Fuzzy Self-Organization, Inferencing, and Rule Generation

Sushmita Mitra and Sankar K. Pal, *Fellow, IEEE*

Abstract— **A connectionist inferencing network, based on a fuzzy version of Kohonen's model already developed by the authors, is proposed. It is capable of handling uncertainty and/or impreciseness in the input representation provided in quantitative, linguistic and/or set forms. The output class membership value of an input pattern is inferred by the trained network. A measure of certainty expressing confidence in the decision is also defined. The model is capable of querying the user for the *more important* input feature information, if required, in case of partial inputs. Justification for an inferred decision may be produced in rule form, when so desired by the user. The connection weight magnitudes of the *trained* neural network are utilized in every stage of the proposed inferencing procedure. The antecedent and consequent parts of the justificatory rules are provided in *natural* forms. The effectiveness of the algorithm is tested on the vowel recognition problem and on two sets of artificially generated nonconvex pattern classes.**

I. Introduction

ARTIFICIAL neural networks or connectionist models, [1]–[4], are massively parallel interconnections of simple neurons that function as a collective system. An advantage of neural nets lies in their high computation rates provided by massive parallelism, so that real-time processing of huge data sets becomes feasible with proper hardware. Information is encoded among the various connection weights in a distributed manner. The utility of fuzzy sets, [5]–[7], lies in their capability, to a reasonable extent, in modeling *uncertain* or ambiguous data so often encountered in real life.

We consider an application of the fuzzy extension to Kohonen's self-organizing neural network [8] or inferencing and rule generation. The model is expected to be capable of handling uncertainty and/or impreciseness in the input representation, inferring the output class membership value for complete and/or partial inputs along with a certainty measure, querying the user for the *more essential* input information and providing justification (in the form of rules) for any inferred decision. The input can be in quantitative, linguistic or set forms or a combination of these. The fuzzy Kohonen's model [8] functions as a partially supervised classifier. During self-organization, the input vector also includes some contextual information (with lower weightage) regarding the finite output membership of the pattern to one or more class(es). The input and output representations and the weight updating mechanism of the said model [8] are described in Section II.

Manuscript received January 21, 1994; revised July 25, 1995.

The authors are with Machine Intelligence Unit, Indian Statistical Institute, Calcutta-700035, India.

Publisher Item Identifier S 1083-4427(96)05358-1.

The inferencing and rule generating capabilities of the fuzzy Kohonen's model (on an unknown test set) are explained in detail in Section III. At the end of the learning phase (self oraganization and calibration), the set of connection weights of the neural model may be said to constitute the *knowledge base* for the classification problem in hand. In the testing phase the network infers the *most likely* output class for unknown test samples using its knowledge base. The magnitude of the connection weight from an input feature component to a neuron output is used to determine the importance of the corresponding attribute. When partial information about a test vector is presented at the input, the model is able to infer its category if the *more essential* feature information is present. Otherwise, the system asks the user for *relevant* information in the order of their relative importance. A measure of certainty expressing belief in the decision is also defined. When asked by the user, the network is capable of justifying its decision in *rule-form* by reasoning backward and using its connection weights and the value of the certainty measure. The antecedent and consequent parts of the rules are generated in linguistic and *natural* terms.

A few of the other existing approaches for neurofuzzy inferencing and classification include neural net driven fuzzy reasoning [9], the MLP-based approach using backpropagation [10], the connectionist expert model [11], the use of logical operators for pattern classification [12], [13], the use of *fuzzy* aggregation connectives [14], learning from fuzzy If-Then rules [15], and fuzzy MLP for classification and rule generation [16]–[18]. Note that these neuro-fuzzy approaches are all based on layered networks using fully supervised learning. Other work on the integration of fuzzy sets and Kohonen's model for clustering are reported in [19], [20]. Our proposed investigation, on the other hand, is based on the fuzzy version of Kohonen's self-organizing model [8] which acts as a partially supervised classifier. This model is extended here for studying its inferencing, querying and rule generation abilities with unknown test data. The effectiveness of the proposed algorithm is demonstrated in Section IV on the vowel recognition problem and on two sets of artificially generated linearly nonseparable, nonconvex pattern classes.

