Incorporation of Fuzziness in ID3 and Generation of Network Architecture

Pawan K. Singal, Sushmita Mitra and Sankar K. Pal

Machine Intelligence Unit, Indian Statistical Institute, Calcutta, India

This article first describes a fuzzy version of ID3, called fuzzy ID3 by incorporating fuzziness at input, output and node levels. A fuzziness measure is computed at each node, in terms of class membership, to take care of the uncertainty arising from overlapping regions. The measure is such that in the crisp (non-overlapping) case, the algorithm boils down to the classical ID3. A confidence factor is estimated at the nodes for both making a decision and determining the rule base for network mapping. In the second part, we deal with a scheme of designing a fuzzy knowledge-based network by encoding an MLP with the rules generated using fuzzy ID3, whereby the network topology is automatically determined. The frequency of samples (representative of a rule) and the confidence factors of unresolved/ambiguous nodes are taken into consideration during mapping. The effectiveness of the system, in terms of recognition scores and speed of convergence, is demonstrated on two real life data sets.

Keywords: Classification; Fuzzy ID3; Neural tree; Neuro-fuzzy paradigm; Rule generation; Soft computing

1. Introduction

Decision trees have been used extensively as classifiers in pattern recognition [1]. By applying the decision tree methodology, a difficult decision can be broken into a sequence of simpler steps. Because of the inherent tree structure, only some of all possible questions need to be asked in the process. Each node in the decision tree is either a leaf node (decision node) or an internal node (a testing node). Each leaf node usually represents a unique class. When a data point reaches a leaf, after traversing the tree in a top-down manner from its root, we decide the class of the data point as that represented by this leaf node. Each internal node represents a test, with respect to a feature, that is made in the process of arriving at a decision.

Quinlan popularised the concept of decision trees by introducing ID3 [2], which stands for *Interactive Dichotomiser 3*. Systems based on this approach use an information theoretic measure of entropy for assessing the discriminatory power of each attribute. ID3 is a popular and efficient method of making decision for classification of *symbolic* data. Generally, it is not suitable in cases where numerical values are to be operated upon. Since most real life problems deal with non-symbolic (numeric, continuous) data, they must be discretised prior to attribute selection. Another problem with ID3 is that it cannot provide any information about the intersection region where the pattern classes are overlapping.

Generally, Artificial Neural Nets (ANNs) consider a fixed topology of neurons connected by links in a pre-defined manner. These connection weights are usually initialised by small random values. Recently, there have been some attempts at improving the efficiency of neural computation by using knowledge-based nets. This helps in reducing the searching space and time while the network traces the optimal solution. Knowledge-based networks [3–5] constitute a special class of ANNs that consider crude domain knowledge to generate the initial network architec-

Correspondence and offprint requests to: S. K. Pal, Machine Intelligence Unit, Indian Statistical Institute, Calcutta 700 035, India. Email: sankar@isical.ac.in

ture, which is later refined in the presence of training data. Such a model has the capability of outperforming a standard MLP, as well as other related algorithms, including symbolic and numerical ones [3,4].

Both decision trees and neural networks are most commonly used tools for pattern classification. In recent years, enormous work has been done in an attempt to combine the advantage of neural networks and decision trees. The new architecture so obtained is called a *neural tree* [6–11]. The neural tree architecture reported in the literature can be grouped according to the learning paradigm employed for their training. Most of the existing neural tree architecture are either directly or indirectly related to feed-forward neural networks. In fact, the characterisation neural tree has been indistinguishably used to describe approaches using feed-forward neural network as a building element, in order to improve the performance of decision trees. This also includes the approaches employing the decision tree as a tool for building and training feed-forward networks.

In the first family of approaches, one attempts to develop a tree structure containing a feed-forward neural network among the nodes. Some of the remarkable work in this area are Sankar and Mammone's *Neural Tree Network* (NTN) [6] and *Competitive Neural Trees* (CNeT) by Behnke and Karayiannis [7]. The architecture of NTN consists of single layered neural network connected in the form of a tree. On the other hand, CNeT has a structured architecture, and performs hierarchical clustering by employing unsupervised learning at node level.

In the second family of approaches, an attempt is made to build neural networks either by developing tree structured neural networks or by mapping decision trees to multilayer neural network. Sethi [9] proposed a procedure for mapping a decision tree into a multilayered feed-forward neural network with two hidden layers. The mapping rules are as follows: (a) the number of neurons in the first hidden layer equals to the number of internal nodes in the tree. Each of these neurons implements one of the decision functions of the internal nodes; (b) the number of neurons in the second hidden layer equals the number of leaf nodes in the tree; (c) the number of neurons in the output layer equals to the number of distinct classes. Ivanova and Kubat [10] have directly mapped a decision tree into a three-layered network, such that each conjunction (in a rule) is modeled by a hidden node. The input domain is partitioned into a set of non-overlapping intervals of attributes, which are then mapped to input nodes of the network. Setiono and Leow [11] computed binary training patterns from the decision rules and attribute intervals. These are used to train a feed-forward network, which is then pruned to generate an optimal topology.

The fusion of fuzzy sets with decision trees enables one to combine the uncertainty handling and approximate reasoning capabilities of the former with the comprehensibility and ease of application of the latter. This enhances the representative power of decision trees *naturally* with the knowledge component inherent in fuzzy logic, leading to better robustness, noise immunity and applicability in uncertain/imprecise contexts. Fuzzy decision trees [12] assume that all domain attributes or linguistic variables have pre-defined fuzzy terms, determined in a data-driven manner using fuzzy restrictions. The information gain measure, used for splitting a node, is modified for fuzzy representation and a pattern can have non-zero match to one or more leaves. Designs of fuzzy decision trees have also been reported in the literature [12–15].

