

Analysis of Speed, Accuracy of Deep Learning using Gini index, HSM Based Fuzzy Decision Trees

S.V.G. Reddy^{1*}, K. Thammi Reddy², V. Valli Kumari³

^{1*} Dept. of CSE, GIT (GITAM University), Visakhapatna, India

² Dept. of CSE, GIT (GITAM University), Visakhapatna, India

³ Dept. of CS and SE, College of Engineering (Andhra University), Visakhapatnam, India

*Corresponding Author: venkat157.reddy@gmail.com, Tel : +91- 996 333 2363

Available online at: www.ijcseonline.org

Received: 02/Oct/2017, Revised: 20/Oct/2017, Accepted: 11/Nov/2017, Published: 30/Nov/2017

Abstract – Deep Learning has gained tremendous importance due to its advancement in various fields of text mining, speech recognition, computer vision, natural language processing etc. The weights of the input layer attributes and the series of hidden layers of deep learning plays a dominant role in its classification speed and accuracy. The weight adjustment algorithm for the Deep Learning is proposed in this paper. Generally, the weights can be determined by mathematical techniques, can be suggested by the domain experts or by considering random weights. In this proposed work, the weights of a neural network are computed mathematically by constructing the fuzzy decision tree. It is proposed to use the maximum heterogeneous split measure(HSM) value of the attribute of the fuzzy decision tree as the weight of the corresponding attribute for the weight adjustment algorithm to classify using neural networks. Fast classification and accuracy is achieved with the computed HSM weights of the deep learning which outperforms when compared with the fuzzy decision tree classifiers. The same work was carried out using the least value of gini index. And in this paper the classification speed, accuracy is compared by considering the gini index and HSM based fuzzy decision trees and analyzed the results.

Keywords – Deep Learning, Heterogeneous split measure, gini index, weight, fuzzy, Decision trees, Classification Accuracy

I. INTRODUCTION

Classification is a very useful and powerful technique with which the hidden knowledge patterns can be extracted from data. There are standard ID3, C4.5 algorithms for classification purpose which uses Entropy as a splitting criterion, but the SLIQ algorithm which is applied here uses HSM as split measure. SLIQ is a decision tree classifier which can deal with both the numeric and categorical attributes. It uses a pre-sorting technique and enables to scale for large data sets irrespective of number of classes, attributes and records thus making it more significant in the data classification. There is a decision tree classifier CLOUDS[1] which creates the splitting points for the numeric attributes.

Crisp decision tree algorithms almost faces the trouble of arriving at sharp decision boundaries which can be rarely seen in the real life classification problems and hence the fuzzy decision trees which are more efficient are used in this paper. The HSM is used as the best split measure for the fuzzy decision trees. The problem with the fuzzy decision trees is, appropriate membership function cannot be identified. In fact, the previous studies / techniques proves that the fuzzy decision trees contains gradual transitions

between attribute values when compared with crisp decision trees. Generally the attributes of the data set is converted into fuzzy values using a triangular or trapezoidal membership function. In this proposed work, the fuzzy values are computed for the split values of an attribute during decision tree construction.

One of the approach to build a decision tree is by using the parameter called HSM[2]. HSM is calculated for all the attributes at various split points and the attribute having maximum value of HSM is decided as the ROOT which is considered as the Best classifier attribute. So, lot of PRIORITY & WEIGHTAGE is given for the Root attribute for classifying the records. That means the maximum value of HSM of an attribute tells that the records of that attribute are well distributed and would be classified with more accuracy and Hence that attribute would be decided as the ROOT of the decision tree.

On the other hand, Deep Learning is a type of Artificial neural network which contains more than one hidden layer and learns to perform the classification tasks directly from images, text, sound. The weight of an attribute of Deep Learning model can be computed using few mathematical

techniques or can be suggested by the domain experts or simply using the Random weights. The proper assigning of weights of neural network leads to speedy computations and achieve more classification accuracy. In this work, it is proposed to assign the weight of attribute of neural network model mathematically by constructing fuzzy decision tree. Then the technique of applying the maximum value of HSM value[3][4][5] of the attribute as the weight of the corresponding attribute to classify the same data set using Neural Networks is proposed in this model. Here, the proposed novel approach aims to fuzzify the decision boundary at each node of the decision tree and build an efficient neural networks model with proposed HSM weights to achieve better classification accuracy. The proposed HSM weights are considered and applied on various types of neural networks such as Deep Learning, Backpropagation, Multi Layer Feed Forward and good results are observed in all the cases.

The rest of the paper is structured in the following way. Section 2, SLIQ & HSM-FDT algorithms which calculates the split values, fuzzy values & HSM values of all the attributes and section 3 is fuzzy decision tree construction. And section 4 - Proposed methodology which illustrates the usage of HSM values for the nodes of fuzzy decision tree as the weights and narrates the classification process using neural networks. Section 5 stresses on Implementations using decision tree and different types of neural networks such as Deep Learning, Backpropagation, Multi Layer Feed Forward and compare the classification accuracy by giving various types of inputs, Section 6 – Results, section 7 compares and Analyze the results of gini index and HSM based weights and their classification speed and accuracy and section 8 is the conclusion.