II. Kohonen's Network as a Fuzzy Classifier

Here we present a brief discussion on the fuzzy version of Kohonen's model [8] that is capable of partially supervised classification of fuzzy patterns. We consider a single layer two-dimensional (2-D) rectangular array of neurons with short range lateral feedback interconnections between neighboring

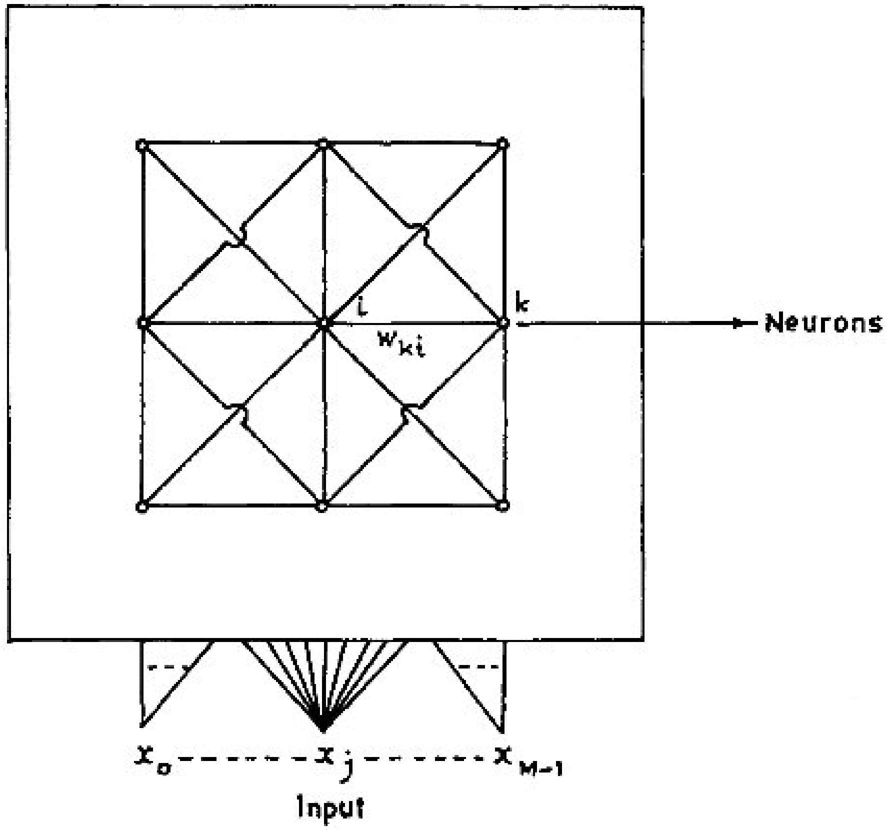

Fig. 1. Kohonen's neural network model [1].

units. In the first stage a set of training data is used by the network to initially self organize the connection weights and then *calibrate* the output space. After a number of sweeps through the training set the output space becomes ordered, as determined by the index of disorder. Then the output space is calibrated with respect to the pattern classes.

Consider the self-organizing network given in Fig. 1. Let M input signals be simultaneously incident on each of an $N \times N$ array of neurons. The output of the ith neuron is defined as

$$\eta_i(t) = \sigma\left[[\mathbf{m}_i(t)]^T \mathbf{x}(t) + \sum_{k \in S_i} w_{ki}\eta_k(t - \triangle t)\right] \quad (1)$$

where $\mathbf{x}$ is the M-dimensional input vector incident on it along the connection weight vector $\mathbf{m}_i$, k belongs to the subset S_i of neurons having interconnections with the ith neuron, w_{ki} denotes the fixed feedback coupling between the kth and ith neurons, $\sigma[\cdot]$ is a suitable sigmoidal output function, t denotes a discrete time index and T stands for the transpose.

If the best match between vectors $\mathbf{m}_i$ and $\mathbf{x}$ occurs at neuron c, then we have

$$\|\mathbf{x} - \mathbf{m}_c\| = \min_i \|\mathbf{x} - \mathbf{m}_i\|, \quad i = 1, 2, \ldots, N^2 \quad (2)$$

where $\|\cdot\|$ indicates the Euclidean norm.

