associated with hidden layers can also be adjusted and the multilayer ANN is thus enabled to learn.

4. Star classification

Around 1910 Ejnar Hertz Sprung in Denmark and Henry Norris Russel at Princeton University in the United States independently plotted the graph of brightness of the stars versus temperature of stars is called H-R diagram. The reason for the existence of the discrete sequences in H-R diagram can be understood in terms of the Vogt-Russel theorem (Chandrashekar 1939, 1951) and its discussion illustrated here (Chandrashekar 1939, Flugge 1958, Johnson et al. 1953). In interpreting the H-R diagram it is important to remember that star formation and evolution is an ongoing process (Chandrashekar 1951).

We have taken five classes namely Hot subdwarfs, White dwarfs, Main Sequence, Normal Giants and Red supergiants, with fifty stars (50) from each class and a total of two hundred fifty (250) stars for training. These five classes (dashed line regions) and accepted terminologies in the H-R diagram are shown in Fig. 2.

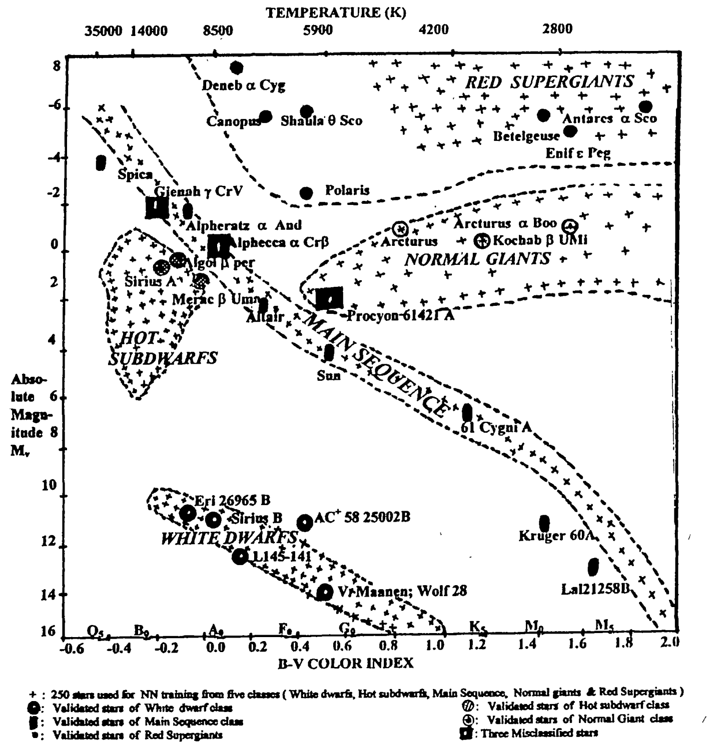

Figure 2. Neural Network prediction of the Hertz Sprung - H.N Russel (HR) diagram.

5. Computer implementation and discussion of the neural network output

In the case of supervised (training with teacher) learning, we may think of the teacher as having knowledge of the environment that is represented by a set of input-output examples. The environment is, however, unknown to the neural network. Suppose now that the teacher and the neural network are both exposed to a training vector (i.e. example) drawn from the environment. By virtue of built in knowledge, the teacher is able to provide the neural network with a desired or target response for that training vector. If this is repeated for all examples known to the teacher it is equivalent to training the Neural Network.

Our main objective is to make the neural network to classify a given input (Star Classes), in the following way. Neural network can predict and classify, this classification is based on training. Thus effective training ensures that the neural network output is accurate and acceptable. Thus showing the classified output of a neural network which is an "apparent" predictive process, is actually a Least Mean Square (LMS) classification process sufficiently accurate for enduse. Practically speaking this can be ensured if only a "proper" choice of training set is used. What constitutes such a choice? Ideally if the neural network is trained with a training set comprising of all possible input-output relations that it will ever encounter in future also. This in many cases is impossible, impractical and unnecessary too (for example to recognize English words we only need twenty six characters namely from a to z). Thus rather a collection of "rich" set of inputs is more meaningful. This richness can be easily seen to be a representation of the diverse and significant features that are to be found in the exhaustive set. For our problem we have chosen fifty stars from each class namely, Hot Subdwarfs, White dwarfs, Main sequence, Normal giants and Red giants, their significant parameters associated with respective class.

We want to check now if this is a rich set? Training is done with data, the output of which is known. This training set is to be validated to ensure that the Neural Network has been subjected, to the entire spectrum of choice of inputs. This can be done if further inputs whose outputs are known is given at the input to check the effectiveness by training.

Now to validate whether the training set is "rich" for this, we gave input and checked the output from trained neural network. For the same input without output features it should give correct classification. We achieved 100% accuracy. Thus the performance of the neural network is highly dependent on training and obviously our training process is correct. For obtaining predicted/classified outputs for the data (the output for which is truly unknown) we fed data of twenty eight stars (Allen 1974) belonging to all classes. The following steps (i) to (vii) gives the computer implementation of this process related to a neural network.

(i) The three input parameters (B-V Color index, Absolute magnitude, and Temperature) for the five classes namely, Hot subdwarfs, White dwarfs, Main Sequence, Normal Giants and Red giants, consisting of fifty stars in each class (Sample data's are shown in Appendix B) are taken.

(ii) Each of these values are normalized between 0 and 1 and is shown in appendix C.

(iii) The above values are encoded as a max-min-max pattern with output features as shown in Box 1.

(iv) This is used to train the Neural Network for 2000 iterations (The C language implementation on CYBER 180 is shown in Appendix A).

(v) The output after 2000 iterations (convergence limit that we have chosen) is compared with known data.

(vi) For validating the training we fed the same input data's as in step (iii) without output features and we achieved 100% accuracy in classification.

(vii) This trained and validated network is now used to classify data the output of which is unknown. This is done by feeding the three parameters (as in step (i)) at the input of the Neural Network and the output from this Neural Network is the classified answer. In our case we choose data totalling twenty eight stars (Allen C W 1974) belonging to all classes as given in Appendix D and the results (classified output) are shown in table 1.

(viii) The process is repeated all over by first training the neural network for 5000, 10,000 iterations and we studied the normalised system error as a function number of hidden layers (1 to 5) and number of units in a hidden layer (1 to 10).

Table 1. Evaluation score table of the star classification.