II. SLIQ & HSM-FDT ALGORITHM

In this approach, fuzzified decision tree would be constructed with HSM as best split measure. So, the concept of split point, fuzzy value and HSM would be explained here. In this proposed work, the Wisconsin data set is used which contains 699 tuples. The data set consists of id, 9 attributes and a class label. There are some missing values and the preprocessing is done to obtain the complete data. The 3-fold cross validation is performed on the data set and three pairs of training and testing sets were prepared. For easy understanding, a sample data of 20 records is taken which contains attributes a1, a2, a3, class label (refer table 1).

Every attribute may contain several split points and the HSM is computed for all the attributes at all the split points. Firstly, to compute the split point, attribute, class label from the sample data is taken. And the attribute is sorted in

ascending order, then due to sorting of the attribute, class label records would also be altered correspondingly. There are only two class labels 1 & 2 in the data set. Then after sorting, the class label is verified from top to bottom in each attribute list. If there is a change observed in the class label from “1 to 2” or “2 to 1”, then the corresponding attribute values related to class label 1 and class label 2(or class label 2 & class label 1) are taken and average them and their average value would be preserved as split point respectively.

The split points would be computed for all the attributes (refer table 2). let's consider attribute a2 which is computed in the following way. Column a2 is sorted, and the corresponding class label have altered. And the change in class label from “1 to 2” or “2 to 1” is verified and we can notice two split points at (58,66) and (66,68). Randomly, let's calculate the split point of attribute a2 at (66,68).

Here the split point would be average of 66, 68 which comes to 67. And the membership value for each record μ by default is taken as $1/c$ (c is the number of class labels used which are 2) which comes to 0.5. Then the standard deviation is calculated for the attribute a2 which comes to 3.451087.

Crisp decision tree algorithms almost faces the trouble of arriving at sharp decision boundaries and to overcome those problems, the fuzzification[6][7] of decision boundary at each node of the decision tree is proposed to provide gradual transitions between attribute values. For the set of records above the split point is treated as top partition, the set of records below the split point is treated as bottom partition and the fuzzy value is computed for all the records of both top, bottom partitions.

Fuzzy value (top partition) = $1 - 1/(1 + \exp(-(\sigma) * (a2 - \text{split point})))$
 Fuzzy value (bottom partition) = $1/(1 + \exp(-(\sigma) * (a2 - \text{split point})))$

Table 1 – sample data set

	A	B	C	D	E
1	sno	a1	a2	a3	class
2	1	38	69	21	2
3	2	42	69	1	2
4	3	43	58	52	2
5	4	44	58	9	2
6	5	46	69	3	2
7	6	46	58	2	2
8	7	47	66	12	1
9	8	48	66	0	1
10	9	49	66	0	1
11	10	50	66	1	1
12	11	51	66	1	1
13	12	52	69	3	2
14	13	54	66	0	1
15	14	54	68	7	2
16	15	55	66	0	1
17	16	55	66	18	1
18	17	55	68	15	2
19	18	56	66	1	1
20	19	56	66	2	1
21	20	61	68	1	2

```
// The attributes sno, a1,a2,...a9, class, m –
number of attributes, n – number of records, sp –
split point
```

```
Function Split()
{
for( I = 1 to m )
{
sort a i
for(j = 1 to n )
{
if ( ( class label [ j ] ==1 && class label [ j+1 ]
==2) ||( class label [ j ] ==2 && class label [ j+1 ]
==1))
sp [ j ] = (aj + a(j+1) ) / 2
} } }
```

Now, let's compute the fuzzy values for attribute a2 (refer table2). Here, the records from 58 to 66 of attribute a2 would be treated as top partition and 68 to last 69 as bottom partition. It means, x1 is computed for attribute a2 from 58 to 66 and other records it is taken as zero value. And x2 is computed for attribute a2 from 68 to last 69 and other records it is taken as zero value. And $x1*\mu$, $x2*\mu$ are computed in the similar manner. The final fuzzy value is computed by merging both $x1*\mu$, $x2*\mu$. Lastly, the sorting is done to get the records from 1 to the end of training data set.