A. The Input Vector

The input to the neural network model consists of two portions. In addition to the input feature representation in linguistic form, there is also some contextual information regarding the fuzzy class membership of each pattern used as training data during the self organization of the network.

Feature Information: The π-function, lying in the range $[0, 1]$, with $\mathbf{x} \in \mathbb{R}^n$ is defined as [21]

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

where $\lambda > 0$ is the radius of the π-function with $\mathbf{c}$ as the central point at which $\pi(\mathbf{c}; \mathbf{c}, \lambda) = 1$. In the fuzzy neural network model we use the π-function (in the one-dimensional (1-D) form) to assign membership values for the input features.

Each input feature F_j is expressed in terms of membership values indicating a degree of belonging to each of the linguistic properties *low, medium,* and *high.* An n-dimensional pattern $\mathbf{X}_i = [F_{i1}, F_{i2}, \ldots, F_{in}]$ is represented as a $3n$-dimensional vector

$$\mathbf{X}_i = [\mu_{\text{low}(F_{i1})}(\mathbf{X}_i), \mu_{\text{medium}(F_{i1})}(\mathbf{X}_i), \mu_{\text{high}(F_{i1})}(\mathbf{X}_i), \ldots, \mu_{\text{high}(F_{in})}(\mathbf{X}_i)] \quad (4)$$

where the μ value indicates the membership to the corresponding linguistic π-set (*low/medium/high*) along each feature axis F_{ij} for pattern $\mathbf{X}_i$.

When input feature F_j is linguistic, its membership values for the π-sets *low, medium*, and *high* in (4) are quantified as

$$\begin{aligned} low &= \left\{\frac{0.95}{L}, \frac{0.6}{M}, \frac{0.02}{H}\right\} \\ medium &= \left\{\frac{0.7}{L}, \frac{0.95}{M}, \frac{0.7}{H}\right\} \\ high &= \left\{\frac{0.02}{L}, \frac{0.6}{M}, \frac{0.95}{H}\right\}. \end{aligned} \quad (5)$$

For example, if F_j is *low*, then its membership values for the primary properties *low (L), medium (M)*, and *high (H)* are 0.95, 0.6, and 0.02, respectively. Similar are the cases if F_j is *medium* or *high*.

When F_j is numerical, we use the π fuzzy set of (3) with appropriate c and λ. Depending on the numeric/linguistic nature of the input feature F_j, (3) or (5) is used to convert F_j to its three-dimensional (3-D) form given by (4). The choice of the λ's and c's for each of the linguistic properties *low, medium*, and *high* are the same as that reported in [8].

The representation of input in terms of π-sets *low, medium*, and *high* also enables the network to accept imprecise/vague features F_j in various forms, viz., F_j is *about* 500, F_j is *between 400 and 500*, F_j is *low, medium, very low, more or less low* or F_j is *missing*, etc. In these cases F_j needs to be transformed into a 3-D vector consisting of membership values corresponding to the three primary properties. A heuristic method for the determination of these membership values is discussed in detail in Section III-A.

Class Information in Contextual Form: To model real-life data, with finite belongingness to more than one class, we incorporate some contextual information regarding class membership as part of the input vector. However during self-organization this part of the input vector is assigned a lower weight so that the linguistic properties dominate in determining the ordering of the output space. During calibration we use the contextual class membership information part of the input vector only, for determining the *hard* labeling of the output space. A separate fuzzy partitioning, that allows scope for producing overlapping clusters, is also introduced. The significance of the contextual class information part in providing partial supervision has been described in [8].

The pattern $\mathbf{X}_i$ is considered to be presented as a concatenation of the linguistic properties in (4) and the contextual information regarding class membership. Let the input vector be expressed as

$$\mathbf{x} = [\mathbf{x}', \mathbf{x}'']^T = [\mathbf{x}', 0]^T + [0, \mathbf{x}'']^T \quad (6)$$

where $\mathbf{x}'$ (attribute part) contains the linguistic information in the $3n$-dimensional space of (4) and $\mathbf{x}''$ (symbol part) covers the class membership information in an l-D space for an l-class problem domain. So the input vector $\mathbf{x}$ lies in an $(3n + l)$-dimensional space. Both $\mathbf{x}'$ and $\mathbf{x}''$ are expressed as membership values.