Star	Actual Class	Neural Network Prediction	Evaluation Score
Algol β Per	Hot Subdwarf	Hot Subdwarf	T
Merak β UMa	Hot Subdwarf	Hot Subdwarf	T
Sirius A	Hot Subdwarf	Hot Subdwarf	T
V. Maanen; Wolf 28	White Dwarf	White Dwarf	T
Eri; 26965B	White Dwarf	White Dwarf	T
Ac⁺ 58 25002B	White Dwarf	White Dwarf	T
L145-141	White Dwarf	White Dwarf	T
Sirius B	White Dwarf	White Dwarf	T
Alpheratz α And	Main Sequence	Hot Subdwarf	T
Alphecca α CrB	Main Sequence	Hot Subdwarf	F
Spica	Main Sequence	Main Sequnce	T
Gienah γ Crv	Main Sequence	Hot Subdwarf	F
Altair	Main Sequence	Main Sequence	T
Sun	Main Sequence	Main Sequence	T
61 Cygni	Main Sequence	Main Sequence	T
Kruger 60 A	Main Sequence	Main Sequence	T
Lal 21258 B	Main Sequence	Main Sequence	T
Procyon; 61421 A	Normal Giant	Main Sequence	F
Arcturus α Boo	Normal Giant	Normal Giant	T
Kochab β UMi	Normal Giant	Normal Giant	T
Arcturus	Normal Giant	Normal Giant	T
Antares α Sco	Red Giant	Red Giant	T
Shaula θ Sco	Red Giant	Red Giant	T
Deneb α Cyg	Red Giant	Red Giant	T
Enif ε Peg	Red Giant	Red Giant	T
Betelgeuse	Red Giant	Red Giant	T
Polaris	Red Giant	Red Giant	T
Canopus	Red Giant	Red Giant	T

Total number of stars taken for output generation	:	26
Number of misclassification	:	03
Accuracy	:	86.60%

5.1 Remarks

In the manner described above, the Neural Network, classified the stars into its proper class without any error for single hidden layer and in that five nodes. This trained network was further subjected to data from twenty eight stars belonging to all classes (Step 5-vi) at its input and output correspondingly predicted / classified. Only three stars are misclassified namely Gienah γ Cr V, Alphecca α Crβ and Procyon : 61421 A. This yields classification which is 86.6% accurate.

The above three misclassified stars are in the vicinity region of Normal giants, Main Sequence and Hot subdwarfs. We may say that the error is due to the Euclidean distance. For the three misclassified stars, this distance is small between classes as compared to its nearest neighbour star of the same class. This result is comparable to or even better than similar classification techniques like Nearest neighbourhood, k-nearest neighbour, clustering techniques, as Neural Network also can be seen as an improved version of Nearest Neighbourhood classifier technique. This method has been successfully employed by many authors to classify UV spectra (Gulati el al., 1994), some schemes for general classification by Automated approach using ANN (Gulati et al., 1994), star classification using spectral and luminosity classification of stars (Klush 1994). The results in these are very encouraging and in particular in the classification of UV spectra where the classification accuracy of 94% has been achieved.

6. Interval estimation

The learning process experienced by a neural network is stochastic in nature. Here we are attempting to predict (actually classify) some behaviour or a response or a pattern associated with physical data (like H-R diagram) on different confidence intervals. The term "confidence interval" has an intuitive meaning as well as a technical meaning. It is natural to expect it to mean "an interval in which one may be confident that a parameter lies". With out loss of generality we have assumed our target problem is exhibiting normal or Gaussian distribution, with mean μ and standard deviation σ. The Fig. 3a shows the relation between confidence coefficient and confidence interval width.

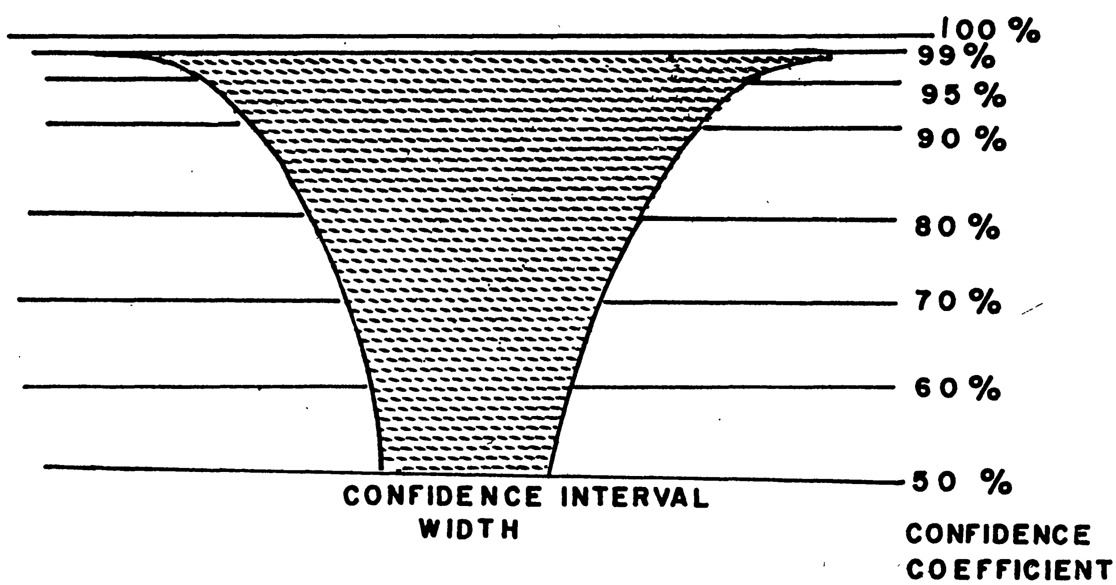

Figure 3a. Relation between confidence coefficient & confidence interval width

Many confidence intervals can be discussed in terms of one-dimensional statistic T(x) a continuous random variable, given probabilities α_1 and α_2 it is possible to find $T_1(\theta)$ and T_2 (θ) such that

$$P_r[T(X) < T_1\ (\theta)\,|\,\theta_1] = \alpha_1 \tag{6.1}$$

$$\&\ P_r[T(X) > T_2\ (\theta)\,|\,\theta_2] = \alpha_2 \tag{6.2}$$

Confidence limits for θ based statistic T is shown in Fig. 3b. For every particular value of θ the probability that T lies between $T_1(\theta) - T_2(\theta)$ is $1\text{-}\alpha_1\text{-}\alpha_2$. The basic idea of confidence intervals is to express confidence $1-\alpha_1-\alpha_2$ that the point (θ,T) lies in the confidence belt after T has been observed.