The present article consists of two parts. In the first part we attempt to develop a fuzzy version of ID3, called *fuzzy ID3*. Fuzzy set-theoretic concepts are introduced at the input, output and node levels of the ID3 algorithm to handle uncertainty. A scheme for linguistic discretisation of continuous attributes is developed. A fuzziness measure is computed at the node level, in terms of class membership, to take care of overlapping classes. Fuzziness is incorporated in such a way that in the crisp case, when the classes are not overlapping, it boils down to the classical ID3. In the case of overlapping classes, it provides extra information regarding the intersecting regions. A confidence factor is estimated at the nodes while reaching a decision and determining the rule base.

The second part deals with a scheme of encoding an MLP with the rules generated using fuzzy ID3. The network topology is automatically determined, and the initial connection weights directly mapped. This results in a fuzzy knowledge-based network. Since a decision tree is used in the process, one can also call it a neural tree. The frequency of samples, representative of a rule, is taken into consideration while mapping the initial weight values of the MLP. A novel approach for mapping confidence factors of unresolved/ambiguous nodes directly into a fuzzy neural network is also described. The resultant knowledge encoded network leads to faster convergence. The effectiveness of the algorithm is demonstrated on two real life data sets, viz., *Vowel* and *Balance Scale* data.

2. ID3 and the Need for Fuzzification

Before going into details of the proposed fuzzy ID3 algorithm, let us first describe the classical ID3 in brief. This is followed by explaining why one needs to incorporate fuzziness in ID3.

2.1. ID3 Algorithm

ID3 uses an information-theoretic approach. The procedure is that at any point we examine the feature that provides the greatest gain in information or, equivalently, the greatest decrease in entropy. Entropy is defined as $-p \log 2\ p$, where probability p is determined on the basis of frequency of occurrence.

The general case is that of N labelled patterns partitioned into sets of patterns belonging to classes C_i, $i = 1,2,3, \ldots, l$. The population in class C_i is n_i. Each pattern has n features, and each feature has $J(\geq 2)$ values. The ID3 prescription for synthesising an efficient decision tree can be stated as follows [16]:

Step 1. *Calculate initial value of entropy*

$$Entropy(I) = \sum_{i=1}^{l} -(n_i/N)\log_2(n_i/N)$$

$$= \sum_{i=1}^{l} -p_i \log_2 p_i \qquad (1)$$

Step 2. *Select that feature which results in the maximum decrease in entropy (gain in information), to serve as the root node of the decision tree.*
Step 3. *Build the next level of the decision tree providing the greatest decrease in entropy.*
Step 4. *Repeat Steps 1 through 3*. Continue the procedure until all subpopulations are of a single class and the system entropy is zero.

At this stage, one obtains a set of leaf nodes (subpopulations) of the decision tree, where the patterns are of a single class. Note that there can be some nodes which cannot be resolved any further.

2.2. Relevance of Fuzzy Sets

As we have mentioned earlier, ID3 cannot provide any information in the intersection region when the classes are highly overlapped. Let us explain why ID3 fails to give any information when there are overlapping pattern classes. In ID3 algorithm we partition the sample space in the form of a tree by using attribute values only. When two sample points from two different overlapping classes lie in the intersection region, the corresponding features for these samples are the same. This implies that they travel through the same path in the decision tree and finally land onto the same node. We cannot split the node further because the gain in entropy ΔEnt will always be zero, which is one of the stopping criteria during tree building. Thus, in the overlapped region, an attribute value fails to provide any decision about the leaf node.

To get more information in this regard, one needs to dig the data further. Intuition tells us that the pattern points of any particular class must be clustered around some characteristic prototype or class centre. We wish to exploit the fact that the points nearer to this centre have higher belongingness to that class, as compared to the points further away from it. This brings in the concept of fuzzy sets which allows a pattern to have finite non-zero membership to more than one class. Here lies the utility of employing fuzzy sets to model overlapping/ambiguous real life pattern classes [17].

Moreover, the conventional ID3 can handle only discrete-valued/symbolic attributes. Real life problems, on the other hand, require the modelling of continuous attributes. Fuzzy sets can also be useful in this aspect.

3. Fuzzy ID3

Fuzzy set-theoretic concepts are introduced at the input, output and node levels of the ID3 algorithm. Linguistic inputs [18,19] enable the handling of continuous attributes at the input. The output is evaluated in terms of class membership [18,19] values. A fuzziness measure is computed at the node level to take care of overlapping classes. This is reducible to the classical entropy, used in the conventional ID3, in the crisp case. A confidence factor is estimated at the nodes while reaching a decision and determining the rulebase.

3.1. Input Representation

Any input feature value is described in terms of some combination of overlapping membership values in the linguistic property sets *low* (L), *medium* (M) and *high* (H). An n-dimensional pattern $\mathbf{F}_i = [F_{i1}, F_{i2}, \ldots, F_{in}]$ is represented as a $3n$-dimensional vector [19,17]

$$\mathbf{F}_i = [\mu_{low(F_{i1})}(\mathbf{F}_i), \mu_{medium(F_{i1})}(\mathbf{F}_i), \mu_{high(F_{i1})}(\mathbf{F}_i), \ldots, \mu_{high(F_{in})}(\mathbf{F}_i)] \qquad (2)$$

where the μ values indicate the membership functions of the corresponding linguistic π-sets *low*, *medium* and *high* along each feature axis. Each μ value is then discretised, using a threshold, to enable a convenient mapping in the ID3 framework.