Table 2 – Data set, split points & fuzzy values

	A	B	C	D	E	F	G	H	I	J	K	L
1	sno	a1	a2	a3	class	μ	σ	x1	x2	$x1*\mu$	$x2*\mu$	f
2	3	43	58	52	2	0.5	3.451087	1	0	0.5	0	0.5
3	4	44	58	9	2	0.5	3.451087	1	0	0.5	0	0.5
4	6	46	58	2	2	0.5	3.451087	1	0	0.5	0	0.5
5	7	47	66	12	1	0.5	3.451087	0.9692635	0	0.48463177	0	0.4846318
6	8	48	66	0	1	0.5	3.451087	0.9692635	0	0.48463177	0	0.4846318
7	9	49	66	0	1	0.5	3.451087	0.9692635	0	0.48463177	0	0.4846318
8	10	50	66	1	1	0.5	3.451087	0.9692635	0	0.48463177	0	0.4846318
9	11	51	66	1	1	0.5	3.451087	0.9692635	0	0.48463177	0	0.4846318
10	13	54	66	0	1	0.5	3.451087	0.9692635	0	0.48463177	0	0.4846318
11	15	55	66	0	1	0.5	3.451087	0.9692635	0	0.48463177	0	0.4846318
12	16	55	66	18	1	0.5	3.451087	0.9692635	0	0.48463177	0	0.4846318
13	18	56	66	1	1	0.5	3.451087	0.9692635	0	0.48463177	0	0.4846318
14	19	56	66	2	1	0.5	3.451087	0.9692635	0	0.48463177	0	0.4846318
15	14	54	68	7	2	0.5	3.451087	0	0.9692635	0	0.4846318	0.4846318
16	17	55	68	15	2	0.5	3.451087	0	0.9692635	0	0.4846318	0.4846318
17	20	61	68	1	2	0.5	3.451087	0	0.9692635	0	0.4846318	0.4846318
18	1	38	69	21	2	0.5	3.451087	0	0.9989954	0	0.4994977	0.4994977
19	2	42	69	1	2	0.5	3.451087	0	0.9989954	0	0.4994977	0.4994977
20	5	46	69	3	2	0.5	3.451087	0	0.9989954	0	0.4994977	0.4994977
21	12	52	69	3	2	0.5	3.451087	0	0.9989954	0	0.4994977	0.4994977

```
//c - number of class labels , n – number of
records
```

```
Function Fuzzyvalue()
{
 $\mu = 1/c$ 
for( i = 1 to sp )
{
 $x1 [ i ] = 1 - 1/(1+\exp( -(\sigma) * ( x - \text{split point})))$ 
 $x2[i] = x1 [ i ] * \mu$ 
}
for( i = sp to n )
{
 $x3 [ i ] = 1 / (1+\exp( -(\sigma) * ( x - \text{split point})))$ 
 $x4[i] = x3 [ i ] * \mu$ 
} }
```

HSM is a node split measure which aims to reduce the height and number of nodes of decision tree using quasi linear mean of exponential function. And the quasi linear mean is heterogeneous in nature, hence the node split measure is referred as heterogeneous node split measure(HSM). HSM is computed at all the split points of all the attributes and the split point at which HSM is maximum is decided as the best split point.

Then the FHSM[8][9] is computed using the final fuzzy value f as

FHSM (split point) =

$$\frac{L1 + L2}{N} \log \left[\left(1 - \frac{L1}{(L1 + L2)} \right) * e^{-\frac{L1}{N}} + \left(1 - \frac{L2}{(L1 + L2)} \right) * e^{-\frac{L2}{N}} \right] +$$

$$\frac{U1 + U2}{N} \log \left[\left(1 - \frac{U1}{(U1 + U2)} \right) * e^{-\frac{U1}{N}} + \left(1 - \frac{U2}{(U1 + U2)} \right) * e^{-\frac{U2}{N}} \right]$$

Where N is the total number of records, L is the lower partition and U is the upper partition. L1, U1 are the sum of fuzzy values of class 1, L2, U2 are the sum of fuzzy values of class 2.

From the table 2, it can be observed that N=20, L1=0, L2=3.451886127, U1=4.846317672, U2=1.5 and by substituting the values in the above FHSM formula and by applying the logarithm to the base 10, FHSM = - 0.0155.

For each attribute, at every change in the class label, split points would be computed and the HSM is calculated for every split point of all the attributes. Let's say, there are four split points for the first attribute a1 and the HSM is computed for all the four points and pick the attribute with maximum value of HSM. Similarly, the HSM is calculated

for all the other attributes and the maximum HSM value of all the attributes is picked. It means the maximum HSM value from all the attributes is taken and selected as the best classifier attribute and also as the ROOT of the decision tree.

In the proposed work, using Wisconsin data set, “a9” at split point 1 and at 295 record with 0.4181 HSM value is chosen as ROOT which can be observed in fig 1.

III. FUZZY DECISION TREE CONSTRUCTION

The ROOT is chosen and it is required to determine the other nodes of the decision tree[10][11][12]. The crucial part is, how to compute the left subtree, right subtree of the Root in order to build the decision tree.

Now, from table 2, observe the values of the $x1*\mu$, $x2*\mu$. Firstly, let's modify the $x1*\mu$ records. Here, the values of $x1*\mu$ from 58 to 66 of a2 attribute remain the same, but from 68 to 69 of a2 attribute, the records would be replaced by $(0.5 - x2*\mu)$ that is, at 68 of a2 attribute (sno 14), the $x1*\mu$ is replaced with $(0.5 - 0.4846318) = 0.015$ and it is calculated for the other records in the similar manner.