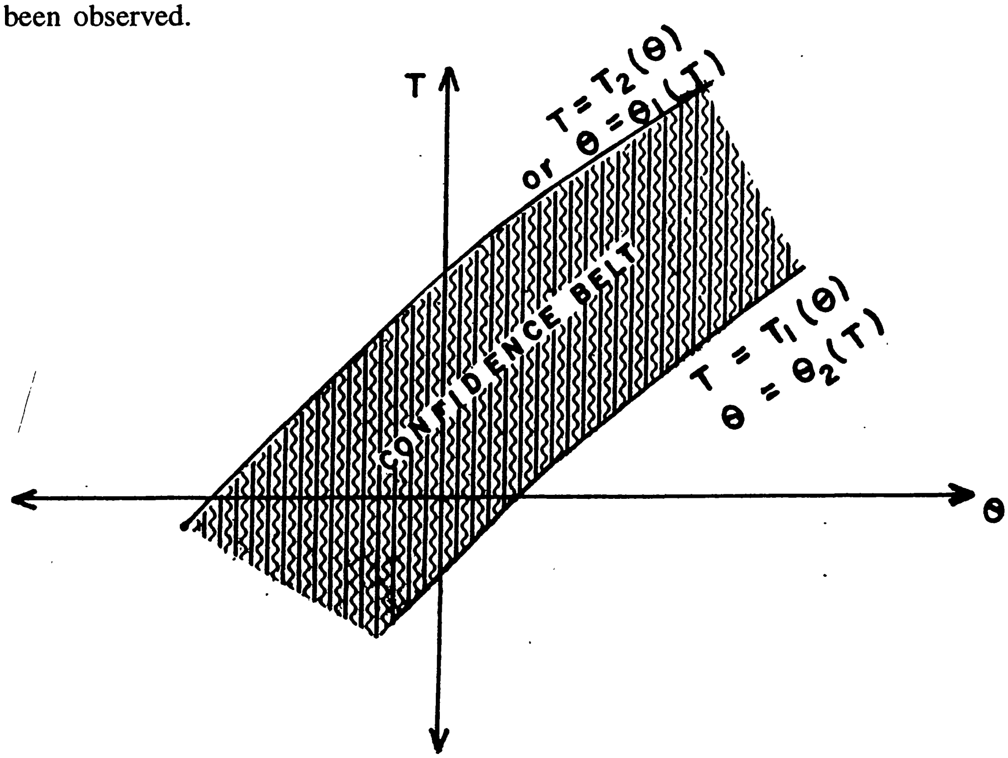

Figure 3b. Confidence limits for q based on the statistic T.

6.1 Computer implementation of confidence interval concept

(6-i) The three input parameters (B-V Color index, Absolute magnitude, and Temperature) for the five classes namely, Hot subdwarfs, White dwarfs, Main Sequence, Normal Giants and Red giants, consisting of fifty stars in each class (Sample data's are shown in Appendix B) are taken. Calculate mean $\bar{x}$ and standard deviation σ for each class, normalize these values between 0 and 1, and then encode between $\bar{x} \pm \sigma/3$ as in Box 1.

(6-ii) Repeat the steps from (iv) to (viii) as discussed in section 5.

(6-iii) The steps from (6-i) to (6-ii) is repeated at different confidence intervals viz., $\bar{x} \pm \sigma/2$, $\bar{x} \pm \sigma$, $\bar{x} \pm 2\sigma$ and $\bar{x} \pm 3\sigma$. Its results are discussed in section 7.

7. Dependence of the Neural Network prediction on an confidence interval, number of iterations and hidden layers

(i) At $\bar{x} \pm \sigma/2$ and $\bar{x} \pm 3\sigma$ confidence interval limits, the normalized system error (NSE) is minimum as shown in Fig. 4a, and the best fit line is Y = – 0.0084X+0.040.

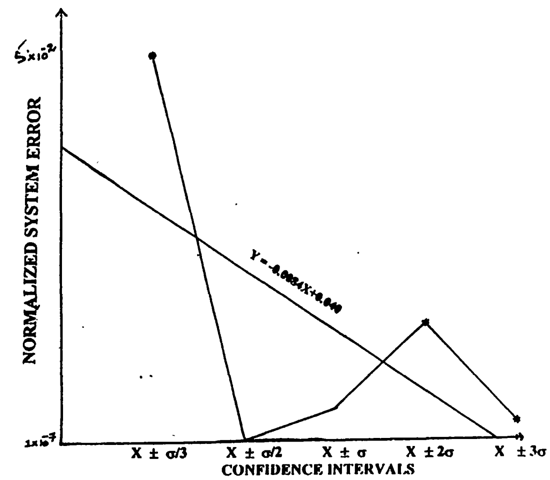

Figure 4a. Normalized System Error versus confidence intervals.

(ii) In all confidence intervals ($\bar{x} \pm \sigma/3$ to $\bar{x} \pm 3\sigma$) the NSE decreases till 10000 iterations and is minimum for $\bar{x} \pm \sigma/2$ limit values and is shown in Fig. 4b.

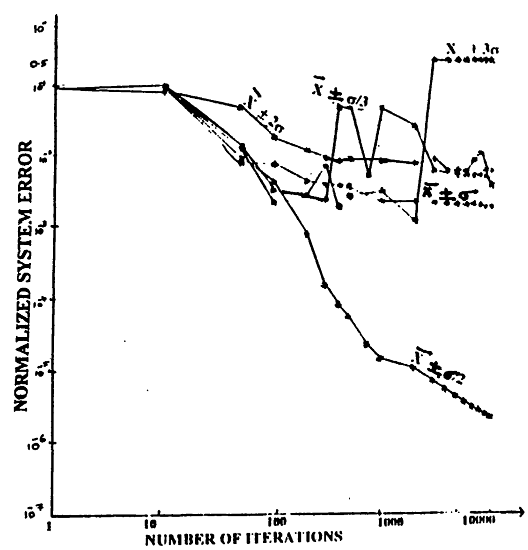

Figure 4b. Normalized System Error versus number of iterations.

(iii) As the number of units in a hidden layer is increased from 1 to 10 the NSE decreases and the best fit line is given by Y = –0.00091X+0.0689 and is shown in Fig. 4c.

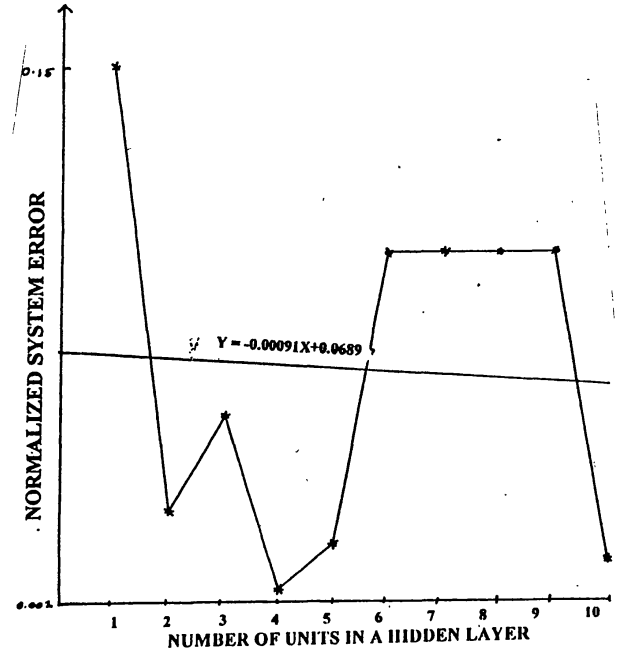

Figure 4c. Normalized System Error versus number of units in a hidden layer.