When the input feature is numerical, we use the π-fuzzy sets (in the one dimensional form), with range [0,1], represented as

$$\pi(F_j;c,\lambda) = \begin{cases} 2\left(1 - \dfrac{\|F_j - c\|}{\lambda}\right)^2, & \text{for } \dfrac{\lambda}{2} \le \|F_j - c\| \le \lambda \\ 1 - 2\left(\dfrac{\|F_j - c\|}{\lambda}\right)^2, & \text{for } 0 \le \|F_j - c\| \le \dfrac{\lambda}{2} \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

where $\lambda(>0)$ is the radius of the π-function with c as the central point. Note that features in linguistic and set forms can also be handled in this framework [19].

3.2. Output Representation

Consider an l-class problem domain. The membership of the ith pattern in class k, lying in the range [0,1], is defined as [17]

$$\mu_{ik}(\mathbf{F}_i) = \frac{1}{1 + \left(\dfrac{z_{ik}}{f_d}\right)^{f_e}} \quad (4)$$

where z_{ik} is the weighted distance of the training pattern $\mathbf{F}_i$ from class C_k, and the positive constants f_d and f_e are the denominational and exponential fuzzy generators controlling the amount of fuzziness in the class membership set.

3.3. Fuzziness Measure

Fuzziness is incorporated at the node level by changing the decision function from classical entropy to a *fuzzy measure FM*. This is defined as

$$\begin{aligned} FM(I) &= \sum_{k=1}^{l} \left(\frac{1}{N} \sum_{i=1}^{N} \min(\mu_{ik}, 1 - \mu_{ik}) \right. \\ &\quad \left. - \left(\frac{n_k}{N}\right) \log_2 \left(\frac{n_k}{N}\right) \right) \quad (5) \\ &= \sum_{k=1}^{l} \left(\frac{1}{N} \sum_{i=1}^{N} \min(\mu_{ik}, 1 - \mu_{ik}) \right. \\ &\quad \left. - p_k \log_2 p_k \right) \end{aligned}$$

where N is the number of pattern points in the training set, l is the number of classes, n_i is the population in class C_i, p_k is the *a priori* probability of the kth class and μ_{ik} denotes the membership of the ith pattern to the kth class.

The expression for FM is so defined that when the class membership values are zero or one (crisp), it boils down to classical entropy. The first term on the right-hand-side of Eq. (5) ensures that pattern points lying in overlapping regions are assigned lower weights during the construction of the decision tree; this is intuitively appealing. The reason for this lower weighting is that such ambiguous patterns (having μ values close to 0.5) lead to an increase in FM, thereby placing an impedance to its minimisation.

3.4. Estimating Confidence of Nodes

Mandal et al. [20] provided a scheme to calculate the *Confidence Factor* (CF) and rulebase in order to infer the belongingness of a point to a particular class in terms of multiple choices (e.g. first choice, second choice). It is defined as

$$CF = \frac{1}{2}\left[\mu' + \frac{1}{l-1}\sum_{j=1}^{l}\{\mu' - \mu_j\}\right] \quad (6)$$

where μ_j is the class membership of the pattern to the jth class C_j, and μ' is the highest membership value. The concept of CF takes care of the fact that the difficulty in assigning a particular class label depends not only upon the highest entry μ', but also on its differences from the other entries μ_j.

Note that for computing CF_1 and CF_2 (when second choice is necessary, leaving out the first choice in Eq. (6) [20], μ' is set equal to first and second highest membership values, respectively. Let CF^k denote the CF_1 value of a pattern corresponding to class C_k (having highest membership value). Then the rule base is as follows:

1. If $0.8 \le CF^k \le 1.0$ then *very likely* class C_k and there is no second choice.
2. If $0.6 \le CF^k < 0.8$ then *likely* class C_k and there is second choice.
3. If $0.4 \le CF^k < 0.6$ then *more or less* likely class C_k and there is second choice.
4. If $0.1 \le CF^k < 0.4$ then *not unlikely class* C_k and there is no second choice.
5. If $CF^k < 0.1$ then *unable to recognise* class C_k and there is no second choice.

In the case of single choice (when rule 1 fires), we update the confidence factor to one, such that

$CF_1 = 1$ and $CF_2 = 0$. For other choices (when rules 2 to 4 fire), we additionally compute CF_2 corresponding to the class with second highest membership value. Finally, an aggregation is made at the node level.

3.5. Algorithm

Let us now describe the algorithm in detail.

