Indian J. pure appl. Math., **25** (1&2); 127-141, January & February 1994

A FUSION OF FUZZY SETS AND LAYERED NEURAL NETWORKS AT THE INPUT, OUTPUT AND NEURONAL LEVELS

SUSHMITA MITRA AND SANKAR K. PAL

Machine Intelligence Unit, Indian Statistical Institute
203, *B. T. Road, Calcutta* 700 035

(Accepted on 10 *May* 1993*)*

A way of incorporating the concepts of fuzzy sets into layered neural networks has been described. The input can be provided in quantitative or linguistic forms while the output may be modeled as membership values. Logical operators, *viz.*, *t-norm T* and *t-conorm S* involving *And* and *Or* neurons, are employed at the neuronal level, and the conventional *back propagation* algorithm is accordingly modified using various fuzzy implication operators. The usefulness of the model for classification is demonstrated on a set of vowel data by developing various methods along with their comparison. Effects of fuzzification at the input and output are also investigated.

1. INTRODUCTION

Artificial neural networks[1-9] are signal processing systems that try to emulate the human brain, *i.e.*, the behaviour of biological nervous systems, by providing a mathematical model of combination of numerous connected in a network. These models are reputed to have the following characteristics : adaptivity (the ability to adjust when given new information), speed (via massive parallelism), fault tolerance (to missing, confusing and/or noisy data) and optimality (as regards error rates in performance).

One of the most exciting developments in early days of pattern recognition is the multilayer perceptron (MLP)[1] which consists of multiple layers of simple, sigmoid processing elements (nodes) or neurons that interact using weighted connections. When used for supervised recognition, the input and output layers respectively. The network discovers the input-output mapping in a sequence of forward and backward passes through the training set, thereby efficiently modeling complex decision regions.

The theory of fuzzy subsets[9-12] provides an approximate and yet effective means for describing the characteristics of a system which is too complex or ill-defined to admit precise mathematical analysis. This theory can model the human thinking process and behaviour, and is reputed to handle (to a reasonable extent) uncertainities arising from deficiencies of information available from a situation; the deficiencies may result from incomplete, ill-defined, not fully reliable, vague and contradictory information. Therefore, we see that fuzzy set theoretic models try to mimic human reasoning and the architecture and information representation schemes of human brain.

Integration of the merits of fuzzy set theory and neural network theory promises to provide, to a great extent, more intelligent systems to handle real life processes. A large number of researchers[13-20] are now concentrating on exploiting these modern concepts in the field of pattern recognition and machine vision.

The present work is an attempt in this regard and described a logical version of the feedforward *multilayer perceptron* (MLP) using the concept of fuzzy sets at various stages. The classifier accepts input vector in terms of the linguistic properties *low, medium* and *high and provides output in terms of class membership values*[13]. *The hidden layer consists of* And *nodes while the output layer is made up of Or* nodes. Two cases of the conjugate pairs of *t-norm T* and *t-conorm S*[21], *viz., max-min* and *product-probabilistic sum,* are utilized to model the logical operations *And* and *Or*. The conventional *backpropagation algorithm* is modified by employing fuzzy implication operators in order to incorporate various amounts of mutual interaction during error propagation. Fig. 1 shows the proposed three-layered network. A heuristic for gradually decreasing the learning rate and momentum is used to help prevent oscillations of the mean square error in the process of convergence. Various algorithms have been described for pattern recognition problems using the above mentioned fusion strategy.

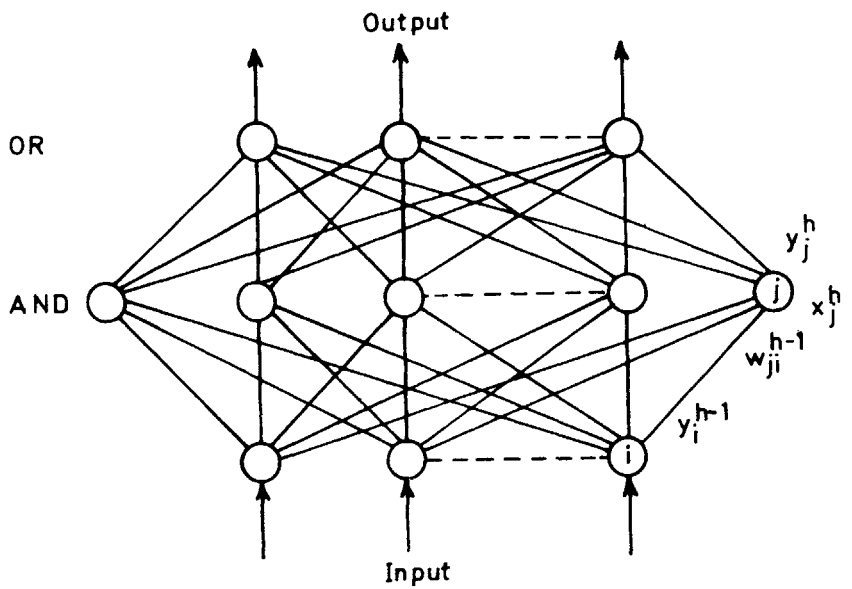

FIG. 1. A three layered neural network implementing AND and OR logic functions at successive layers.