Here we consider the definition of $\mathbf{x}''$. The membership of the ith pattern to Class C_k is defined as [6]

$$\mu_k(\mathbf{X}_i) = \frac{1}{1 + \left(\frac{z_{ik}}{F_d}\right)^{F_e}} \quad (7)$$

where $0 \leq \mu_k(\mathbf{X}_i) \leq 1$, z_{ik} is the weighted distance of the ith pattern from the kth class and 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. However, when the pattern classes are known to be nonfuzzy (from the training set), z_{ik} may be set to 0 for the class to which the pattern belongs and to *infinity* for the remaining classes, so that $\mu_k(\mathbf{X}_i) \in \{0, 1\}$.

For the ith input pattern we define

$$\mathbf{x}'' = s * [\mu_1(\mathbf{X}_i), \cdots, \mu_l(\mathbf{X}_i)]^T \quad (8)$$

where $0 < s \leq 1$ is the scaling factor. To ensure that the norm of the linguistic part $\mathbf{x}'$ predominates over that of the class membership part $\mathbf{x}''$ in (6) during self-organization, we choose $s < 0.5$.

During calibration of the output space the input vector chosen is $\mathbf{x} = [0, \mathbf{x}'']$, where $\mathbf{x}''$ is given by (8), such that

$$\mu_q(\mathbf{X}_i) = \begin{cases} 1, & \text{if } q = k \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

for $k \in \{1, \ldots, l\}$ and $s = 1$. The N^2 neuron outputs η_i are calibrated w.r.t. the l classes. The resulting *hard* (labeled) partitioning of the output space may be used to qualitatively assess the topological ordering of the pattern classes w.r.t. the input feature space. We also consider a *fuzzy* partitioning of the output space.

B. The Algorithm

Consider an $N \times N$ array of neurons such that the output of the ith neuron is given by (1), with the subset S_i of neurons being defined as its r-neighborhood N_r where $0 \leq r \leq 3$.

Weight Updating: Initially the components of the $\mathbf{m}_i$'s may be set to small random values lying in the range [0, 1]. Let the best match between vectors $\mathbf{m}_i$ and $\mathbf{x}$, selected using (2), occur at neuron c. The weight updating expression is stated as

$$\mathbf{m}_i(t+1) = \begin{cases} \mathbf{m}_i(t) + h_{ci} * (\mathbf{x}(t) - \mathbf{m}_i(t)), & \text{for } i \in N_r,\ r = 0, 1, \cdots, 3 \\ \mathbf{m}_i(t), & \text{otherwise} \end{cases} \quad (10)$$

where N_r describes a r neighborhood around neuron c such that r decreases with time. Here the gain factor h_{ci} [8] is considered to be *bell-shaped* like the π-function, such that $|h_{ci}|$ is the largest when $i = c$ and gradually decreases to zero with increasing distance from c. Besides, $|h_{ci}|$ also decays with time.

Index of Disorder: An index of disorder D may be defined to provide a measure of this ordering. Let *msd* denote the mean square distance between the input vector and the weight vectors in the r neighborhood of neuron c. We define

$$msd = \frac{1}{|trainset|} \sum_{\mathbf{x} \in trainset} \times \left[\sum_{r=0}^{3} \left\{ \left(\frac{1}{|N_r|} \sum_{i \in N_r} \|\mathbf{x} - \mathbf{m}_i\|^2 \right) * (1 - r * f) \right\} \right] \quad (11)$$

where $|trainset|$ refers to the number of input pattern vectors in the training set.