(iv) As we increase the number of hidden layers the NSE increases, its best fit line is given by Y=0.073X+0.07 and is shown in Fig. 4d.

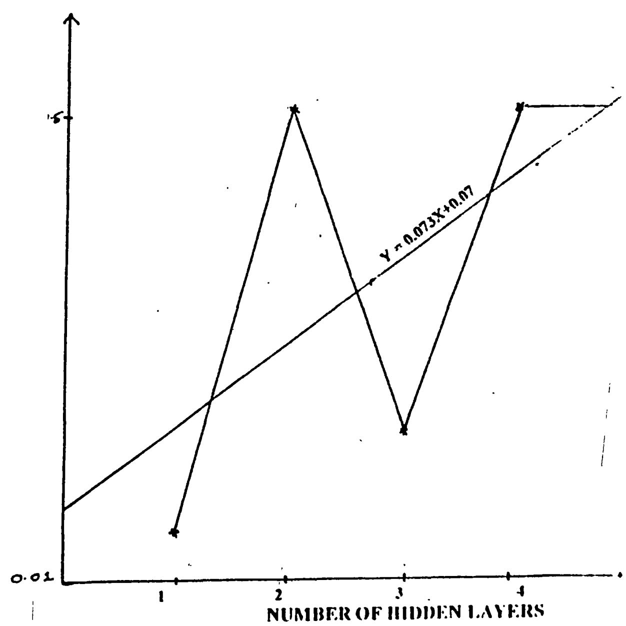

Figure 4d. Normalized System Error versus number of hidden layers.

8. Remarks

There is no exact mathematical proof / law for how many number of hidden layers and nodes in that layer to use (Timothy Masters, 1993). Using too few will starve the network and too many will increase the training time and may cause overfitting. Some follow geometric pyramid rule in which the number of neurons decrease from input to output in a pyramid like fashion. For example refer to Fig. 1c let m,h,n represent the number of input, hidden, and output nodes respectively, then number of hidden nodes in a layer j is $h = (mn)^{1/2}$

(if m=12 units, n=3 units then h=6 units). If there are two hidden layers say h_1 and h_2, then define $r = (n/m)^{1/3}$, $h_1 = mr^2$ and $h_2 = mr$.

We studied the normalised system error as a function of number of hidden layers (1 to 5) and number of units in a hidden layer (1 to 10) and we achieved best performance when number of hidden layer is one and in that layer three nodes are taken. Further we have carried out similar classification and prediction of elements in a periodic table and crystal class and we observed that one hidden layer and three nodes in that layer is sufficient upto eight classification problem. For sixteen class problem we found the normalized system error to be higher.

9. Conclusion

The max-min-max subarray input features are used with Generalized Delta Rule with encouraging results. This is an attempt to show the advancement and capability of a neural networks. Further an important problem like star classification has been shown to be amenable to neural network based classification techniques. It would be interesting to extend this mapping to other aspects of Astrophysics classification problem like Galaxies and other Cosmic structure, classification perhaps using other types of neural networks. The dependency of input parameters on a confidence intervals, number of iterations and number of hidden layers are also studied.

References

Allen C.W., 1974, Astrophysical Quantities, 3rd Ed.

Bernard Widrow, 1990, Proc. of the IEEE, 78 No. 9, 1415.

Chandrashekar S., 1939, An Introduction to the study of stellar structure, "Chicago I11., Chicago Univ. Press.

Chandrashekar S., 1951, The structure, the composition, the source of energy of the Stars, Mc Graw Hill, 598.

Flugge S., 1958, Encyclopedia of Physics, Vol. L1 Astrophysics II : Stellar Structure, Springer Verlag, 75.

Gulati R.K. et al., 1994, ApJ, Vol. 426, 340.

Gulati R.K. et al., 1994, Vistas in Astronomy, Vol. 38, 293.

Jain A. K. et al., March 1996, IEEE Computers, 31.

Johnson H.L. et al., 1953, ApJ, 17, 313.

Klusch M., 1994, Vistas in Astronomy, Vol. 38, 299.

Simon Haykin, 1995, Neural Networks, Mc Millan College Publishing Company, New York.

Timothy Masters, 1993, Practical Neural Network Recipes in C++, Academic Press Inc.

Yi Shang et al., March 1996, IEEE Computer, 45.

Appendix A

Here we suggest a C language implementation for the algorithm.

```
1.  Define constants a used throughout the function

2.  (a) Generate random numbers (Computer independent)
    (b) Initialize weights with random numbers beweeen –0.5 and +0.5

3.  Learning( )
    {
        (a) Read user input data input file during the file iteration session
        (b) Arrange all the input values in descending order as stated in Box 1 from step 1 to
        (c) Specify architecture of net and values of learning parameters
        (d) Allocate dynamic strorage for the net.
        (e) Read file containing weights and thresholds.
        (f) Create file for net architecture and learning parameters.
        (g) Create file for saving weights and thresholds learned from training.
    }

4.  Bottom up calculation( )
    {
        (a) Check several condition to see whether learning should terminate.
        (b) Check System error
    }

5.  Output generation( )
    {
        Create input data for output generation
    }

6.  Update weights ( ) /* Only if computer hangs */
    {
    If the number of iterations are very large then store the weights in a separate file after specified steps. (say 200, 400,
    600,...)
    }

7.  Main ( )
    {
        do
        {
            Scanf ("%f" & select);
            Switch(Select[o])
            Case 'o' : case 'O';
            Output generation( );
            break;
                Case '1' : Case 'L';
                Learning( );
                break;
        }
        while
        {
            (cont[o]=='y' || (cont[O]=='Y'));
        }
    }
```

Box 1. Max-Min-Max Generalised Delta Rule (GDR)