Relation structures, realized by *max-min* or *min-max* and *product-probabilistic sum* operators, introduced into the proposed fuzzy model help in classifying patterns that exhibit a logical structure. Logical operators like *And* and *Or* are seen to ease the computational burden of the neural model to some extent, thereby enhancing its speed. (This will enable simpler hardware implementation for pattern recognition problems). The fuzzy input representation allows an n-dimensional feature space to be decomposed into 3^n overlapping sub-regions corresponding to the three primary

properties *viz., low, medium* and *high.* This enables the model to utilise more local information of the feature space and is found to be more suitable in handling overlapping pattern classes. Besides, the model can handle inputs presented in numerical as well as linguistic forms.

The effectiveness of the network (with various algorithms) is demonstrated on a set of speech data. Effects of fuzzification at the input as well as the output are also investigated.

In this connection it is worth mentioning that Pedrycz[22] used only the logical operators *max* and *min* to handle multi-class problems using two-layered nets and a different performance index. The inputs to the network consisted of all logical combinations (minterms) of the input variables, and the *Lukasiewicz* implication operator was used. Watanabe *et al*[23] also used *min-max* oprations but with two kinds of weight vectors and a different scheme of backpropagation for a three-layered network. Krishnapuram and Lee[24] used fuzzy aggregation connectives with compensatory behaviour (lying in the range between the two extremes, *viz., min* and *max*) as the activation functions of the neurons. A modified version of backpropagation was used to determine the proper type of the aggregation at each node and its parameters, given an approximate dependency structure of the network. Moreover, in all these cases (unlike in the proposed model), the input and output were nonfuzzy.

2. Multi-Layer Perceptron

Consider the layered network given in Fig. 1. The conventional MLP [1] is made up of simple neurons implementing the *weighted sum* and *sigmoid* functions (in place of the *And* and *Or* functions in Fig. 1). The total input x_j^{h+1} received by neuron j in layer $h + 1$ is defined as

$$x_j^{h+1} = \sum_i y_i^h \, w_{ji}^h - \theta_j^{h+1} \qquad \ldots (1)$$

where y_i^h is the state of the ith neuron in the preceding hth layer, w_{ji}^h is the weight of the connection from the ith neuron in layer h to the jth neuron in layer $h + 1$ and θ_j^{h+1} is the threshold of the jth neuron in layer $h + 1$.

The output of a neuron in any layer other than the input layer ($h > 0$) is a monotonic non-linear function of its total input and is given as

$$y_j^h = \frac{1}{1 + e^{-x_j^h}}. \qquad \ldots (2)$$

For nodes in the input layer

$$y_j^0 = x_j^0 \qquad \ldots (3)$$

where x_j^0 is the jth component of the input vector clamped at the input layer.

The Least Mean Square (LMS) error in output vectors, for a given network weight vector w, is defined as

$$E(w) = \frac{1}{2} \sum_{j,c} (y^H_{j,c}(w) - d_{j,c})^2 \quad \ldots (4)$$

where $y^H_{j,c}(w)$ is the state obtained for output node j in layer H in input-output case c and $d_{j,c}$ is its desired state specified by the teacher. The error $E(w)$ is minimized by the *back propagation* algorithm using gradient-descent. We start with any set of weights and repeatedly update each weight by an amount

$$\Delta w^h_{ji}(t) = -\varepsilon \frac{\partial E}{\partial w_{ji}} + \alpha \Delta w^h_{ji}(t-1) \quad \ldots (5)$$

where the learning rate ε controls the descent, $0 \leq \alpha \leq 1$ is the damping coefficient or momentum and t denotes the number of the iteration currently in progress.

3. Fuzzy MLP using Logical Operators

The proposed model consists of logical neurons employing conjugate pairs of *t-norms* T and *t-conorms* S in place of the *weighted sum* and *sigmoid* functions of the conventional MLP (described in Section II). The *back propagation* algorithm is modified to incorporate the logical operations in the error derivative term. The components of the input vector consist of the membership values to the overlapping partitions of linguistic properties *low, medium* and *high* corresponding to each input feature. During training, supervised learning is used to assign output membership values lying in the range [0, 1] to the training vectors. A heuristic for gradually decreasing the learning rate and the momentum is used to help avoid spurious local minima and usually prevent oscillations of the mean square error in the weight space, in the process of convergence to a minimum error solution.

A. *Input Vector Representation*

An n-dimensional pattern $F_i = [F_{i1}, F_{i2}, \ldots, F_{in}]$ can be represented as a $3n$-dimensional vector

$$F_i = [\mu_{low(F_{i1})}(F_i), \mu_{medium(F_{i1})}(F_i), \mu_{high(F_{i1})}(F_i), \ldots, \mu_{high(F_{in})}(F_i)] \quad \ldots (6)$$

where the μ values indicate the membership functions to the corresponding linguistic π-set[11] for each feature axis. Fig. 2 shows the overlapping structure of the three π-functions for a particular input feature F_j.. Thus an n-dimensional feature space is decomposed into 3^n overlapping sub-regions corresponding to the three primary properties. This enables the model to utilise more local information of the feature space for better handling of uncertainties arising from the overlapping regions.