The expression for the index of disorder is given as

$$D = msd(nt - kn) - msd(nt) \quad (12)$$

where $msd(nt)$ denotes the mean square distance by (11) at the end of the ntth sweep through the training set and kn is a suitable positive integer denoting the step size. Further, D is calculated relative to an interval of kn sweeps. Initially $ncnt$ is set to 1. Then

$$ncnt = \begin{cases} ncnt + 1, & \text{if } D < \delta \\ ncnt, & \text{otherwise} \end{cases} \quad (13)$$

where $0 < \delta \leq 0.001$. The self organization process is terminated when $ncnt > 3$.

Partitioning During Calibration: After self-organization is complete, we go on to the calibration of the neuronal output space. During calibration the input vector $\mathbf{x} = [0, \mathbf{x}'']$ obtained from (6)–(9) is applied to the neural network. Let the $(i1)_k$th neuron generate the highest output η_{f_k} for Class C_k. We define a membership value for the output of neuron i when calibrated for Class C_k simply as

$$\mu_k(\eta_{ik}) = \frac{\eta_{ik}}{\eta_{f_k}}, \quad \text{for } i = 1, \cdots, N^2, \text{ and } k = 1, \cdots, l \quad (14)$$

such that $0 \leq \mu_k(\eta_{ik}) \leq 1$ and $\mu_k(\eta_{ik}) = 1$ for $i = (i1)_k$.

Each neuron i may be marked by the output Class C_k, among all l classes, that elicits the maximal response η_{ik}. This generates a hard partitioning of the output space. On the other hand, each neuron i has a finite belonging or output membership $\mu_k(\eta_i)$ to Class C_k by (14). We may generate *crisp* boundaries for the fuzzy partitioning of the output space by considering for each of the l classes the α-cut set $\{i \mid \mu_k(\eta_i) > \alpha'\}, 0 < \alpha' \leq 1$, where α' is a suitably chosen value.

Testing Phase: After self-organization, the model encodes all input data information, along with the corresponding contextual class membership values, distributed among its connection weights. During calibration, the neurons are labeled by the pattern classes and the corresponding membership values assigned. This is the desired partially supervised fuzzy classifier. Self-organization and calibration together constitute the training phase of the proposed model. In the final stage, a separate set of test patterns is supplied as input to the neural network model and its performance evaluated.

During this phase the input test vector $\mathbf{x} = [\mathbf{x}', 0]^T$, consisting of only the linguistic information in the $3n$ dimensional space defined by (4), is applied to the network. Let the $p1$th and $p2$th neurons generate the highest and second highest outputs η_{f_p} and η_{s_p}, respectively, (using the largest response criterion), for test pattern $\mathbf{p}$. Then $\mu_{k_1}(\eta_{f_{p,m}})$ and $\mu_{k_2}(\eta_{s_{p,m}})$ are defined as the highest and second highest output membership values generated by pattern $\mathbf{p}$, during testing, with respect to Classes C_{k_1} and C_{k_2}, respectively. These expressions are discussed in detail in the following section with reference to (28)–(30).

III. Inferencing in the Fuzzy Neural Network

Here we concentrate on the inferencing and rule generating capabilities of the fuzzy Kohonen's net-based model on a set of unknown test data. It is to be noted that now the input vector consists of the feature information part only (with no associated class information). The network is expected to be able to infer the correct classification for the test data. Handling of imprecise inputs is possible and natural decision may be obtained associated with a certainty measure denoting the confidence in the decision. The model is capable of:

1) inferencing based on complete and/or partial information;
2) querying the user for unknown input variables that are key to reaching a decision; and
3) producing justification for inferences in the form of *If-Then* rules.

A. Input Feature Representation

The input feature value F_j (for pattern $\mathbf{p}$) can be in quantitative, linguistic or set forms or a combination of these. It is represented as a combination of memberships to the three primary linguistic properties *low*, *medium*, and *high* as in (4), modeled as π-functions.

1) Quantitative Form: When the information is in exact numerical form like F_j *is* r_1, say, we determine the membership values for the different linguistic feature properties *low*, *medium*, and *high* in the corresponding 3-D space of (4) by the π-function using (3).

2) Linguistic Form: When the input is given as F_j *is prop* (say), where *prop* stands for any of the primary linguistic properties *low*, *medium*, or *high*, the membership values in the 3-D space of (4) are assigned using the π-sets of (5).