Similarly, let's modify the $x2*\mu$ records. Here the values of $x2*\mu$ from 58 to 66 of a2 attribute would be replaced by $(0.5 - x1*\mu)$, but from 68 to 69 of a2 attribute, the records remain the same. That is, at 58 of a2 attribute (sno 3), the $x2*\mu$ is replaced with $(0.5 - 0.5) = 0$, at 66 of a2 attribute (sno 7), the $x2*\mu$ is replaced with $(0.5 - 0.48463177) = 0.015$ and it is repeated for the other records.

Now the $x1*\mu$ and $x2*\mu$ list of values were updated. Then the updated $x1*\mu$ values are taken as the fuzzy values to compute the left node for the ROOT a9. Now, the HSM is calculated for all the attributes a1 to a9 at various split points excluding a9 (As a9 is the ROOT). Now, the attribute having maximum value of HSM at a split point would become the left node for ROOT “a9”. That is a4 attribute at split point 1 at 245 record with HSM value as 0.43286 has become the left node for ROOT a9 which can be observed in the fig 1.

Similarly, the updated $x2*\mu$ values are taken as the fuzzy values to compute the right node for the ROOT a9. Now, the HSM is calculated for all the attributes a1 to a9 at various split points excluding a9 and a4 (As a9 is the ROOT, a4 is the left child). Now, the attribute having maximum value of HSM at a split point would become the right node for ROOT “a9”. That is a5 attribute at split point 2 with HSM value as 0.41595 has become the right node for ROOT a9 which can be observed in the fig 1.

Then the fuzzified decision tree is constructed shown in fig 1, using HSM as the best classifier attribute. The tree would be, (Root – A9, split point - 1, HSM value - 0.4181), (left child – A4, split point - 1, HSM value - 0.43286), (right child – A5, split point -2, HSM value -0.41595) and so on and the complete decision tree is built in the same manner.

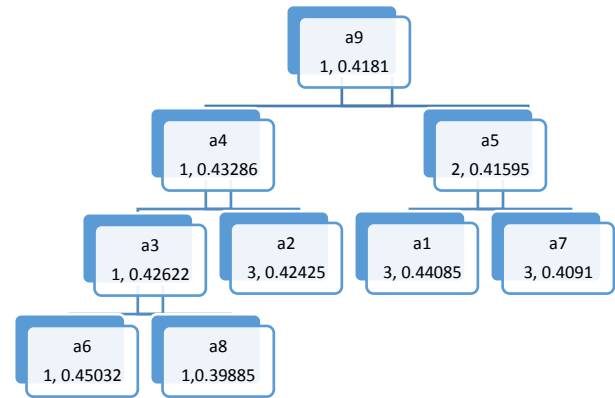


Fig .1. The decision tree with Root, & other nodes with their HSM values.

IV. PROPOSED METHODOLOGY

The KEY point is, the HSM is calculated for all the attributes at various split points and the maximum value of HSM of an attribute is decided as the ROOT which is considered as the Best classifier attribute for the complete fuzzified decision tree. Similarly, all the other nodes of the decision tree is built using the maximum value of HSM as explained above in the Fuzzy decision tree section. Now, the HSM values of the nodes(attributes) of the decision tree are considered as the weight of that corresponding attributes in our proposed work.

This point is like a Bridge from Decision tree to Neural Networks which works collaboratively. That is, the results of fuzzy decision tree are taken and implemented for the neural network. That is, the maximum HSM values of a1, a2, a3, ...a9 attributes which are 0.44085, 0.42425, 0.42622, ...0.4181 were considered as the “weights” of those corresponding attributes to classify using neural networks.

The Weight Adjustment Algorithm for the complete proposed methodology is as follows :

1. Read the data set, // 9 attributes and a class label
2. Sort an attribute and find the split point,
3. Compute the fuzzy values of attribute above the split point and below the split point,
4. Calculate the HSM value of that attribute using the fuzzy values,

5. Similarly calculate the HSM values for all the attributes and pick the maximum HSM value,
6. Choose one attribute with maximum HSM value as the ROOT of fuzzy decision tree,
7. Similarly compute the other nodes and build the fuzzy decision tree,
8. Pick the HSM values of all the nodes(attributes) of fuzzy decision tree and assign them as weights to the corresponding attributes,
9. The data set which is normalized and multiplied with HSM weights are given to the different types of neural network such as Deep Learning, Backpropagation, Multi Layer Feed forward, Run and compute the execution time and classification accuracy.

//W – weight of attribute , m – number of attributes, n – number of records

Function HSMweight()

```
{
for( i = 1 to m)
{
for( j = 1 to n)
{
X I j = ( j - j min ) / ( j max - j min )
// normalize the data between 0 and 1
y I j = X I j * W I //HSM
weight is multiplied to the input attribute
} } }
```

Using the above weights, the testing is done with three types of input data such as a) Wisconsin data set b) normalized Wisconsin data set, c) normalized data with HSM weights (HSM weighted inputs) and implemented on various types of neural networks such as Deep Learning, Backpropagation and Multi Layer Feed Forward and effective results are observed in all three cases.