When F_j is numerical we use the π-fuzzy sets[11] (in the one-dimensional form), with range [0, 1], given as

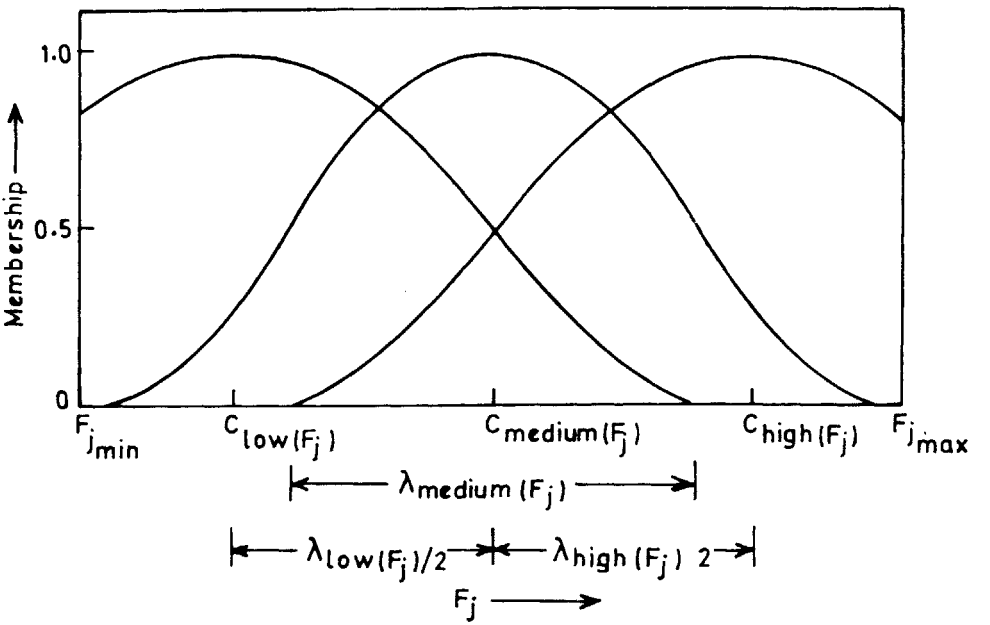

FIG. 2. Overlapping structure of the π-functions for the linguistic properties low, medium and high.

$$\pi(F_j; c, \lambda) = \begin{cases} 2\left(1 - \dfrac{|F_j - c|}{\lambda}\right)^2, & \text{for } \dfrac{\lambda}{2} \le |F_j - c| \le \lambda \\ 1 - 2\left(\dfrac{|F_j - c|}{\lambda}\right)^2, & \text{for } 0 \le |F_j - c| \le \dfrac{\lambda}{2} \\ 0, & \text{otherwise} \end{cases} \quad \ldots (7)$$

where $\lambda > 0$ is the radius of the π-function with c as the central point.

Let $F_{j\max}$ and $F_{j\min}$ denote the upper and lower bounds of the dynamic range of feature F_j in all L pattern points, considering numerical values only. Then for the three linguistic property sets we define[13]

$$\lambda_{medium}(F_i) = \frac{1}{2}(F_{j\max} - F_{j\min})$$

$$c_{medium}(F_j) = F_{j\min} + \lambda_{medium(F_j)} \quad \ldots (8)$$

$$\lambda_{low(F_j)} = \frac{1}{fdenom}(c_{medium(F_j)} - F_{j\min})$$

$$c_{low(F_j)} = c_{medium(F_j)} - 0.5 * \lambda_{low(F_j)} \quad \ldots (9)$$

$$\lambda_{high(F_j)} = \frac{1}{fdenom}(F_{j\max} - c_{medium(F_j)})$$

$$c_{high\,(F_j)} = c_{medium\,(F_j)} + 0.5 * \lambda_{high\,(F_j)} \quad \ldots (10)$$

where $0.5 \leq f\,denom \leq 1.0$ is a parameter controlling the extent of overlapping. The combination of the choices of λ and c ensures that for each quantitative input feature at least one of μ_{low}, μ_{medium} and μ_{high} is greater than 0.5.

When the input feature is linguistic, its membership values for the π-sets *low*, *medium* and *high* are quantified as[13]

$$\begin{aligned} low &\equiv \left\{ \frac{0.95}{L}, \frac{0.6}{M}, \frac{0.02}{H} \right\} \\ medium &\equiv \left\{ \frac{0.7}{L}, \frac{0.95}{M}, \frac{0.7}{H} \right\} \\ high &\equiv \left\{ \frac{0.02}{L}, \frac{0.6}{M}, \frac{0.95}{H} \right\} \end{aligned} \quad \ldots (11)$$

Details on input representation are available in Rosenfeld and Kim[13], Ruotvitz[14].

B. *Output Vector Representation*

In real-life problems, the data are generally ill-defined with overlapping or fuzzy class boundaries such that each pattern used in training may possess finite belongingness to more than one class. To model such data we clamp the desired membership values, lying in the range [0, 1], at the output nodes during training. When a separate set of test patterns is presented at the input layer, the output nodes are able to generate the class membership values of the patterns to the corresponding classes.