(1) Arrange all the patterns in an array
$[X_{mn}]\ [W] = [B_i]$
where W is the weight matrix, X is the input patterns, B is the number of classes

$$\left[\begin{array}{ccccccc}
x^1_{11} & x^1_{12} & x^1_{13} & \cdot & \cdot & \cdot & x^1_{1N} \\
x^1_{21} & x^1_{22} & x^1_{23} & \cdot & \cdot & \cdot & x^1_{2N} \\
x^1_{31} & x^1_{32} & x^1_{33} & \cdot & \cdot & \cdot & x^1_{3N} \\
\cdot & & \cdot & & & & \\
x^1_{(i-1)1} & x^1_{(i-1)2} & x^1_{(i-1)3} & \cdot & \cdot & \cdot & x^1_{(i-1)N} \\
x^1_{i1} & x^1_{i2} & x^1_{i3} & \cdot & \cdot & \cdot & x^1_{iN} \\
x^1_{(i+1)1} & x^1_{(i+1)2} & x^1_{(i+1)3} & \cdot & \cdot & \cdot & x^1_{(i+1)N} \\
\cdot & & & & & & \\
x^1_{(A-1)1} & x^1_{(A-1)2} & x^1_{(A-1)3} & \cdot & \cdot & \cdot & x^1_{(A-1)N} \\
x^1_{A1} & x^1_{A2} & x^1_{A3} & \cdot & \cdot & \cdot & x^1_{AN} \\
\hdashline
x^2_{11} & x^2_{12} & x^2_{13} & \cdot & \cdot & \cdot & x^2_{1N} \\
x^2_{21} & x^2_{22} & x^2_{23} & \cdot & \cdot & \cdot & x^2_{2N} \\
x^2_{31} & x^2_{32} & x^2_{33} & \cdot & \cdot & \cdot & x^2_{3N} \\
\cdot & & & & & & \\
x^1_{(j-1)1} & x^2_{(j-1)2} & x^2_{(j-1)3} & \cdot & \cdot & \cdot & x^2_{(j-1)N} \\
x^2_{j1} & x^2_{j2} & x^2_{j3} & \cdot & \cdot & \cdot & x^2_{jN} \\
x^2_{(j+1)1} & x^2_{(j+1)2} & x^2_{(j+1)3} & \cdot & \cdot & \cdot & x^2_{(j+1)N} \\
\cdot & & & & & & \\
x^2_{(B-1)1} & x^2_{(B-1)2} & x^2_{(B-1)3} & \cdot & \cdot & \cdot & x^2_{(B-1)N} \\
\hdashline
x^2_{B1} & x^2_{B2} & x^2_{B3} & \cdot & \cdot & \cdot & x^2_{BN} \\
\cdot & & & & & & \\
\cdot & & & & & & \\
\cdot & & & & & & \\
x^n_{Z1} & x^n_{Z2} & x^n_{Z3} & \cdot & \cdot & \cdot & x^n_{ZN}
\end{array}\right]
\left[\begin{array}{c} \\ \\ W \\ \\ \end{array}\right]
=
\left[\begin{array}{c} 1^{ST}\ \text{Class} \\ \hdashline 2^{nd}\ \text{Class} \\ \hdashline \\ \end{array}\right]$$

Note :

(a) $(Rx_{1N}-Rx_{2N})>(Rx_{2N}-Rx_{3N})\ldots>(Rx_{(i-1)N}-Rx_{iN})=(Rx_{(1+2)N}-Rx_{(i+1)N})<(Rx_{(i+3)}-Rx_{(i+2)})<\ldots<(Rx_{AN}-Rx_{(A-1)N})$
where Rx1N means Fow x_{1N}. Similarly we can do for second, third, … n^{th} classes.

(b) A + B + … + Z = M

(2) Calculate mean ($\bar{x}$) and standard deviation (σ) for each class input pattern paratmers.

(3) Encode the values between $\bar{x}+3\sigma$ and $\bar{x}-3\sigma$ limits.

(4) Keep the highest magnitude input feature column into the left most.

(5) Present an input and specify the desired output.

(6) If there are four classes declare output classes as 00, 01, 10 & 11 Don't tollow Gray order.

(7) Initialize all the weights to small random values (–0.5 to 0.5)

(8) Compute the actual output of the network Y_j

$$Y_j = \frac{1}{1.005+e^{-(\text{Net X} + \theta j)/\theta_0}}$$

Where Net X = $\Sigma\, W_j\, iX_i$, θ_j = threshold or bias, θ_o : will modify the shape of the Sigmoid function. The non linear function used is the Sigmoid.

(9) Calculate the error between the actual output Y_j & the desired output d_j
The error term is
$\delta_k = O_k\,(1-O_k)\,(d_k - O_k)$ where k is output layer
$\delta_j = O_j(1-O_j)\,\Sigma\,\delta_k\,W_{kj}$, where j is the hidden layer above the node j.

(10) Adjust the weights by
$\Delta\, W_{ji}(n+1) = \alpha\,\Delta\,W_{ji}\,(n) + \eta\,(\delta_j\,Y_i)$
where η is the learning parameter (0.70), α : is the momentum coefficient (default = 0.90)

(11) Repeat steps 7 through 10 for each training pair until the weights and error settle down.