V. IMPLEMENTATION USING DEEP LEARNING

Deep Learning[13][14][15] is gaining lot of importance in the recent times. Deep learning has become so popular in the fields of pattern recognition and computer vision etc.. Deep learning generally uses two types of networks such as convolutional neural network and Autoencoders. The Sparse Autoencoder is used for the proposed work. The network comprises of input layer, two hidden layers, softmax layer, output layer. The two hidden layers are implemented using encoders. First the hidden layers are trained in an unsupervised fashion and train the softmax layer and finally join all the hidden and softmax layers to form a deep

network which is trained in a supervised fashion. The first hidden layer's encoder reads the input and extract main features and the second hidden layer's encoder reads the features that were extracted by the first hidden layer (encoder) and still learns the small representations (micro level features) of the input data.

In fact the deep learning neural networks (refer Fig – 2) classifies the data in a most efficient way. Hence, the testing is performed by giving the normalized Wisconsin data set and HSM weighted inputs to the network and verified the total execution time. It is observed that the classification efficiency is same for the two cases, but the HSM weighted inputs have executed the code much faster than the normalized data. The corresponding observations are presented in Results section.

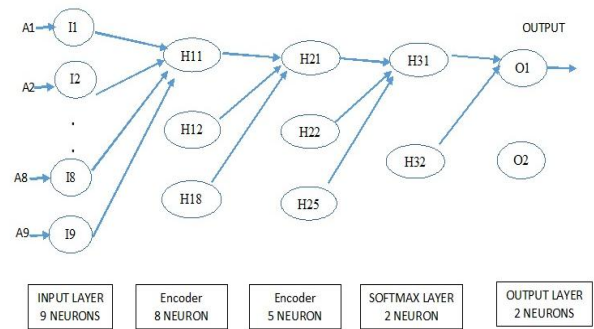


Fig 2 – Deep Learning

Implementation using Back Propagation neural networks:

Backpropagation[16][17] is nothing but propagating the error backward, and after the adjustment of the weights, the optimal classification is achieved. In this paper, it is proposed to measure and compare the classification accuracy in three aspects. They are:

- 1) check the classification accuracy of the data set using fuzzy decision tree,
- 2) check the classification accuracy of the data set by normalizing the data between 0 and 1 using neural networks,
- 3) check the classification accuracy for the normalized data set with the multiplied HSM weights using neural networks.

The first aspect would be, the generation of the fuzzy decision tree using the train data set. Then the classification accuracy is measured by applying the test data set for the fuzzy decision tree. After generating the Rules from the fuzzy decision tree, then the test data set is given to the Rules and classification accuracy is measured and it is

observed that the code is run with eight errors out of 232 test records with this fuzzy decision tree which comes to 96.55% efficiency.

The second aspect would be, the same data set is taken, normalize the data set between 0 and 1(refer function `HSMweight()`), and then classify the data using neural networks. It means, the train data, test data, and the neural network configuration file which contains “Input_Neurons, Hidden_Neurons, Output_Neurons, Learning_Rate, Momentum, Train_Input_Records, Train_Output_Records, No_of_Iteration ” are given to the neural network code, run it, and measure the classification accuracy. Regarding the neural networks, the multi layer(Input layer, hidden layer, output layer) neural network model(refer fig 3) with back propagation is considered. The input layer comprises of 9 neurons, hidden layer of 8 neurons and the output layer with 3 neurons and learning rate of 0.25 ,the momentum of 0.9 is considered and the training, testing records, number of iterations are given to the neural network model. The input layer is given with the 9 attributes of the normalized data set, and the output layer gives an output of 001(1) or 010(2) to three neurons where (001)1 is benign and (010)2 is malignant. The sigmoid Activation function ($1 / (1+e^{-x})$) is used in our model where x is the linear function of weight, attribute and the bias. The error is calculated at the output layer and it is shared back to the neurons of the model using the concept of back propagation. It is observed that the code is run with four errors out of 232 test records with the neural networks which comes to 98.27% accuracy.

The third aspect would be, the same data set is taken, normalize the data set between 0 and 1 and multiply with the HSM weights (refer function `HSMweight()`), and then classify the data using neural networks with back propagation as it is done in the second aspect . It means, the train data, test data, and the neural network configuration file are given to the neural network code and measure the classification accuracy. It is observed that the code is run with three errors out of 232 test records with the neural networks which comes to 98.7% accuracy which is a biggest improvement of the classification accuracy .

Implementation using Multi-Layer Feed Forward neural networks:

The network which does not contain cycles or the feedback loops is called a feed forward neural network. Here, the network comprises of input layer, hidden layer and output layer. The testing is done using the Wisconsin data set, normalized Wisconsin data set and HSM weighted inputs on the network comprising of single, two, three and four hidden layers (refer Fig-5)and got good results in all the cases.

Please refer Results.