For an l-class problem domain, the membership of the ith pattern to class C_k, lying in the range [0, 1], is defined as

$$\mu_k(\boldsymbol{F}_i) = \frac{1}{1 + \left(\frac{z_i k}{F_d} \right)^{F_e}} \quad \ldots (12)$$

where z_{ik} is the weighted distance between the ith pattern and the mean of the kth class. It is such that as z_{ik} decreases, $\mu_k(\boldsymbol{F}_i)$ increases and *vice versa*. The positive constants F_d and F_e are the denominational and exponential fuzzy generators controlling the amount of fuzziness in this class-membership set.

Then for the ith input pattern we define the desired output of the jth output node as

$$d_j = \mu_j(\boldsymbol{F}_i) \quad \ldots (13)$$

During testing the output of the jth output neuron denotes the inferred membership values of a test pattern to the jth class.

Note that the desired output vector **d** has more *natural* information in terms of membership values with respect to the means of all the classes. This is unlike the teacher-specified *hard* labellings based on the training set[13]. This helps reduce

oscillations of the decision boundaries (during training) in case of *ambiguous* patterns. The test patterns are also classified based upon their *natural* membership values.

C. *T-Norm and T-Conorm*

We consider two special cases of the conjugate pair of *t-norm* T and *t-conorm* S[21] represented by the *And* and *Or* nodes at the hidden and output layers of Fig. 1 respectively. At the *And* nodes in the hidden layer we use $T(a, b) = a \wedge b$, while at the *Or* nodes in the output layer we have $S(a, b) = a \vee b$. The *max* and *min* operators are defined as

$$T^m(a, b) = \min(a, b)$$

$$S^m(a, b) = \max(a, b) \quad \ldots (14)$$

while the *product* and *probabilistic sum* operators are given as

$$T^p(a, b) = ab$$

$$S^p(a, b) = a + b - ab \quad \ldots (15)$$

such that $T^p(a, b) \leq T^m(a, b) \leq S^m(a, b) \leq S^p(a, b)$ and (T^p, S^p), (T^m, S^m) are the two special cases of the conjugate pair (T, S).

Let us consider the three-layered model as shown in Fig. 1 with $H = 2$. The output y_j^1 of the j^{th} neuron in the first layer is defined as

$$y_j^1 = {}_iT \left[S \left(y_i^0, w_{ji}^0 \right) \right]. \quad \ldots (16)$$

Here the input vector in the $3n$-dimensional space of eqn. (6), y_i^0 is the output of neuron i in the input layer as defined by eqn. 3) and w_{ji}^0 denotes the corresponding connection weight. The T operation is performed over all i S operation outputs corresponding to the neurons in the input layer.

Analogously, the output y_k^2 in the second layer is given as

$$y_k^2 = {}_jS \left[T \left(y_j^1, w_{kj}^1 \right) \right] \quad \ldots (17)$$

where the S operation is performed over all j T operation outputs corresponding to the neurons in the first layer.

Note that for the *probabilistic sum* operator S^p, the output is computed iteratively as

$$y_k^2 = S_{n-1}^p$$

such that

$$S_{j+1}^p = a_{j+2} + S_j^p - a_{j+2} S_j^p \quad \ldots (18)$$

for $j = 1, 2, \ldots, n - 2$, where

$$S_1^p = a_1 + a_2 - a_1 a_2$$

$$a_j = y_j^1 \; w_{kj}^1 . \qquad \ldots (19)$$

D. *Fuzzy Implication*

It is to be mentioned that the pair T^P and S^P of eqn. (15) are interactive and their results depend on the values of both the arguments. However the lattice operations T^m and S^m of eqn. (14) are completely noninteractive. Hence we use various implication operators to introduce different amounts of interaction during backpropagation of errors with the T^m and S^m operations.

For two variables X and Y, we have the *crisp* implication operator defined as

$$X \rightarrow Y = \| X \subset Y \| = \begin{cases} 1 & \text{if } X \leq Y \\ 0 & \text{otherwie} \end{cases} . \qquad \ldots (20)$$

This is observed to cause either *activity* or *disactivity* depending upon the individual conditions. To alleviate this problem, we introduce some interaction between the two variables using fuzzy concepts. A few fuzzy implication operators used to incorporate a degree of the amount of inclusion or containment are given below.

- Lukasiewicz

$$\| X \subset Y \| = T^m (1, 1 - X + Y) \qquad \ldots (21)$$

- Kleene-Dienes-Lukasiewicz

$$\| X \subset Y \| = 1 - X + XY \qquad \ldots (22)$$

- Kleene-Dienes

$$\| X \subset Y \| = S^m (1 - X, Y) \qquad \ldots (23)$$

- Early Zadeh

$$\| X \subset Y \| = S^m (T^m (X, Y), (1 - X)) \qquad \ldots (24)$$

where the S^m and T^m operators are as defined in eqn. (14). It is to be noted that these operators introduce various degrees of mutual interaction between the two variables and hence their choice may be application dependent.