1. *Calculate initial value of fuzziness measure (FM) using* Eq. (5).
2. *Select a feature to serve as the root node of the decision tree.*
   (a) For each attribute A_i, i = 1,2, . . ., $3n$, partition the original population into two subpartitions according to the values a_{ij} ($j = 0$ or 1, stands for the attribute value 0 or 1) of the attribute A_i. Although there are n_{ij} patterns coming down branch a_{ij}, these patterns need not necessarily belong to any single class.
   (b) Evaluate the *FM* for each branch.
   (c) The decrease in FM as a result of testing attribute A_i is $\Delta\text{FM}\ (i) = FM(\text{I}) - FM(I,A_i)$.
   (d) Select an attribute A_k that yields the greatest decrease in *FM*, i.e. for which $\Delta FM\ (k) > \Delta FM(i)$, for all $i = 1,2,\ldots, l,\ i \neq k$.
   (e) The attribute A_k is then root of the decision tree.
3. *Build the next level of the decision tree.* Select an attribute $A_{k'}$ to serve as the level 1 node such that, after testing on $A_{k'}$ along all branches, we obtain the maximum decrease in FM.
4. *Repeat steps 3 through 5.* Continue the process until all sub-populations reaching a leaf node are of any single class or the decrease in *FM*, i.e. ΔFM, is zero. Mark the terminal nodes which have pattern points belonging to more than one class. Such nodes are termed as *unresolved* nodes.
5. *For each unresolved node do the following*:
   (a) Calculate the confidence factors CF_1 and CF_2 as in Eq. (6).
   (b) Identify the classes corresponding to which there is at least one CF_1 or CF_2.
   (c) For each pattern point in the node, if $CF_1 \geq 0.8$ then put $CF_1 = 1$, $CF_2 = 0$.
   (d) Consider the classwise average summation of the *CF* values.
   (e) Mark the classes getting the highest and second highest *CF* values. Declare this node as the representative of the two classes found with membership corresponding to the two highest *CF* values.

4. Generating Network Architecture

Before going into the details of rule generation and network mapping, we introduce the different parameters of a multilayer perception (MLP). Let the output of a neuron in any layer (h) of an MLP, other than the input layer ($h = 0$), be $y_j^h = 1/[1 + \exp(-\Sigma_i y_i^{h-1} w_{ji}^{h-1})]$, where y_i^{h-1} is the state of the ith neuron in the preceding $(h-1)$th layer and w_{ji}^{h-1} is the weight of the connection from the ith neuron in layer $h-1$ to the jth neuron in layer h. For nodes in the input layer, y_j^0 corresponds to the jth component of the input vector. Note that $x_j^h = \Sigma_i y_i^{h-1} w_{ji}^{h-1}$.

The $3n$-dimensional input vector of Eq. (2) is clamped at the input layer to the input nodes $[y_1^0, y_2^0, \ldots, y_{3n}^0]$. Here $y_1^0, \ldots, y_{3n}^0$ refer to the activations of the $3n$ neurons in the input layer. The l-dimensional output vector, in terms of class membership values (μ) of patterns by Eq. (4), is clamped at the l nodes in the output layer of the MLP. During training, the weights are updated by backpropagating errors with respect to these membership values such that the contribution of uncertain/ambiguous pattern vectors is automatically reduced.

4.1. Rule Generation and Encoding

Now we go into the details of the knowledge encoding algorithm using the decision tree generated by fuzzy ID3. Let us consider the leaf nodes only. The path from the root to a leaf can be traversed to generate the rule corresponding to a pattern from that class. In this manner, one obtains a set of rules for all the pattern classes, in the form of intersection of the features/attributes encountered along the traversal paths. The ith attribute is marked as A_i or $\bar{A}_i$, depending on whether the traversal is made along value a_{i1} or a_{i0}. Each rule is marked by its frequency, that is the number of pattern points reaching this leaf node. Note that each leaf node that has pattern points corresponding to only one class is termed *resolved*. We assign the confidence value *CF* of each such rule as unity.

The rules corresponding to the unresolved leaf nodes are also generated by traversing the corresponding paths from the root. Here each rule pertains to two or more classes. Hence we obtain two or more frequencies at such nodes. The *CF* is computed using Eq. (6). Each rule can have both CF_1 and CF_2, when there exists a second choice according to the rulebase provided in Section 3.4.

Let r_{ki} be the ith rule for class C_k with frequency f_{ki}. Each rule is mapped using a single hidden node,

modeling the conjunct, that connects the attributes corresponding to the appropriate pattern class. Note that each class can have one or more rules. Therefore, one generates at least l hidden nodes, in a single hidden layer, for an l-class problem. The weight w^1_{ki}, between output node k (class C_k) and hidden node i (rule r_{ki}), is set at $f'_{ki} + \epsilon$, where ϵ is a small random number, $f'_{ki} = \frac{f_{ki}}{\sum_i f_{ki}} * CF^k$ and CF^k is the CF computed for class C_k by rule r_{ki}. The weight $w^0_{iA_j}$ between attribute A_j and hidden node i is clamped to $\frac{f'_{ki}}{Card(r_{ki})} + \epsilon$. Here $Card(r_{ki})$ indicates the number of features/attributes encountered along the traversal path from the root to the leaf containing the pattern corresponding to rule r_{ki} of class C_k. In other words, $Card(r_{ki})$ is the number of operands in the conjunct of rule r_{ki} for class C_k.

4.2. Example Demonstrating Network Mapping

Here we illustrate the scheme of mapping the decision tree obtained from fuzzy ID3 into the neural network with an example. Let the training set consist of 15 sample points, to be classified into three classes according to two continuous-valued features F_1 and F_2. These features are transformed to L_1, M_1, H_1, L_2, M_2, H_2 using Eq. (2). Let the sample decision tree be as shown in Fig. 1.

The root node is split at attribute M_1 by step 2 of the algorithm. There are 6, 4, 5 samples belonging to classes 1, 2, 3, respectively. The path $M_1 = 0$ results in a resolved leaf node for four patterns from class C_1. The path $M_1 = 1$ further splits on attribute L_1. In this manner, we obtain resolved leaf nodes for classes C_2 and C_3. All resolved nodes have $CF = 1$.