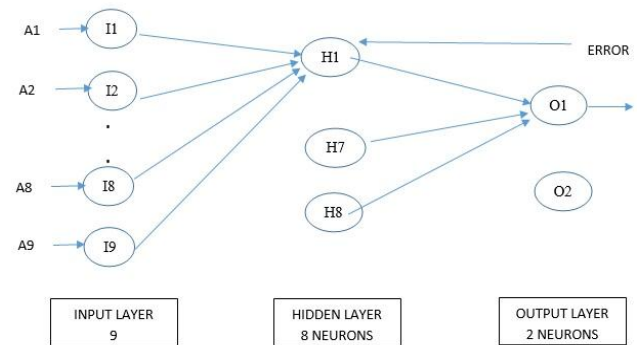


Fig 3 – Backpropagation network

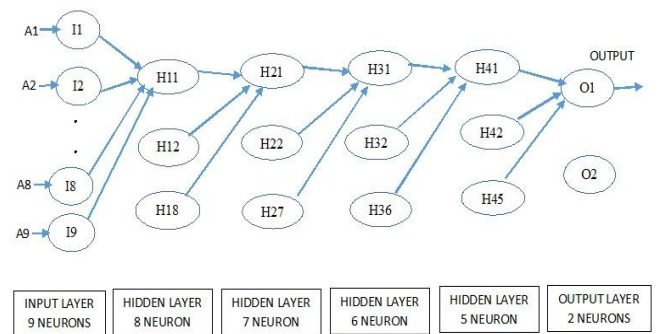


Fig 4 – Multi Layer Feed Forward neural network

VI. RESULTS

The Results related to the Deep Learning, Backpropagation and Multi Layer Feed Forward neural networks are illustrated in the following.

Deep Learning:

When the different forms of input data(as explained above) are given to the Deep Learning, the Results are in the following manner (Refer table 3, Fig 6) and the execution speed is increased by 220%.

Back Propagation neural networks :

When the different forms of input data(as explained above) are given to the Backpropagation neural network, the Results are in the following manner (refer table 4, Fig 7).

Multi-Layer Feed Forward neural networks:

When the different forms of input data(as explained above) are given to the Multi-Layer Feed Forward neural networks with different number of hidden layers, the Results are in the following manner. (Refer table 5, Fig 8)

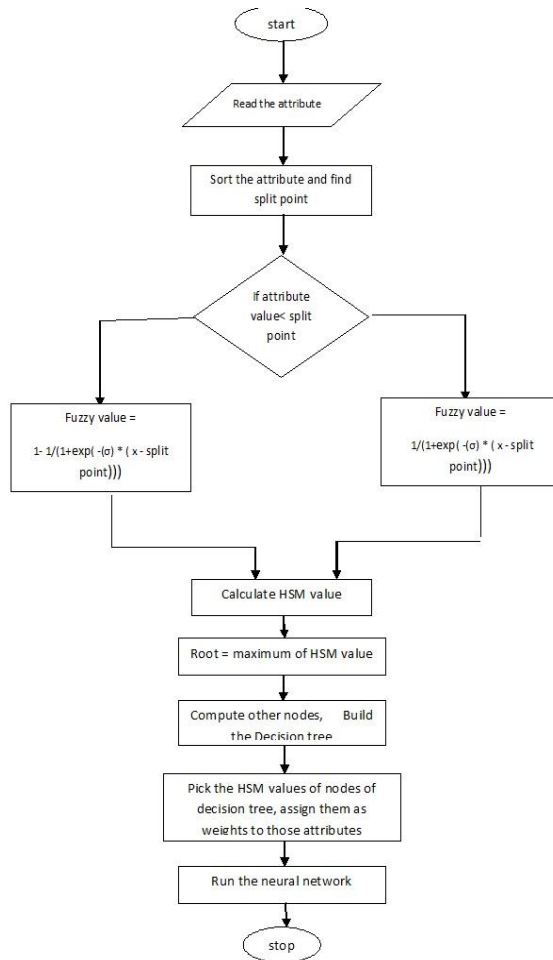


Fig 5 – flow chart for the complete methodology

Fig 5 – flow chart for the complete methodology

Table 3 – Execution time of Deep Learning

Implementation	Total time of Execution(seconds)
Normalized Wisconsin data set	6.9 ± 0.1
HSM weighted inputs	2.1 ± 0.1

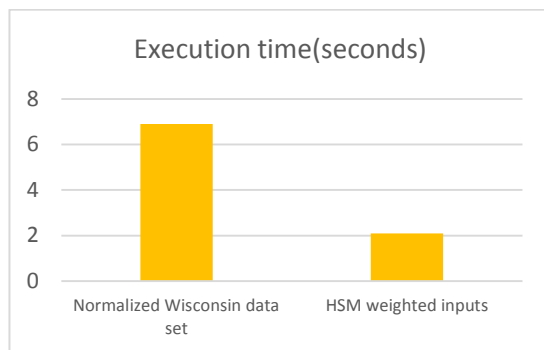


Fig 6 – Deep Learning execution time

Table 4 – classification accuracy of decision tree & back propagation neural network

Sno	Description of the implementation	Classification Accuracy
1	Wisconsin Data set using Decision tree	96.5%
2	Wisconsin Data set which is normalized and using neural networks	98.2%
3	Wisconsin Data set which is normalized and multiplied with HSM weights using neural networks	98.7%