Appendix: B: Data for Classification

White Dwarfs			*Hot Subdwarfs*			*Main sequence*			*Normal giants*			*Red giants*		
B-V color index	Abs. Magn-itude	Temp. K	B-V color index	Abs. Magn-itude	Temp. K	B-V color index	Abs. Magn-itude	Temp. K	B-V color index	Abs. Magn-itude	Temp. K	B-V color index	Abs. Magn-itude	Temp. K
-0.20	10.00	11000	-0.30	6.30	13000	-0.40	-5.80	23000	2.00	-0.40	1000	2.00	-6.00	1500
-0.20	11.00	12000	-0.43	5.70	27000	-0.37	-4.90	19000	2.00	-1.00	1500	1.90	-6.70	1600
-0.35	10.40	12000	-0.40	5.00	23000	-0.34	-4.00	16000	2.00	-1.50	1500	1.90	-6.00	1600
-0.25	11.20	11500	-0.40	4.40	23000	-0.32	-3.50	13500	1.95	-0.20	1550	1.80	-6.00	1750
-0.25	10.50	11500	-0.40	4.00	23000	-0.29	-2.90	12800	1.90	-0.10	1600	1.80	-6.50	1750
-0.10	11.40	10000	-0.43	3.50	27000	-0.27	-2.50	12000	1.90	-0.60	1600	1.80	-7.00	1750
-0.10	10.80	10000	-0.44	3.00	28000	-0.24	-2.00	11500	1.90	-1.20	1600	1.70	-6.00	2000
-0.10	10.20	10000	-0.43	2.50	27000	-0.20	-1.50	11000	1.85	-0.50	1700	1.90	-6.50	1600
-0.05	11.00	9000	-0.40	2.00	23000	-0.15	-0.90	10500	1.80	-0.75	1750	1.70	-7.00	2000
0.00	12.00	8000	-0.42	1.50	25000	-0.11	-0.50	10100	1.80	-0.50	1750	1.70	-7.50	2000
0.00	11.00	8000	-0.42	1.00	25000	-0.08	0.00	9000	1.80	-1.00	1750	1.60	-6.20	2200
0.00	10.50	8000	-0.40	0.00	23000	-0.04	0.50	8800	1.70	0.00	2000	1.60	-6.50	2200
0.05	12.20	8300	-0.35	-0.50	14000	0.00	1.00	8500	1.70	-0.50	2000	1.60	-7.00	2200
0.05	10.60	8300	-0.30	-0.80	10000	0.10	1.50	8400	1.70	1.20	2000	1.60	-7.80	2200
0.10	12.40	8000	-0.35	-0.40	14000	0.10	2.00	8000	1.70	-0.20	2000	1.55	-7.00	2500
0.10	11.70	8000	-0.20	0.30	11000	0.15	2.40	7800	1.70	-0.60	2000	1.50	-6.20	2800
0.10	11.00	8000	-0.25	1.00	13500	0.20	2.80	7400	1.65	0.20	2100	1.50	-6.90	2800
0.15	12.00	7500	-0.10	-1.50	10000	0.25	3.10	7000	1.60	0.00	2100	1.50	-7.90	2800
0.20	13.00	7400	-0.05	2.50	9000	0.30	2.40	6800	1.60	0.30	2100	1.45	-7.00	2900
0.20	12.20	7400	-0.30	2.10	13000	0.35	3.80	6500	1.60	-0.50	2100	1.45	-7.00	2900
0.20	11.50	7400	-0.15	4.00	10500	0.40	3.90	6300	1.55	0.30	2500	1.45	-7.50	2900
0.25	12.50	7000	-0.18	4.50	10800	0.45	4.20	6000	1.55	0.00	2500	1.40	-6.20	3100
0.25	11.80	7000	-0.20	5.20	11000	0.50	4.40	5800	1.50	0.40	2800	1.40	-6.50	3100
0.30	13.50	6800	-0.25	5.80	12500	5.50	4.60	5600	1.50	0.00	2800	1.40	-7.00	3100
0.30	12.50	6800	-0.30	5.40	13000	0.60	5.00	5500	1.50	-0.30	2800	1.40	-7.50	3100
0.30	12.00	6800	-0.25	5.00	12000	0.65	0.52	5300	1.35	0.50	3300	1.40	-7.80	3100
0.35	13.00	6500	-0.25	5.00	12000	0.70	5.50	5000	1.30	0.00	3600	1.35	-7.00	3300
0.40	14.00	6300	-0.35	4.50	14000	0.75	5.80	4900	1.30	0.50	3600	1.30	-6.40	3600
0.40	13.00	6300	-0.25	4.50	12500	0.80	6.00	4800	1.30	0.90	3600	1.30	-7.00	3600
0.40	12.50	6300	-0.30	4.00	13000	0.86	6.20	4600	1.25	0.50	3700	1.30	-7.50	3600
0.45	13.50	6000	-0.20	4.00	10000	0.90	6.60	4500	1.20	0.50	3800	1.30	-7.90	3600
0.50	14.30	5800	-0.35	3.60	14000	1.00	7.00	4200	1.15	1.00	3900	1.35	-7.00	3300
0.50	13.50	5800	-0.25	3.50	12500	1.09	7.20	4000	1.00	1.11	4200	1.30	-6.40	3600
0.50	13.00	5800	-0.15	3.40	10500	1.10	7.40	4000	1.10	1.16	4200	1.30	-7.00	3600
0.60	14.80	5500	-0.35	3.00	14000	1.20	7.80	3800	1.15	1.00	3900	1.30	-7.50	3600
0.60	14.00	5500	-0.30	3.00	13000	1.30	8.40	3600	1.15	1.00	3900	1.30	-8.00	3600
0.60	13.40	5500	-0.20	3.00	11000	1.40	8.90	3100	1.15	1.50	3900	1.25	-7.00	3700
0.70	15.20	5100	-0.35	2.50	14000	1.45	9.20	2900	1.12	2.30	4000	1.20	-6.70	3800
0.70	14.50	5100	-0.25	2.50	12500	1.51	10.00	2800	1.00	0.60	4200	1.30	-6.40	3600
0.70	14.00	5100	-0.15	2.50	10500	1.55	10.50	2800	1.00	1.00	4200	1.30	-7.00	3600
0.80	15.80	4800	-0.30	2.00	13000	1.60	11.20	2200	1.00	1.50	4200	1.30	-7.50	3600
0.80	15.00	4800	-0.20	2.00	11000	1.66	12.00	2100	1.00	2.00	4200	1.15	-6.50	3900
0.80	15.00	4800	-0.10	2.00	10000	1.69	12.40	2000	0.96	1.86	4300	1.10	-6.20	4000
0.80	14.40	4800	-0.15	1.50	10500	1.74	12.80	1600	0.93	1.20	4350	1.10	-6.90	4000
0.90	16.00	4500	-0.25	1.50	12000	1.67	13.40	2200	1.35	0.55	3350	1.10	-7.40	4000
0.90	15.50	4500	-0.35	1.50	14000	1.80	13.80	1750	1.70	0.10	2000	1.10	-7.90	4000
0.90	15.00	4500	-0.20	1.00	11000	1.85	14.50	1700	2.05	-1.00	1500	1.05	-6.40	4100
0.95	15.50	4400	-0.25	0.50	12500	1.94	15.60	1500	1.42	0.70	3000	1.00	-6.90	4200
1.00	16.00	4200	-0.30	1.00	13000	1.90	15.00	1600	1.76	0.20	2300	1.05	-7.70	4000
1.00	15.40	4200	-0.25	0.50	12500	1.94	15.60	1550	1.42	0.70	3100	1.00	-6.90	4200