There is one unresolved node U, with two samples

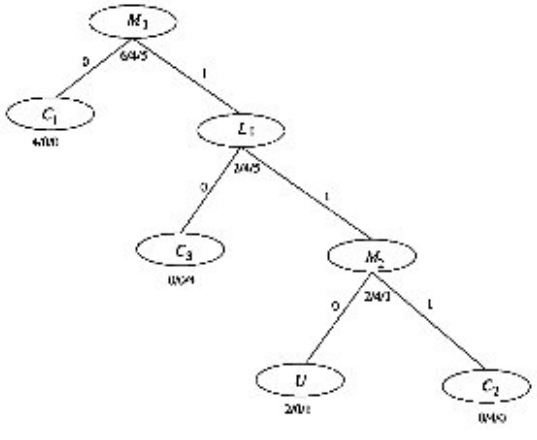

Fig. 1. Sample decision tree generated by fuzzy ID3.

Table 1. Class membership of sample data in unresolved node.

Sample	Class		
	C_1	C_2	C_3
d_1	0.80	0.02	0.60
d_2	0.70	0.13	0.50
d_3	0.16	0.10	0.90

from C_1, 0 samples from C_2 and one sample from C_3. Let the data points in 'U' be represented as d_1, d_2, d_3. Table 1 indicates the membership of these patterns to the three different classes by Eq. (4). Note that a pattern point can have finite membership to all three classes.

The CF_1 and CF_2 (sample level) values of d_1, d_2, d_3 corresponding to the classes with highest and second highest frequencies, using Eq. (6), are given in Table 2 (the classes being indicated within parentheses). As d_3 has $CF_1 = 0.835$ (≥ 0.8 by step 5(c) of the fuzzy ID3 algorithm), we clamp $CF_1 = 1$ and $CF_2 = 0$.

The aggregated CF for class C_1 is $(0.645 + 0.54 + 0)/3 = 0.365$, while that for class C_3 is $(0.395 + 0.291 + 1)/3 = 0.562$, using step 5(d) of the algorithm. As class C_3 has a higher CF value, we put CF_1 for node 'U' as 0.562 for class C_3 and $CF_2 = 0.365$ for class C_1 using step 5(e).

The rules corresponding to the decision tree of Fig. 1 are as follows:

(i) $M_1 \rightarrow C_1$; 4, 1.
(ii) $M_1 \wedge L_1 \wedge M_2 \rightarrow C_2$; 4, 1.
(iii) $M_1 \wedge \overline{L_1} \rightarrow C_3$; 4, 1.
(iv) $M_1 \wedge L_1 \wedge \overline{M_2} \rightarrow$ 3, 0.562, 0.365, C_3, C_1, 1, 2.

Note that

(a) the two numbers on the right-hand-side of rules (i)–(iii) indicate the number of pattern points satisfying that rule (frequency f_{ki}) and its confidence (CF), respectively;

Table 2. Confidence factor of sample data in unresolved node.

Sample	Confidence factor	
	CF_1	CF_2
d_1	0.645 (C_1)	0.395 (C_3)
d_2	0.54 (C_1)	0.291 (C_3)
d_3	0.835 (C_3)	−0.09 (C_1)

(b) in rule number (iv) (unresolved node 'U') the entities after '$\rightarrow$' indicate, respectively, the frequency of the rule, first and second confidence factors (CF_1, CF_2), classes corresponding to these confidence factors and the frequency of the samples belonging to these classes which satisfy this rule.

The corresponding mapped neural network is provided in Fig. 2.

Since there is a total of four rules, we use four hidden nodes in the hidden layer. As class C_1 has two rules with frequency 4 & confidence 1, and frequency 2 & confidence 0.365, the output node is connected to two hidden nodes with initial output layer weights of $f'_{11} = \frac{4}{4+2}*1 = \frac{2}{3}, f'_{14} = \frac{2}{4+2}*0.365 = 0.121$, respectively. Analogously, for class C_3 one has the initial output layer weights to the two hidden nodes as $\frac{4}{4+1}*1 = \frac{4}{5}, \frac{1}{4+1}*0.562 = 0.112$, respectively. Class C_2, having a single rule with $CF = 1$, is connected to the corresponding hidden node with a weight of value 1.

If there is only one output node connected with any hidden node, the weight on the hidden-output link percolates down to the weights in the input layer in proportion to the number of input attributes connected to that hidden node. Hence rule 1, with $\overline{M_1}$, provides a weight of $\frac{\frac{2}{3}}{-1}$ (taking account of the negative attribute $\overline{M_1}$).

When there is more than one output node connected to a single hidden unit (due to the presence of an unresolved node), then the maximum of all the weights on the links from that node to all connected output nodes is taken. This propagates to the weights in the input layer in proportion to the number of inputs attribute connected to that hidden node. For example, rule 4 shows that two output nodes denoting classes C_1 and C_3 will have connection to one common hidden node with weights 0.121 and 0.112, respectively. The maximum of these is 0.121. Hence, the weights in the corresponding input layer links (with three attributes) become $\frac{0.121}{3} = 0.04$, $\frac{0.121}{3} = 0.04$, $\frac{0.121}{-3} = -0.04$, respectively.

Finally, the connection weights of this knowledge encoded MLP are refined using the backpropagation algorithm.

5. Results

Here we present some results demonstrating the effectiveness of the knowledge-based MLP encoded using the fuzzy ID3 algorithm on *Vowel* data [17] (available at *http://www.isical.ac.in/~sushmita/patterns*), and the *Balance Scale* data [21]. As a comparison, the performance of the conventional MLP has also been provided.