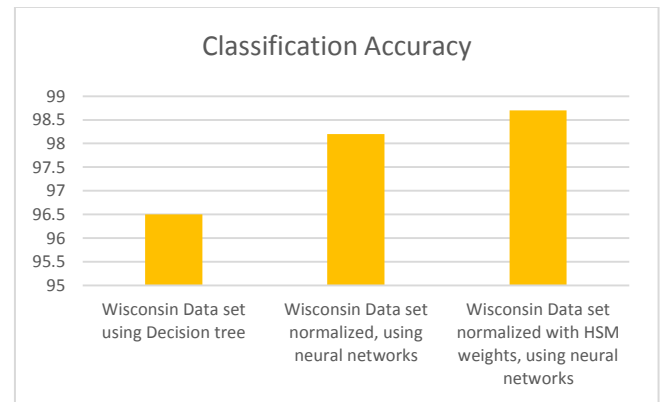


Fig 7 – Backpropagation Neural Network Accuracy

Table 5 – classification accuracy of Multi Layer Feed Forward neural network

Implementation	Classification Accuracy			
	Using 1 Hidden layer	Using 2 Hidden layers	Using 3 Hidden layers	Using 4 Hidden layers
Wisconsin data set	95.4	95.5	95.9	96
Normalized Wisconsin data set	96.8	97	97.1	97.2
HSM weighted inputs	96.9	97.1	97.4	97.7

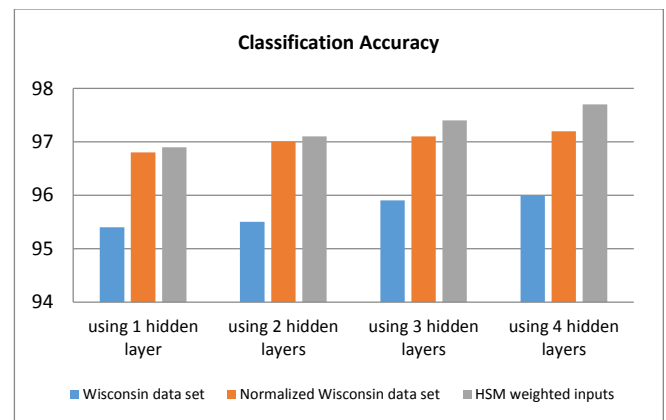


Fig 8 – Multi Layer Feed Forward Neural Network Classification Accuracy

VII. COMPARISON AND ANALYSIS OF RESULTS

(GINI INDEX VS HSM BASED)

The weights are computed by considering the gini index and HSM and they are used to implement using Deep Learning, back propagation and feed forward neural networks. And it is observed that there is an increase in the classification speed of 220% using HSM and an increase of 150% using gini index implementation using Deep Learning(refer table 6 & fig 9) . And the classification Accuracy achieved is almost the same by considering gini index and HSM based concepts implemented using back propagation, feed forward neural networks.

The same work is carried out on page blocks data set by considering gini index and implemented using deep learning. This data set comprises of 5473 records, 10 attributes and a class label. The class label contains 5 classes 1,2,3,4,5. The execution time is very rapid with the proposed concept where there is an increase of 300% faster classification(refer table 6 & fig 9).

Table 6 – classification execution time of various data sets using Deep learning

Implementation	Wisconsin data set		Page blocks data set
	Gini (seconds)	HSM (seconds)	Gini (seconds)
Normalized data set	6.3	6.9	16
Weighted input	2.5	2.1	4
Increase in the fast classification rate	150%	220%	300%

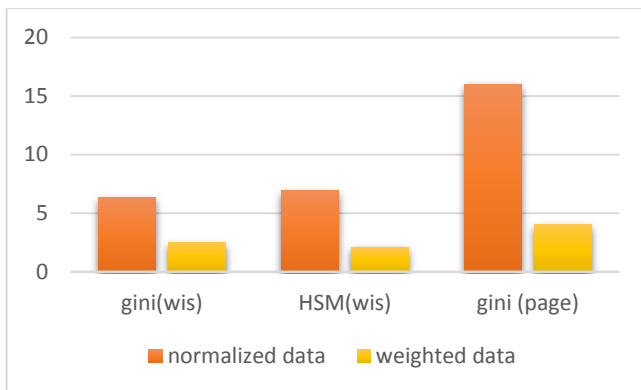


Fig 9 – Deep Learning execution time in seconds

VIII. CONCLUSION

The fuzzy decision tree is constructed using HSM as the best split measure. To enhance the speed & accuracy of the classification using neural networks, the maximum HSM value of each attribute is taken as the Weight of the

corresponding attribute for the weight adjustment algorithm and tested using Deep Learning, Backpropagation, Multi Layer Feed Forward neural networks and achieved very good results. And as a future work, there are few parameters like information gain etc. which plays a dominant role in the classification of the data using various supervised learning algorithms and the values of those parameters can be taken as the weight and compare all the parameters and choose the one which would give the best classification accuracy. Even the genetic algorithm can be applied to suggest the best optimal parameter to derive the weights. If the classification speed is the criteria, then HSM based implementation may be applied and if the classification accuracy is required, the gini index or HSM based implementation may be applied for the neural networks. In the recent times, the BIG DATA challenges for the data storage, data analysis and querying etc. It takes long time to do the analysis of big data. The proposed gini index or HSM based concept can be applied for the big data to classify using a neural network in a faster way.