The proposed model can also handle the linguistic hedges [22] *very*, *more or less* and *not*. Consider a fuzzy set A with the *Concentration (Con)* and *Dilation (Dil)* operators [6] defined as

$$\begin{aligned} \mu_{Con(A)}(z) &= (\mu_A(z))^2 \\ \mu_{Dil(A)}(z) &= (\mu_A(z))^{\frac{1}{2}}. \end{aligned} \quad (15)$$

Using (5) and (15), we define the hedges *very* and *more or less (Mol)* as

$$\begin{aligned} very\ low &\equiv \{Con(L), Con(M), Con(H)\} \\ very\ medium &\equiv \{Con(L), Dil(M), Con(H)\} \\ very\ high &\equiv \{Con(L), Con(M), Con(H)\} \end{aligned} \quad (16)$$

and

$$\begin{aligned} \textit{more or less low} &\equiv \{Con(L), Dil(M), Dil(H)\} \\ \textit{more or less medium} &\equiv \{Dil(L), Con(M), Dil(H)\} \\ \textit{more or less high} &\equiv \{Dil(L), Dil(M), Con(H)\}. \end{aligned} \tag{17}$$

Note that the use of the π-functions for modeling the linguistic functions causes the membership value for the property *low (high)* to decrease in the case of *very low (very high)*. This accounts for the use of the functions $Con(L)$ and $Con(H)$ in these cases [22]. On the other hand, *very medium* causes the membership value for *medium* to increase, thereby justifying the use of $Dil(M)$ in this case.

The hedge *not* is defined as

$$\mu_{NOT(A)}(z) = 1 - \mu_A(z). \tag{18}$$

3) Set Form: Here the input is a mixture of linguistic hedges and quantitative terms. Since the linguistic term increases the impreciseness in the information, the membership value of a quantitative term should be lower when modified by a hedge [22]. The modifiers used are *about, less than, greater than*, and *between*.

For an input F_j *is about* r_1, we use

$$\mu(\textit{about } r_1) = \{\mu(r_1)\}^{1.25} \tag{19}$$

where $F_j = r_1$ is the quantitative input. Note that $\mu(r_1)$ is computed in the corresponding 3-D space of (4) by using (3).

When the input under consideration is F_j *is less than* r_1, the expression becomes

$$\mu(\textit{less than } r_1) = \begin{cases} \{\mu(r_1)\}^{\frac{1}{2}}, & \text{if } r_1 \geq c_{prop} \\ \{\mu(r_1)\}^2, & \text{otherwise} \end{cases} \tag{20}$$

where c_{prop} denotes c_{low}, c_{medium}, and c_{high}, respectively, for each of the corresponding three overlapping partitions as in (4), and $\mu(r_1)$ is computed as explained above.

Similarly, for an input F_j *is greater than* r_1, the expression becomes

$$\mu(\textit{greater than } r_1) = \begin{cases} \{\mu(r_1)\}^{\frac{1}{2}}, & \text{if } r_1 \leq c_{prop} \\ \{\mu(r_1)\}^2, & \text{otherwise} \end{cases}. \tag{21}$$

This also holds for the modifier *more than*.

Input information of the form F_j *is between* r_1 *and* r_2 or F_j *is less than* r_2 *and/but greater than* r_1 may be modeled as the geometric mean of the two component membership values as

$$\begin{aligned} &\mu(\textit{between } r_1 \textit{ and } r_2) \\ &\quad = \{\mu(\textit{less than } r_2) * \mu(\textit{greater than } r_1)\}^{\frac{1}{2}}. \end{aligned} \tag{22}$$