Appendix C: Input values for the Neural Network

White Dwarfs				Hot Subdwarfs				Main sequence				Normal Giants				Red Giants			
B-V Colour Index	Abs. Magn-itude	Temp. K	Output	B-V colour Index	Abs. Magn-itude	Temp. K	Output	B-V colour index	Abs. Magn-itude	Temp. K	Output	B-V colour index	Abs. Magn-itude	Temp. K	Output	B-V colour index	Abs. Magn-itude	Temp. K	Output
0.730	0.680	0.610	0 0 0	0.720	0.643	0.630	0 0 1	0.950	0.576	0.515	0 1 1	0.950	0.520	0.515	0 1 0	0.910	0.522	0.730	1 1 0
0.730	0.680	0.600	0 0 0	0.707	0.637	0.770	0 0 1	0.950	0.570	0.515	0 1 1	0.940	0.513	0.515	0 1 0	0.983	0.531	0.730	1 1 0
0.715	0.685	0.620	0 0 0	0.710	0.630	0.730	0 0 1	0.950	0.576	0.515	0 1 1	0.930	0.513	0.517	0 1 0	0.916	0.540	0.730	1 1 0
0.725	0.692	0.615	0 0 0	0.710	0.624	0.730	0 0 1	0.945	0.570	0.515	0 1 1	0.930	0.510	0.517	0 1 0	0.918	0.545	0.730	1 1 0
0.725	0.685	0.615	0 0 0	0.710	0.620	0.730	0 0 1	0.940	0.565	0.516	0 1 1	0.930	0.520	0.520	0 1 0	0.921	0.561	0.730	1 1 0
0.740	0.594	0.600	0 0 0	0.707	0.615	0.770	0 0 1	0.940	0.578	0.515	0 1 1	0.940	0.515	0.515	0 1 0	0.923	0.562	0.730	1 1 0
0.740	0.682	0.600	0 0 0	0.706	0.610	0.780	0 0 1	0.935	0.579	0.515	0 1 1	0.920	0.510	0.520	0 1 0	0.926	0.560	0.730	1 1 0
0.745	0.690	0.690	0 0 0	0.707	0.605	0.770	0 0 1	0.930	0.574	0.515	0 1 1	0.920	0.505	0.520	0 1 0	0.930	0.565	0.730	1 1 0
0.750	0.700	0.580	0 0 0	0.710	0.620	0.730	0 0 1	0.930	0.558	0.515	0 1 1	0.910	0.518	0.520	0 1 0	0.935	0.571	0.730	1 1 0
0.750	0.690	0.580	0 0 0	0.708	0.595	0.750	0 0 1	0.930	0.575	0.515	0 1 1	0.920	0.515	0.522	0 1 0	0.939	0.575	0.730	1 1 0
0.750	0.685	0.580	0 0 0	0.708	0.590	0.750	0 0 1	0.920	0.580	0.515	0 1 1	0.910	0.510	0.522	0 1 0	0.942	0.580	0.730	1 1 0
0.755	0.702	0.583	0 0 0	0.710	0.580	0.730	0 0 1	0.920	0.575	0.516	0 1 1	0.910	0.502	0.522	0 1 0	0.946	0.585	0.730	1 1 0
0.755	0.695	0.583	0 0 0	0.715	0.585	0.640	0 0 1	0.920	0.570	0.516	0 1 1	0.905	0.510	0.525	0 1 0	0.950	0.590	0.730	1 1 0
0.755	0.686	0.583	0 0 0	0.720	0.572	0.600	0 0 1	0.920	0.575	0.517	0 1 1	0.905	0.510	0.525	0 1 0	0.955	0.595	0.730	1 1 0
0.760	0.704	0.580	0 0 0	0.715	0.576	0.640	0 0 1	0.920	0.592	0.517	0 1 1	0.900	0.518	0.528	0 1 0	0.960	0.600	0.730	1 1 0
0.760	0.697	0.580	0 0 0	0.720	0.577	0.610	0 0 1	0.915	0.578	0.517	0 1 1	0.900	0.511	0.528	0 1 0	0.965	0.604	0.730	1 1 0
0.760	0.690	0.580	0 0 0	0.725	0.590	0.600	0 0 1	0.910	0.574	0.517	0 1 1	0.900	0.506	0.528	0 1 0	0.970	0.608	0.730	1 1 0
0.765	0.700	0.575	0 0 0	0.740	0.565	0.590	0 0 1	0.910	0.582	0.515	0 1 1	0.900	0.501	0.528	0 1 0	0.975	0.611	0.730	1 1 0
0.770	0.710	0.574	0 0 0	0.745	0.605	0.630	0 0 1	0.910	0.580	0.516	0 1 1	0.755	0.510	0.529	0 1 0	0.950	0.604	0.730	1 1 0
0.770	0.720	0.574	0 0 0	0.720	0.601	0.605	0 0 1	0.905	0.583	0.520	0 1 1	0.895	0.505	0.529	0 1 0	0.985	0.618	0.730	1 1 0
0.770	0.725	0.574	0 0 0	0.735	0.620	0.608	0 0 1	0.995	0.575	0.520	0 1 1	0.895	0.518	0.531	0 1 0	0.990	0.619	0.730	1 1 0
0.775	0.705	0.570	0 0 0	0.770	0.632	0.610	0 0 1	0.900	0.583	0.520	0 1 1	0.890	0.515	0.531	0 1 0	0.885	0.622	0.730	1 1 0
0.775	0.698	0.570	0 0 0	0.725	0.634	0.625	0 0 1	0.900	0.580	0.521	0 1 1	0.890	0.510	0.531	0 1 0	0.900	0.622	0.730	1 1 0
0.780	0.715	0.568	0 0 0	0.720	0.630	0.630	0 0 1	0.900	0.580	0.521	0 1 1	0.890	0.505	0.531	0 1 0	0.805	0.626	0.730	1 1 0
0.780	0.705	0.568	0 0 0	0.725	0.625	0.620	0 0 1	0.990	0.580	0.521	0 1 1	0.891	0.505	0.531	0 1 0	0.810	0.630	0.730	1 1 0
0.760	0.700	0.568	0 0 0	0.725	0.625	0.625	0 0 1	0.990	0.577	0.521	0 1 1	0.891	0.510	0.533	0 1 0	0.815	0.585	0.730	1 1 0
0.785	0.710	0.565	0 0 0	0.715	0.620	0.640	0 0 1	0.890	0.580	0.525	0 1 1	0.885	0.516	0.536	0 1 0	0.812	0.635	0.730	1 1 0
0.790	0.720	0.563	0 0 0	0.725	0.620	0.625	0 0 1	0.890	0.582	0.525	0 1 1	0.880	0.500	0.536	0 1 0	0.825	0.638	0.730	1 1 0
0.790	0.710	0.563	0 0 0	0.720	0.616	0.630	0 0 1	0.885	0.578	0.528	0 1 1	0.880	0.501	0.536	0 1 0	0.830	0.640	0.730	1 1 0
0.750	0.705	0.563	0 0 0	0.730	0.615	0.610	0 0 1	0.880	0.578	0.528	0 1 1	0.880	0.509	0.536	0 1 0	0.838	0.642	0.730	1 1 0
0.795	0.715	0.560	0 0 0	0.715	0.614	0.640	0 0 1	0.880	0.585	0.528	0 1 1	0.880	0.516	0.537	0 1 0	0.840	0.646	0.730	1 1 0
0.800	0.723	0.558	0 0 0	0.725	0.610	0.625	0 0 1	0.880	0.580	0.531	0 1 1	0.880	0.510	0.536	0 1 0	0.850	0.650	0.730	1 1 0
0.800	0.723	0.558	0 0 0	0.735	0.610	0.605	0 0 1	0.875	0.585	0.531	0 1 1	0.880	0.505	0.536	0 1 0	0.859	0.652	0.730	1 1 0
0.800	0.715	0.558	0 0 0	0.715	0.610	0.640	0 0 1	0.950	0.589	0.531	0 1 1	0.881	0.502	0.536	0 1 0	0.860	0.654	0.730	1 1 0
0.800	0.710	0.558	0 0 0	0.720	0.610	0.630	0 0 1	0.940	0.585	0.531	0 1 1	0.880	0.501	0.536	0 1 0	0.870	0.658	0.730	1 1 0
0.810	0.728	0.558	0 0 0	0.730	0.610	0.610	0 0 1	0.930	0.520	0.533	0 1 1	0.880	0.517	0.537	0 1 0	0.880	0.668	0.730	1 1 0
0.810	0.720	0.555	0 0 0	0.715	0.605	0.640	0 0 1	0.930	0.513	0.536	0 1 1	0.875	0.510	0.538	0 1 0	0.890	0.669	0.730	1 1 0
0.810	0.714	0.555	0 0 0	0.725	0.605	0.625	0 0 1	0.930	0.513	0.536	0 1 1	0.870	0.505	0.538	0 1 0	0.895	0.672	0.730	1 1 0
0.820	0.732	0.554	0 0 0	0.735	0.605	0.605	0 0 1	0.940	0.510	0.553	0 1 1	0.870	0.501	0.538	0 1 0	0.901	0.680	0.730	1 1 0
0.720	0.725	0.551	0 0 0	0.720	0.605	0.630	0 0 1	0.920	0.520	0.525	0 1 1	0.870	0.515	0.538	0 1 0	0.905	0.685	0.730	1 1 0
0.820	0.720	0.551	0 0 0	0.730	0.600	0.610	0 0 1	0.920	0.515	0.520	0 1 1	0.870	0.505	0.539	0 1 0	0.910	0.692	0.730	1 1 0
0.830	0.738	0.548	0 0 0	0.740	0.600	0.600	0 0 1	0.910	0.515	0.534	0 1 1	0.865	0.518	0.540	0 1 0	0.916	0.700	0.730	1 1 0
0.830	0.730	0.548	0 0 0	0.735	0.595	0.605	0 0 1	0.920	0.501	0.535	0 1 1	0.865	0.511	0.540	0 1 0	0.919	0.704	0.730	1 1 0
0.830	0.730	0.548	0 0 0	0.725	0.595	0.625	0 0 1	0.910	0.523	0.551	0 1 1	0.860	0.506	0.541	0 1 0	0.994	0.708	0.730	1 1 0
0.830	0.724	0.548	0 0 0	0.740	0.595	0.600	0 0 1	0.910	0.512	0.545	0 1 1	0.860	0.501	0.541	0 1 0	0.917	0.714	0.730	1 1 0
0.880	0.900	0.545	0 0 0	0.730	0.590	0.610	0 0 1	0.905	0.534	0.546	0 1 1	0.860	0.502	0.542	0 1 0	0.930	0.718	0.730	1 1 0
0.840	0.735	0.545	0 0 0	0.720	0.590	0.630	0 0 1	0.905	0.546	0.549	0 1 1	0.860	0.503	0.730	0 1 0	0.935	0.725	0.730	1 1 0
0.840	0.735	0.545	0 0 0	0.720	0.590	0.630	0 0 1	0.905	0.546	0.549	0 1 1	0.860	0.503	0.730	0 1 0	0.935	0.725	0.730	1 1 0
0.850	0.740	0.542	0 0 0	0.725	0.585	0.620	0 0 1	0.910	0.516	0.537	0 1 1	0.855	0.520	0.690	0 1 0	0.940	0.730	0.730	1 1 0
0.850	0.734	0.542	0 0 0	0.775	0.585	0.640	0 0 1	0.903	0.506	0.536	0 1 1	0.855	0.520	0.660	0 1 0	0.944	0.736	0.730	1 1 0