The 871 Indian Telugu vowel sounds, collected by trained personnel, were uttered by three male speakers in the age group of 30 to 35 years, in a Consonant-Vowel-Consonant context. The details of the method are available in Pal and Dutta Majumder [18]. The data set has three features, F_1, F_2 and F_3, corresponding to the first, second and third vowel format frequencies obtained through spectrum analysis of the speech data. Note that the boundaries of the classes in the given data set are seen to be ill-defined (fuzzy). Figure 3 shows a 2D projection of

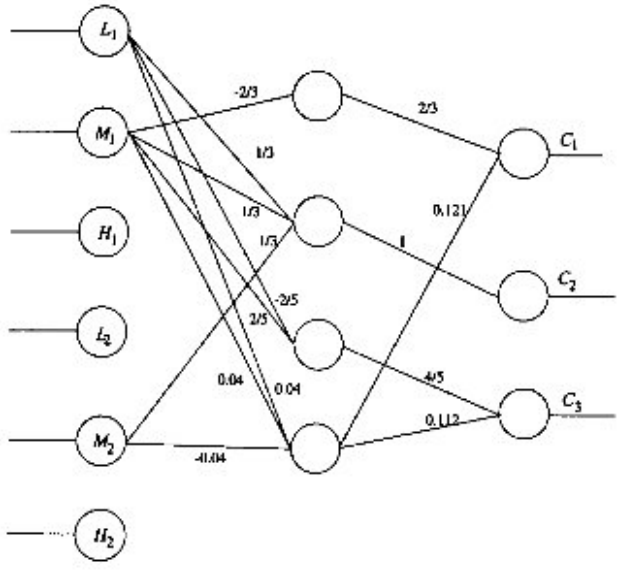

Fig. 2. Sample mapped network.

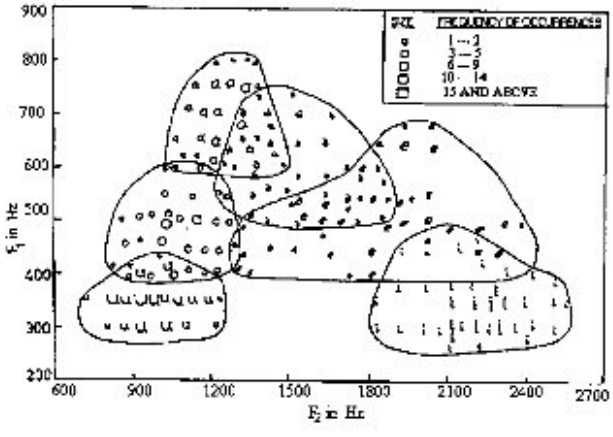

Fig. 3. Vowel diagram in $F_1 - F_2$ plane.

Table 3. Comparative performance with nine hidden nodes, using knowledge encoding with ID3 and conventional MLP, for *Vowel* data.

	Training								Testing						
	Set (%)	Scores (%)						# sweep	Scores (%)						
		1	2	3	4	5	6	Net		1	2	3	4	5	6	Net
I	10	50.0	85.7	93.7	85.7	68.4	100.	83.5	2550	59.0	81.7	85.9	92.0	71.2	86.5	80.9
	20	61.5	93.7	84.8	86.2	67.5	74.2	77.7	939	74.6	78.1	84.9	91.0	74.8	79.3	80.8
D	30	40.0	88.0	78.0	86.3	75.4	73.5	75.9	507	71.1	76.6	87.0	92.5	76.7	75.6	80.7
	40	40.7	76.4	82.0	84.7	76.5	80.2	77.0	780	68.9	83.6	86.7	91.3	76.1	78.0	81.4
3	50	48.5	81.4	84.7	90.5	69.6	82.0	78.2	451	67.6	84.8	85.1	92.2	81.9	75.8	82.2
M	10	50.0	85.7	87.5	85.7	73.7	94.1	82.2	2632	56.0	85.3	84.6	91.2	74.4	84.0	80.9
	20	53.8	93.7	84.8	86.2	67.5	71.4	76.5	1771	57.6	79.4	84.9	91.8	72.4	75.7	78.4
L	30	40.0	88.0	78.0	84.1	70.5	77.3	75.1	2491	59.6	79.7	86.1	90.6	76.0	74.8	79.3
	40	48.1	79.4	82.1	83.0	75.3	80.3	77.3	2560	75.6	83.6	86.7	89.1	74.6	78.0	81.2
P	50	51.4	81.4	84.7	90.5	67.6	80.9	77.8	32300	64.6	84.8	85.0	87.0	78.1	75.8	80.1

the 3D feature space of the six vowel classes (∂, *a*, *i*, *u*, *e*, *o*) in the $F_1 - F_2$ plane, for ease of depiction.

The *Balance scale* data [21] consists of 625 instances generated to model psychological experimental results. There are four numeric attributes corresponding to the *left weight, left distance, right weight* and *right distance*, and three output classes, viz. *tip right, tip left* and *balanced* (referred to as 1, 2 and 3 in the sequel).

Tables 3, 4 and 6 provide the performance of the networks initially encoded using rules generated by ID3 and fuzzy ID3 for *Vowel* data. Comparison is made with the conventional MLP, initialised using random weights. Different training set sizes 10%, 20%, 30%, 40%, 50% are used. Recognition scores (classwise and overall) for both training and test sets are provided, along with the number of training sweeps. Classes 1 to 6 indicate the six vowel classes ∂, *a*, *i*, *u*, *e*, *o*, respectively. The mean square error is used as the stopping criterion. The comparative performance for *Balance* data is provided in Table 5.