E. *The Backpropagation Algorithm based on Logical Operations*

Use of logical operators in place of the more conventional *weighted sum* and *sigmoid* functions of the MLP necessitates modification of the derivatives involved in the traditional backpropagation scheme[1]. The LMS error E of eqn. (4) is minimized by the gradient-descent technique of eqn. (5) with the restriction that $0 \leq w_{ji}^h \leq 1$ for $0 \leq h \leq 2$. The error derivative $\frac{\partial E}{\partial w_{ji}}$ is computed as

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial y_j} \frac{\partial y_j^h}{\partial w_{ji}} . \qquad \ldots (25)$$

For the output layer ($h = H$) we substitute

$$\frac{\partial E}{\partial y_j} = y_j^H - d_j. \qquad \ldots (26)$$

In case of the hidden layer we use

$$\frac{\partial E}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \frac{\partial y_k^{h+1}}{\partial y_j} = \sum_k \left(y_k^H - d_k \right) \frac{\partial y_k^{h+1}}{\partial y_i} \qquad \ldots (27)$$

where neurons j and k lie in layers h and $h + 1$ respectively.

In order to evaluate the derivative $\frac{\partial y_j^h}{\partial w_{ji}}$ of eqn. (25), for $h > 0$, let us define

$$sm^h = \begin{cases} {}_{l \neq i}T\left[S\left(y_l^{h-1}, w_{jl}^{h-1} \right) \right] & \text{if } h = 1 \\ {}_{l \neq i}S\left[T\left(y_l^{h-1}, w_{jl}^{h-1} \right) \right] & \text{otherwise} \end{cases}$$

where the $T(S)$ operation at layer h is performed over all l $S(T)$ operation outputs from the neurons in the preceding layer $h - 1$, provided $l \neq i$, for $h = 1(2)$ respectively; also let

$$sm_i^h = \begin{cases} S\left(y_i^{h-1}, w_{ji}^{h-1} \right) & \text{if } h = 1 \\ T\left(y_i^{h-1}, w_{ji}^{h-1} \right) & \text{otherwise.} \end{cases} \qquad \ldots (29)$$

Using eqns. (16,17,28,29), we have

$$\frac{\partial y_j^h}{\partial w_{ji}} = \begin{cases} \frac{\partial}{\partial w_{ji}} \left[T\left(sm^h, sm_i^h \right) \right] & \text{if } h = 1 \\ \frac{\partial}{\partial w_{ji}} \left[S\left(sm^h, sm_i^h \right) \right] & \text{otherwise} \end{cases} \qquad \ldots (30)$$

where the *t-norm* T and *t-conorm* S are given by either of eqns. (14) or (15) in order to model the logical operators *And* and *Or*.

(i) The max and min operators

Using any of the implication operators from eqns. (20-24) and eqns. (28,29) in eqn. (30), we compute the derivative $\frac{\partial y_j^h}{\partial w_{ji}}$ for the conjugate pair (T^m, S^m) as

$$\frac{\partial y_j^h}{\partial w_{ji}} = \begin{cases} \| w_{ji}^{h-1} \subset y_i^{h-1} \| * \| sm^h \subset sm_i^h \| & \text{if } h = 2 \\ \| y_i^{h-1} \subset w_{ji}^{h-1} \| * \| sm_i^h \subset sm^h \| & \text{otherwise} \end{cases} \qquad \ldots (31)$$

where $h > 0$. Similarly, the sensitivity measure $\frac{\partial y_k^h}{\partial y_j}$ from the hth layer by eqn. (27), for $h = 2$, is evaluated as

$$\frac{\partial y_k^h}{\partial y_j} = || y_k^{h-1} \subset w_{jk}^{h-1} || * || sm^h \subset sm_k^h || \quad \ldots (32)$$

where sm^h and sm_k^h are given by eqns. (28,29) with k substituted for i.

(2) The product and probabilistic sum operators

On the other hand, for the conjugate pair (T^P, S^P) using eqns. (15), (18-19) and (28-30), we have

$$\frac{\partial y_j^h}{\partial w_{ji}} = \begin{cases} (1 - sm^h)\, y_i^{h-1} & \text{if } h = 2 \\ sm^h\,(1 - y_i^{h-1}) & \text{otherwise.} \end{cases} \quad \ldots (33)$$

Analogously, we compute the sensitivity measure as

$$\frac{\partial y_k^h}{\partial y_j} = (1 - sm^h)\, w_{jk}^{h-1}. \quad \ldots (34)$$

Substituting the values of $\frac{\partial y_j^h}{\partial w_{ji}}$ and $\frac{\partial y_k^h}{\partial y_j}$ from eqns. (31-32) or eqns. (33-34), as the case may be into eqns. (25, 27) enables one to evaluate the error derivative $\frac{\partial E}{\partial w_{ji}}$ of eqn. (5) and thereby update the connection weights during training. This constitutes the *back propagation* algorithm for the proposed network incorporating logical nodes.

3. Variation of ε and α

Here, the ε of eqn. (5) is gradually decreased in discrete steps, taking values from the chosen set {2, 1, 0.5, 0.3, 0.1, 0.05, 0.01, 0.005, 0.001}, while the momentum factor α is also decreased.