Note: (1) B-V colour index : B-V Colour index

(2) Abs. Magn-itude : Absolute magnitude

(3) Temp. K : Temperature in K

Appendix D:

Data values of the twenty eight stars belonging to all classes for Neural network validation.

B-V Colour Index	Absolute Magnitude M_v	Absolute Temperature K	STAR	ACTUAL CLASS
-1.02	0.40	10000	Algol β Per	Hot Subdwarfs
0.00	1.40	9500	Merak β UMa	Hot Subdwarfs
-1.85	0.65	10500	Sirius A	Hot Subdwarfs
0.52	14.00	6100	V. Maanen; Wolf 28	White Dwarf
-0.85	10.80	10000	Eri; 26965B	White Dwarf
0.44	11.20	6800	Ac $^+$ 58 25002B	White Dwarf
1.80	12.35	7800	L145-141	White Dwarf
0.10	11.10	8500	Sirius B	White Dwarf
-0.90	-1.80	10500	Alpheratz α And	Main Sequence
0.10	0.10	8500	Alphecca α Cr β	Main Sequence
-0.50	-3.80	32000	Spica	Main Sequence
-0.25	-1.80	11000	Gienah γ Crv	Main Sequence
0.24	2.20	7000	Altair	Main Sequence
0.52	3.90	5900	Sun	Main Sequence
1.14	6.40	4000	61 Cygni	Main Sequence
1.42	10.90	2950	Kruger 60 A	Main Sequence
1.63	12.60	2000	Lal 2125B	Main Sequence
5.80	2.00	5900	Procyon; 61421 A	Normal Giant
1.56	-0..80	2600	Arcturus α Boo	Normal Giant
1.16	-0.10	3900	Kochab β UMi	Normal Giant
0.90	-1.00	4600	Arcturus	Normal Giant
0.80	-0.80	4800	Arcturus	Red Giant
0.43	-5.90	6300	Shaula θ Sco	Red Giant
0.12	-7.90	7800	Deneb α Cyg	Red Giant
1.52	-5.00	2700	Enif ε Peg	Red Giant
1.40	-5.20	2950	Betelgeuse	Red Giant
0.43	-2.60	6300	Polaris	Red Giant
0.28	-5.90	7200	Canopus	Red Giant