Tables 3 and 4 correspond to a modified version of the algorithm ID3. Fuzzy linguistic attribute values are considered. The frequency of the training pattern is taken into account while computing the initial weights. Table 3 considers nine rules for the six classes. This is mapped to nine hidden nodes. Table 4, on the other hand, considers 15 rules. Note that we consider only the rules with high frequency of occurrence in Table 3. It can be seen from Table 3 that the proposed algorithm gives better results, both in terms of recognition scores and number of train-

Table 4. Comparative performance with 15 hidden nodes, using knowledge encoding with ID3 and conventional MLP, for *Vowel* data.

	Training								Testing						
	Set (%)	Scores (%)						#sweep	Scores (%)						
		1	2	3	4	5	6	Net		1	2	3	4	5	6	Net
I	10	50.0	85.7	87.5	85.7	68.4	100.	82.2	970	66.7	85.3	84.6	89.0	70.7	83.4	80.4
	20	61.5	87.5	84.8	86.2	72.5	74.3	78.3	554	61.0	79.4	84.1	91.8	73.0	80.0	79.6
D	30	40.0	92.0	78.0	84.1	73.8	77.3	76.3	434	63.5	79.7	86.1	90.6	74	79.5	80.0
	40	44.4	79.4	82.1	81.3	75.3	80.2	76.7	377	64.4	85.4	86.7	90.2	74.6	79.8	81.0
3	50	48.6	79.1	82.3	89.2	70.6	83.1	77.8	267	64.9	84.8	85.1	89.6	79.0	75.8	80.8
M	10	50.0	85.7	87.5	85.7	68.4	100.	82.2	1256	59.1	82.9	84.6	89.0	74.5	85.9	80.9
	20	61.5	93.7	84.8	86.2	75.0	74.0	79.5	644	62.7	79.4	84.2	91.0	74.8	75.2	79.0
L	30	50.0	92.0	80.0	81.8	75.4	73.6	76.7	542	69.2	78.1	86.9	90.6	74.0	78.7	80.4
	40	40.7	79.4	82.0	84.7	72.8	80.2	76.4	575	66.7	85.4	87.6	90.2	73.0	78.9	80.8
P	50	62.9	81.4	84.7	90.5	69.6	80.9	79.2	444	75.6	84.8	86.2	89.6	76.1	75.8	81.2

Table 5. Comparative performance of knowledge encoded MLPs for *Balance* data.

Model	Train set (%)	Recognition scores (%)								No. of sweeps
		Training				Testing				
		1	2	3	Net	1	2	3	Net	
	10	97.9	57.7	97.4	94.2	89.2	12.8	88.4	82.8	312
ID3	20	98.0	36.7	96.5	93.0	93.7	24.6	92.6	87.7	252
	30	98.3	22.8	97.5	91.9	97.3	6.9	96.2	89.6	272
	40	97.8	16.6	97.0	91.1	96.9	16.8	95.6	90.3	271
	10	99.6	88.0	98.8	98.3	90.9	15.6	86.2	82.8	309
MLP	20	97.0	35.3	97.2	91.8	95.2	12.3	92.7	87.5	331
	30	97.6	26.8	96.5	91.8	96.2	14.3	91.9	87.8	307
	40	97.9	28.3	95.9	91.3	97.2	10.2	93.5	88.6	312
Neuro-Fuzzy	10	98.7	57.6	98.0	95.6	89.7	8.9	87.8	82.5	297
	20	98.7	41.0	97.5	93.9	93.8	9.3	94.2	87.5	252
ID3	30	97.4	28.4	97.0	91.8	96.4	11.8	95.0	88.9	242
[13]	40	98.1	24.3	97.0	92.0	96.8	11.5	95.5	90.0	237

Table 6. Comparative performance with 25 hidden nodes, using knowledge encoding with fuzzy ID3 and conventional MLP, for *Vowel* data.

	Set (%)	Training							# sweep	Testing						
		Scores (%)								Scores (%)						
		1	2	3	4	5	6	Net		1	2	3	4	5	6	Net
F	10	50.0	85.7	93.7	85.7	68.4	100.	83.5	739	65.1	84.1	85.3	91.2	75.0	77.9	80.6
U	20	67.8	80.8	83.4	92.6	77.2	76.5	80.5	389	69.5	78.1	84.2	91.0	77.8	77.9	80.7
Z	30	40.0	88.0	80.0	86.3	75.4	73.6	76.2	291	67.3	79.7	86.9	92.5	76.7	75.6	80.7
Z	40	40.7	76.4	82.1	83.0	77.8	80.2	77.0	251	71.1	85.4	85.7	90.2	77.0	78.0	81.6
ID3	50	51.4	81.4	84.7	89.1	72.5	83.1	79.2	180	67.6	84.8	83.9	88.3	83.8	75.8	81.7
M	10	50.0	85.7	87.5	85.7	68.4	100.	82.2	833	60.6	87.8	84.6	91.2	76.6	77.3	80.7
	20	61.5	93.7	84.8	86.2	72.5	74.3	79.6	445	61.0	78.0	85.6	91.8	74.2	77.9	79.5
L	30	40.0	88.0	80.0	84.1	75.4	77.4	76.7	341	71.1	78.1	86.9	90.6	71.9	77.9	79.9
	40	40.7	76.5	82.1	83.0	74.1	80.3	76.1	267	71.1	85.4	88.6	92.4	73.0	78.0	81.5
P	50	57.1	83.7	84.7	90.5	69.6	80.9	79.0	207	67.6	84.8	85.1	90.9	75.2	75.8	80.3

ing sweeps. This is because the encoding of prior knowledge results in a faster convergence. The gain in terms of recognition scores is however not that explicit in Table 4 involving 15 hidden nodes.