Let the various values of ε be indicated by $\varepsilon_0 = 2, \varepsilon_1 = 1, \ldots\ldots, \varepsilon_q = 0.001$ such that ε_i indicates the $(i+1)$th value of ε. Let $\alpha_0 = 0.9$ and $\alpha_1 = \alpha_2 = \ldots. = \alpha_q = 0.5$. We use

$$i = \begin{cases} i + 1 & \text{if } mse\,(nt - kn) - mse(nt) < \delta \\ i & \text{otherwise} \end{cases} \quad \ldots (35)$$

where $i = 0$ initially, $| \varepsilon | = q + 1$ and $0 < \delta \leq 0.0001$. Note that $mse(nt)$ is the mean square error at the end of the nt^{th} sweep through the training set and kn is a positive integer such that mse is sampled at intervals of kn sweeps. The process is terminated when $i > q$ and $\varepsilon_q = 0.001$.

At convergence, the proposed logical version of the MLP is expected to have learned the relationship between the input-output representations of the training patterns. This is the desired pattern classifier.

4. Implementation and Results

The proposed three-layered logical neural net was used for developing two models (viz. P and KDL) for classification of vowel data consisting of a set of 871 Indian Telugu vowel sounds[11] uttered in a Consonant-Vowel-Consonant context by three male speakers in the age group of 30 to 35 years. The data set had three features; F_1, F_2 and F_3 corresponding to the first, second and third formant frequencies obtained through spectrum analysis of the speech data. Fig. 3 shows a 2D projection of the 3D feature space of the six vowel classes (∂, a, i, u, e, o) in the $F_1 - F_2$ plane. The input was represented in the $3n$-dimensional linguistic form of eqn. (6). The network was trained using 10% samples from each representative pattern class of the data set (for the results of Tables 1-3). We selected *fdenom* = 0.8 in eqns. (9-10), $F_d = 2, F_e = 2$ in eqn. (12) and *kn* = 10, δ = 0.0001 in eqn. (35).

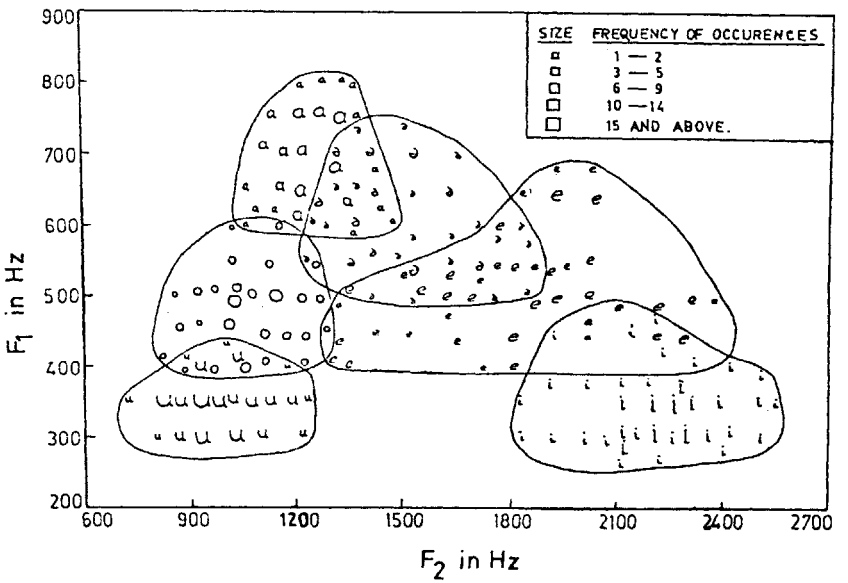

FIG. 3. Vowel diagram in the $F_1 - F_2$ plane.

We use two measures of classification performance for the training set. The output, after a number of updating steps, is considered a *perfect match* if the value of each output neuron y_j^H is within a margin of 0.1 of the desired membership value d_j. This is a *stricter* criterion than the *best match*, where we test whether the j^{th} neuron output y_j^H has the maximum activation when the j^{th} component d_j of the desired output vector also has the highest value, provided $y_j^H > 0.5$. Note that the desired fuzzy output vector $\mathbf{d}$ is dependent on the class means and hence on the training set selected. Therefore, a higher value of the *perfect match* p is more

indicative of good performance as compared to the maximum-activation-related *best match b* which is more dependent on the choice of the training patterns.

In Tables 1-3, *mse*, *p* and *b* refer to the training set (10 percent samples) well mse_t and *overall best score t* are indicative of the test set (remaining 90% samples). Besides, the criterion for *b* is stricter than that for *t* in the sense that *t* denotes simply the best match while *b* indicates the best match provided that $y_j^H > 0.5$ when $d_j > 0.5$. Hence sometimes we observe *t* to be greater than *b*. Note that *KDL* refers to the *Kleene-Dienes-Lukasiewicz* operator by eqn. (22) (used during backpropagation of error) for the conjugate pair (T^m, S^m) of eqn. (14) while *P* refers to the logical operators (T^P, S^P) of eqn. (15).