ACKNOWLEDGEMENT

I am very much thankful to my supervisor Prof.K.Thammi Reddy, my co supervisor Prof. V.Valli Kumari for their guidance and constant monitoring of my PhD work and gratitude to my colleague KVSRRP Varma for his support.

REFERENCES

- [1] Alsabti, K., Ranka, S., & Singh, V. (1998). *CLOUDS: A decision tree classifier for largedatasets*. In Knowledge discovery and data mining (p 2–8),
- [2] B.chandra, Venkata Nareshbabu Kuppili, *Heterogeneous node split measure for decision tree construction*, IEEE 2011, p872-877
- [3] Chandra, B., & Paul, P. (2007). *On improving the efficiency of SLIQ decision tree algorithm*. In Proceedings of IEEE international joint conference on neural networks, IJCNN – 2007,
- [4] S. Sathyamoorthy, "Data Mining and Information Security in Big Data", International Journal of Scientific Research in Computer Science and Engineering, Vol.5, Issue.3, pp.86-91, 2017.
- [5] Chengming, Q. (2007). *A new partition criterion or fuzzy decision tree algorithm*. In Intelligent information technology application, workshop on 2–3 December 2007 (p 43–46),
- [6] Adamo, J. M. (1980). *Fuzzy decision trees*. Fuzzy Sets and Systems, vol 4,issue 3, 1980,p 207–219,
- [7] Cristina, O., & Wehenkel, L. (2003). *A complete fuzzy decision tree technique*. Fuzzy Sets and Systems, vol 138, 2003, p 221–254,
- [8] Prakash bethapudi, E. Sreenivas Reddy, Kamadi VSRP Varma, *Gaussian fuzzy HSM algorithm for classification of BIRADS data set using Decision Tree*, IJCTA, 2015,
- [9] Shalini Bhaskar Bajaj, Akshaya Kubba, "FHSM : Fuzzy heterogeneous split algorithm for decision trees", IEEE 2014.
- [10] Breiman, L., Friedman, J. H., Olshen, J. A., & Stone, C. J. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth International Group,
- [11] Chandra, B., Mazumdar, S., Arena, V., & Parimi, N. (2002). *Elegant decision tree algorithms for classification in data mining*. In Proceedings of the 3rd international conference

- on information systems engineering (workshops), IEEE ,CS (pp. 160–169),
- [12] Chandra, B., & Paul, P. (2006). *In Robust algorithm for classification using decision trees*, CIS-RAM 2006 (p. 608–612). IEEE,
- [13] Srinivas Jagirdar, K. Venkata Subba Reddy and Ahmed Abdul Moiz Qyser, "Cloud Powered Deep Learning-Emerging Trends", International Journal of Computer Sciences and Engineering, Vol.4, Issue.6, pp.186-190, 2016.
- [14] Victoria j hodge, simon o keefe, jim Austin, *Hadoop neural network for parallel and distributed feature selection*, Elsevier(Neural networks), 2016, vol 78, p 24-35,
- [15] Jihun kim, jong hong kim, gil-jin jang, minho lee, *fast learning method for convolutional neural networks using extreme learning machine and its application to lane detection*, Elsevier (Neural networks), 2017, vol 87, p 109-121
- [16] M.W.Gardner, S.R. Dorling, *Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences*, Elsevier, Atmospheric environment, 1998, pp 2627 – 2636,
- [17] Bing Cheng, D.M. Titterington, *Neural Networks: A Review from a Statistical Perspective*. Vol 9, no 1, 1994, p2-54.

Authors Profile

S.V.G.REDDY completed M-Tech(CST) from Andhra University and currently pursuing PhD (computer science & engineering) from JNTU Kakinada. He is working as Associate Professor, Department of CSE, GIT, GITAM University. His area of research work is data mining and machine learning. He has guided various B.Tech, M-Tech projects and had publications in several journals. His area of interest is big data, Internet of things.



Prof. K.THAMMI REDDY completed his PhD (computer science & engineering) during 2008 from JNTU Hyderabad. He is working as Professor, Department of CSE, GIT, and also serving as Director, IQAC, GITAM University. His area of research work is data mining, machine learning and cloud computing with Hadoop. He has guided various B.Tech, M-Tech projects and had publications in several reputed journals and conferences.



Prof. V.VALLI KUMARI completed her PhD(CSSE) from Andhra University during 2006. She is working as professor, Department of CSSE, college of engineering, Andhra University. Her area of interest is Software Engineering, Network Security & Cryptography, Privacy issues in Data Mining and Web Technologies. She has guided various B.Tech, M-Tech projects and had publications in several reputed journals and conferences. She received best researcher award and other various awards in the fields of teaching and research. She has undergone and completed various research projects.