In Table 5 we use three hidden nodes during network encoding, mapping only the highest frequency rule corresponding to each class of the *Balance* data. The performance of the network encoded using the modified ID3 is compared with that of the conventional MLP (with no encoding) and the *Neuro-Fuzzy ID3* [13].

Table 6 demonstrates the performance of the network for *Vowel* data using fuzzy ID3, that incorporates fuzziness in the node level of the tree. The use of confidence factor and class membership resulted in the generation of 25 rules. Here unresolved nodes of the decision tree were also taken into account. The performance is always better in terms of the number of training sweeps. The gain in terms of recognition scores is not as evident as in Table 3. This leads us to the conclusion that the smaller the network, the more noticeable is the improvement of the proposed algorithm.

6. Conclusions and Discussion

A fuzzy version of the ID3 algorithm, for handling uncertainty in overlapping regions, is developed. A method to initially encode the connection weights of an MLP using the fuzzy ID3 algorithm has been described. Fuzziness is incorporated at the node

level to tackle unresolved nodes. The concepts of confidence and membership are used to arrive at a suitable decision at the node level. In the crisp case, this decision function reduces to the classical expression involving entropy. Rules are generated in linguistic form and are mapped onto a fuzzy neural network architecture. Each rule corresponds to a separate hidden node. The confidence factor and frequency of sample points are used to determine the initial connection weight values.

This type of knowledge encoding leads to faster convergence, as compared to the conventional MLP using random initial weights. Unlike the models reported elsewhere [9–11], the mapping scheme used here takes into account the frequency of the representative rules, in addition to handling uncertainty/fuzziness at various levels. It is observed that the effectiveness of the algorithm becomes more evident in a smaller network. In other words, the fewer the number of hidden nodes the greater is the improvement shown by our algorithm, in terms of both recognition scores and the number of training cycles/sweeps, as compared to that of conventional MLP involving random initial weights. This fact has also been established earlier by Banerjee et al. [5].

The rules extracted using fuzzy ID3 need to be evaluated in quantitative terms. The effect of missing attributes on the overall performance is to be investigated. Studies related to issues of convergence have been made experimentally. Some theoretical analysis is also required. Future research is envisaged along these directions.

References

1. Ripley BD. Pattern Recognition and Neural Networks. New York: Cambridge University Press, 1996
2. Quinlan JR. Induction on decision trees. Machine Learning 1986; 1: 81–106
3. Fu LM. Knowledge-based connectionism for revising domain theories. IEEE Trans Systems, Man and Cybern 1993; 23: 173–182
4. Towell GG, Shavlik JW. Knowledge-based artificial neural networks. Artif Intell 1994; 70: 119–165
5. Banerjee M, Mitra S, Pal SK. Rough fuzzy MLP: Knowledge encoding and classification. IEEE Trans Neural Networks 1998; 9: 1203–1216
6. Sankar A, Mammone RJ. Growing and pruning neural tree networks. IEEE Trans Computers 1993; 42: 291–299
7. Behnke S, Karayiannis NB. Competitive neural trees for pattern classification. IEEE Trans Neural Networks 1998; 9: 1352–1369
8. Cios KJ, Liu N. A machine learning method for generation of a neural network architecture: a continuous ID3 algorithm. IEEE Trans Neural Networks 1992; 3: 280–291
9. Sethi IK. Entropy nets: from decision trees to neural networks. Proc IEEE 1990; 78: 1605–1613
10. Ivanova I, Kubat M. Initialisation of neural networks by means of decision trees. Knowledge-Based Syst 1995; 8: 333–344
11. Setiono R, Leow WK. On mapping decision trees and neural networks. Knowledge-Based Syst 1999; 12: 95–99
12. Janikow CZ. Fuzzy decision trees. Issues and method. IEEE Trans Systems, Man and Cybern 1998; 28: 1–14
13. Ichihashi H, Shirai T, Nagasaka K, Miyoshi T. Neuro fuzzy ID3: A method of inducing fuzzy decision trees with linear programming for maximizing enrtopy and algebraic methods. Fuzzy Sets and Systems 1996; 81(1): 157–167
14. Maher P, Clair DS. Uncertain reasoning in an ID3 machine learning framework. Proc IEEE Int Conf on Fuzzy Systems, San Francisco, USA, 1993; 7–12
15. Runkler T, Roychowdhury S. Generating decision trees and membership functions by fuzzy clustering: Proc EUFIT, Aachen, Germany, 1999; 1–5
16. Pao YH. Adaptive Pattern Recognition and Neural Networks. Reading, MA: Addison-Wesley, 1989
17. Pal SK, Mitra S. Neuro-fuzzy Pattern Recognition: Methods in Soft Computing. New York: John Wiley, 1999
18. Pal SK, Dutta Majumder D. Fuzzy Mathematical Approach to Pattern Recognition. New York: John Wiley (Halsted Press), 1986
19. Pal SK, Mitra S. Multi-layer perceptron, fuzzy sets and classification. IEEE Trans Neural Networks 1992; 3: 683–697
20. Mandal DP, Murthy CA, Pal SK. Formulation of a multi-valued recognition system. IEEE Trans Systems, Man and Cybern 1992; 22: 607–620
21. Blake C, Merz C. UCI repository of machine learning databases. University of California, Irvine, Department of Information and Computer Sciences, http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998.