Table 1 demonstrates a comparison of the performance of the proposed logical models *P* and *KDL* on the vowel data (using one hidden layer with *m* hidden nodes). It is observed that the model *KDL* fared poorer as compared to the model *P*. This is because of the fact that the max and min operations of *KDL* caused the different *input-connection weight* combinations to be less interactive, with only the maximum or minimum term(s) controlling the neuron activation(s). The *product* and *probabilistic sum* operators, being more *co-operative*, yielded better results. Because of the same reason, if we replace the logical operator by the original sigmoidal function the results (shown as model *O*) improve. The use of logical operators seemed to cause deterioration in the classification efficiency of the neural model, perhaps due to the inherent loss of some information.

Effects of fuzzification at the input and output have been investigated in detail using the logical model *P* with $m = 20$ hidden nodes. The amount of overlapping between the linguistic properties *low, medium* and *high (at the input) was varied by altering the radius of the π-function corresponding to the linguistic set* medium. Let $\lambda_{medium} = fnos * \lambda_{medium}$ where *fnos* = 1 for the value of λ_{medium} given by eqn. (8). As we decrease *fnos*, the radius λ_{medium} decreases around c_{medium} such that ultimately the linguistic property *medium* covers an insignificant portion of the corresponding feature axis. Thereby its role in overcoming any confusion in generating the decision surface along certain regions of the feature space diminishes. On the other hand, as we increase *fnos* the radius λ_{medium} increases around c_{medium} such that the amount of overlapping between the π-functions increases. In general, there is no pronounced change in the output performance for a wide variation in input overlapping as shown in Table 2. However, very large amount of overlapping (*fnos* ≥ 1.2) among the linguistic properties of the input feature, as expected, leads to poorer recognition scores (especially for class *a*).

Table 3 demonstrates the variation in performance of the above-mentioned logical model *P* with different amounts of fuzziness at the output. Here F_d and F_e of eqn. (12) were varied as shown. It is to be noted that higher values of F_e and lower values of F_d help in enhancing the contrast among the output membership values and generally lead to a higher recognition score. However too high contrast ($F_d < 2$) resulted in poor performance. The logical model *P* has good overall performance for $F_d = 2$ with $F_e = 2, 4$ and $F_d = 3$ with $F_e = 4$ (corresponding to high contrast).

Table 1 Comparative Performance of the Fuzzy Logical Models

Model		*P*			*KDL*			*O*		
Nodes m =		19	20	21	19	20	21	19	20	21
perf	p (%)	28.3	34.2	25.9	2.4	1.2	1.2	82.4	70.6	67.1
best	b (%)	84.7	78.9	75.3	28.3	18.9	28.3	78.9	82.4	84.7
mse		.006	.007	.008	.032	.039	.034	.001	.002	.002
T	∂ (%)	66.1	76.8	32.3	46.3	100	67.6	82.7	85.3	95.1
e	a (%)	91.5	46.5	98.8	0.0	0.0	0.0	97.9	99.0	77.4
s	i (%)	98.1	100	89.9	83.5	99.2	55.9	100	100	98.4
t	u (%)	99.2	65.9	92.1	87.9	69.1	45.4	88.2	99.4	95.9
s	e (%)	81.0	85.3	89.9	60.3	40.9	74.2	95.3	92.1	91.9
e	o (%)	92.0	91.3	99.3	10.3	0.0	48.9	92.8	100.0	95.2
t	t (%)	88.1	84.1	90.5	50.1	48.7	52.4	93.7	96.5	92.8
	mse_t	.009	.009	.01	.034	.037	.035	.003	.004	.007

Table 2 Effect of Fuzzification at Input with Model *P*

fnos =	0.5	0.6	0.7	0.8	0.9	1.0	1.1
perfect p (%)	10.6	15.3	18.9	22.4	30.6	34.2	34.2
best b (%)	71.8	71.8	75.3	80.0	77.7	78.9	77.7
mse	.011	.011	.09	.009	.008	.007	.008
Overall t (%)	82.5	81.1	85.2	85.8	84.1	84.1	87.1
mse_t	.014	.013	.012	.01	.009	.009	.009

Table 3 Efect of Fuzzification at Output with model *P*

F_d =	2			3		4		5		
F_e =	1	2	4	2	4	2	4	1	2	4
perfect p (%)	34.2	34.2	25.9	25.9	27.1	55.3	40.0	43.6	60.0	50.6
best b (%)	69.5	78.9	80.0	83.6	84.7	85.9	70.6	71.8	71.8	61.2
mse	.006	.007	.014	.008	.01	.005	.007	.004	.004	.006
Overall t (%)	77.6	84.1	86.1	81.0	83.9	82.1	80.2	75.2	78.1	72.2
mse_t	.007	.009	.015	.009	.014	.005	.009	.004	.003	.006

It is revealed under investigation that this choice of parameter values enables the membership function curves of the various pattern classes to represent the dynamic range of the given pattern points more suitably.