If any input feature F_j is *not available* or *missing*, we clamp the three corresponding input vector components (incident on the neurons) $x_k = x_{k+1} = x_{k+2} = 0.5$, such that $k = (j-1)*3+1$. Here $1 \leq k \leq 3n+l$ and $1 \leq j \leq n$, where n is the dimension of the input pattern vector. Note that here we are dealing with the $3n$-dimensional $\mathbf{x}'$ constituent (feature information) of the input vector $\mathbf{x}$ of (6). We define

$$\textit{no information} \equiv \left\{\frac{0.5}{L}, \frac{0.5}{M}, \frac{0.5}{H}\right\} \tag{23}$$

as 0.5 represents the *most ambiguous* value in the fuzzy membership concept. We also tag these vector components with values $\textit{ininf}_k = \textit{ininf}_{k+1} = \textit{ininf}_{k+2} = 1$. This is a tag used for determining whether the corresponding neuron is *known* or *unknown* by (24) and (25). It is to be mentioned that for the remaining $\{3(n-|j|)+l\}$ input vector components of $\mathbf{x}$ of (6), the corresponding variables $\textit{ininf}_k$ are tagged with 0 (where $|j|$ denotes the number of *missing* input features). This indicates absence of ambiguity in the corresponding input information components.

The appropriate input feature membership values obtained by (3)–(5) and (15)–(23) are clamped at the input neurons.

B. Forward Pass

Initially the system prompts the user for the input feature information that may be provided in any of the forms listed in Section III-A. The components of the l-D contextual class information part $\mathbf{x}''$ of the input vector $\mathbf{x}$ (of (6)) along with the corresponding $\textit{ininf}_k$'s are kept clamped at 0. The N^2 neuron outputs η_i are computed using (1). Associated with each neuron i are

1) its confidence estimation factor $\textit{conf}_i$;
2) a variable $\textit{unknown}_i$ providing the sum of the weighted information from the input components x_k having $\textit{ininf}_k = 1$;
3) a variable $\textit{known}_i$ giving the sum of the weighted information from the (remaining) *nonambiguous* input constituents with $\textit{ininf}_k = 0$.

Note that when there are no input components x_k tagged with $\textit{ininf}_k = 1$, we have $\textit{unknown}_i = 0$ for all the N^2 neurons. The contextual class information part $\mathbf{x}''$ of $\mathbf{x}$ is kept clamped at zero and therefore produces no contribution in this stage. For neuron i we define

$$\begin{aligned} \textit{unknown}_i &= \sum_k m_{ik} x_k \\ \textit{unden}_i &= \sum_k |m_{ik}| \end{aligned} \tag{24}$$

for all k having $\textit{ininf}_k = 1$ and

$$\textit{known}_i = \sum_k m_{ik} x_k \tag{25}$$

for all k with $\textit{ininf}_k = 0$. For the N^2 neurons, we have

$$\textit{noinf}_i = \begin{cases} 1, & \text{if } |\textit{known}_i| \leq |\textit{unknown}_i| \\ 0, & \text{otherwise} \end{cases} \tag{26}$$

where $\textit{noinf}_i$ for the ith neuron is a tag analogous to $\textit{ininf}_k$ for the kth input component. It may be mentioned that for the kth input vector component x_k, we assign $\textit{ininf}_k$ as explained inSection III-A. Neuron i with $\textit{noinf}_i = 1$ signifies absence of meaningful information and is an indicator to the transmission of a larger proportion of weighted *ambiguous information* as compared to *more certain information* from the input vector. On the other hand, an input vector component x_k with $\textit{ininf}_k = 1$ implies missing constituent input information.

Using (1), (24), and (26), we define

$$conf_i = \begin{cases} \left|\frac{\eta_i}{unden_i}\right|, & \text{if } noinf_i = 1 \\ \eta_i, & \text{otherwise} \end{cases} \tag{27}$$

Note that $conf_i$ is comparable either among the set of neurons having $noinf_i = 1$, or among those with $noinf_i = 0$, but not between neurons belonging to these two different sets. For neurons with $noinf_i = 0$, the value of $conf_i$ is higher for neurons having larger η_i and is a measure of confidence in the corresponding decision. But when $noinf_i = 1, conf_i$ gives a measure of confidence of the *ambiguous* neuron output. This is because as $unden_i$ by (24) (absolute sum of connection weights from *ambiguous* inputs) increases, the confidence $conf_i$ decreases and vice versa.