5. CONCLUSIONS

A way of integrating fuzzy set theory at the input, output and neural levels of layered neural networks has been provided. Two models were developed based on *product* and *probabilistic* sum operators, and *max* and *min* operations. The *product*

and *probabilistic sum* operators are found to perform better than the *max* and *min* operators due to the involvement of greater co-operative interaction among the *input-connection weight* combinations. Incorporation of fuzziness in the input and output representations is seen to improve the performance of the conventional MLP. The network is seen to be robust with respect to variations in input overlapping. Higher contrast in the output membership values usually resulted in better performance.

The use of *And* and *Or* neurons in the MLP is expected to decrease the amount of computations required. The hardware implementation of logical neurons might also be easier.

The logical model should be capable of efficient rule generation (expressed as disjunction of conjunctive clauses) and has an useful application in expert system design.

ACKNOWLEDGEMENT

The authors gratefully acknowledge Mr A. Ghosh for his assistance in preparing the final version of the manuscript.

REFERENCES

1. R. P. Lippmann, "An introduction to computing with neural nets", *IEEE Acoustics, Speech and Signal Processing Magazine*, vol. 61, pp.4-22, 1987.
2. J. Dayhoff, *Neural Network Architectures : An Introduction*. New York: Van Nostrand Reinhold, 1990.
3. Y. H. Pao, *Adaptive Pattern recognition and Neural Networks*. Reading, MA: Addison Wesley, 1989.
4. A. Ghosh, N. R. Pal and S. K. Pal, "Image segmentation using a neural network," *Biological Cybernetics*, vol. 66, no. 2, pp. 151-158, 1991.
5. A. Ghosh and S. K. Pal, "Neural network, self-organization and object extraction," *Pattern Recognition Letters*, vol. 13, no. 5, pp. 387-397, 1992.
6. A. Ghosh, N. R. Pal, and S. K Pal, "Object background classification using Hopfield type neural network," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 6, no. 5, pp. 989-1008, 1992.
7. J. Basak, C. A. Murthy, S. Chaudhury, and D. D. Majumder, "A connectionist model for category perception : theory and implementation," *IEEE Trans. on Neural Networks*, Vol. 4, No. 2, pp. 257-69, March 1993.
8. J. Basak, S. Chaudhury, S. K Pal, and D. D. Majumder, "Matching of structural shape descriptions with Hopfield net," *International J. of Pattern Recognition and Artificial Intelligence*, vol. 7, pp. 377-404 1993.
9. J. C. Bezdek and S. K. Pal, eds., *Fuzzy Models for Pattern Recognition: Methods that Search for Structures in Data*. NY: IEEE Press, 1992.
10. G. J. Klir and T. Folger, *Fuzzy Sets, Uncertainty and Information*. Reading, MA: Addison Wesley, 1989.
11. S. K. Pal and D. Dutta Majumder, *Fuzzy Mathematical Approach to Pattern Recognition*. New York: Wiley (Halsted Press), 1986.
12. A. Kandel, *Fuzzy Mathematical Techniques with Applications*. Reading, MA: Addison-Wesley, 1986.
13. S. K. Pal and S. Mitra, "Multilayer perceptron, fuzzy sets and classification," *IEEE Transactions on Neural networks*, vol. 3, no. 5, pp. 683-697, 1992.

14. S. Mitra and S. K. Pal, "Self-organizing neural network as a fuzzy classifier," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 2, 1994, (to appear).
15. A. Ghosh, N. R. Pal, and S. K. Pal, "Self-organization for object extraction using multilayer neural network and fuzziness measures," *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 1, pp. 54-68, 1993.
16. S. Mitra and S. K. Pal, "Neuro-fuzzy expert systems : overview with a case study," in *Fuzzy Reasoning in Information, Decision and Control Systems*, (S. G. Tzafestas and A. N. Venetsanopoulos, eds.), Boston: Kluwer Academic Press, (to appear).
17. S. K. Pal and A. Ghosh, "Neuro-fuzzy image processing : relevance and feasibility," in *Neural and Fuzzy Systems: The Emerging Science of Intelligence and Computing*, New York: SPIE Press, 1993, (to appear).
18. *Proc. of First International Conference on Fuzzy Logic and Neural Networks (IIZUKA 90)*, (Iizuka, Fukuoka, Japan), July 1990.
19. *Proc. of Second Internatinal Conference on Fuzzy Logic and Neural Networks (IIZUKA 92)*, (Iizuka, Fukuoka, Japan), July 1992.
20. *Proc. of first IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, (SanDiego, USA), March 1992.
21. I. B. Turksen, "Interval valued fuzzy sets based on normal forms," *Fuzzy Sets and Systems*, vol. 20, pp. 191-210, 1986.
22. W. Pedrycz, "Neurocomputations in relational systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 289-297, 1991.
23. T. Watanabe, M. Matsumoto, and T. Hasegawa, "A layered neural network model using logic neurons," in *Proceedings of the 1990 International Conference on Fuzzy Logic and Neural Networks Iizuka*, (Japan), pp. 675-678, 1990.
24. R. Krishnapuram and J. Lee, "Fuzzy-set-based hierarchical networks for information fusion in computer vision," *Neural Networks*, vol. 5, pp. 335-350, 1992.