Let neurons $p1$ and $p2$ generate the highest and second highest output responses η_{f_p} and η_{s_p}, respectively, for pattern **p** with input vector **x** (as explained in Section II-B). If neither $noinf_{p1} = 1$ nor $noinf_{p2} = 1$, then the system finalises the decision inferred irrespective of whether the input information is complete or partial. In case of partial inputs, this implies presence of all the *necessary* features required for taking the decision. It may be mentioned that the weights (learned during training) play an important part in determining whether a missing input feature information is *essential* for the final inferred decision or not. This is because these weights are used in computing the $noinf_i$'s for the neurons by (24)–(26) and they in turn determine whether the inferred decision may be taken.

We decide that η_{f_p} and η_{s_p} are in favor of Classes C_{k1} and C_{k2}, respectively, when

$$\begin{aligned} m_{(p1)(3n+k1)} &= \max_k [m_{(p1)(3n+k)}] \\ m_{(p2)(3n+k2)} &= \max_k [m_{(p2)(3n+k)}] \end{aligned} \tag{28}$$

where $k = 1, \ldots, l$. Note that the $(3n + l)$-dimensional connection weights $\mathbf{m}_i$'s are *learned* during self-organization and constitute the *knowledge base* for the problem after appropriate labeling during calibration.

The inferred highest and second highest output memberships $\mu_{k_1}(\eta)$ and $\mu_{k_2}(\eta)$ to Classes C_{k_1} and C_{k_2}, respectively, are given as

$$\begin{aligned} \mu_{k_1}(\eta) &= \frac{m_{(p1)(3n+k1)}}{s} \\ \mu_{k_2}(\eta) &= \frac{m_{(p2)(3n+k2)}}{s} * \frac{\eta_{s_p}}{\eta_{f_p}} \end{aligned} \tag{29}$$

with $p_1 = p1, p_2 = p2, k_1 = k1$, and $k_2 = k2$, if $m_{(p1)(3n+k1)} \geq m_{(p2)(3n+k2)} * \frac{\eta_{s_p}}{\eta_{f_p}}$. Otherwise

$$\begin{aligned} \mu_{k_1}(\eta) &= \frac{m_{(p2)(3n+k2)}}{s} * \frac{\eta_{s_p}}{\eta_{f_p}} \\ \mu_{k_2}(\eta) &= \frac{m_{(p1)(3n+k1)}}{s} \end{aligned} \tag{30}$$

where $p_1 = p2, p_2 = p1, k_1 = k2$ and $k_2 = k1$. Here p_1 and p_2 refer to the neurons inferred to be generating the highest and second highest membership values to Classes C_{k_1} and C_{k_2}, respectively, and s is the scaling factor from (8).

Note that the difficulty in arriving at a particular decision in favor of Class C_{k_1} is dependent not only on the highest membership value $\mu_{k_1}(\eta)$ but also on its differences with other class membership values $\mu_k(\eta)$ where $k \neq k_1$. To take this factor into account, a certainty measure for the neuron p_1 is defined as

$$bel_{p_1}^{k_1} = \frac{1}{s} * \left[m_{(p_1)(3n+k_1)} - \sum_{k=1}^{l} m_{(p_1)(3n+k)} \right] \tag{31}$$

where $k \neq k_1$ and $bel_{p_1}^{k_1} \leq 1$. Here k_1 and p_1 are obtained from (29) and (30). The higher the value of $bel_{p_1}^{k_1} (>0)$, the lower is the difficulty in deciding an output Class C_{k_1} and hence the greater is the degree of certainty of the output decision.

Depending on the value of $bel_{p_1}^{k_1}$, the final inferred output may be given in natural form irrespective of whether the input is fuzzy/deterministic and complete/partial. This is explained in detail in Section III-D.

C. Querying

If the system has not yet reached conclusions to complete the session, it must find an input feature with *unknown* activation and ask the user for its value. If either $noinf_{p_1} = 1$ or $noinf_{p_2} = 1$ by (26), where p_1 and p_2 are obtained from (29) and (30), we begin the querying phase. We select the *unknown* output neuron i_1 from among the p_1th and/or p_2th neuron(s) with $noinf_{p_1} = 1$ and/or $noinf_{p_2} = 1$ such that $conf_{i_1}$ by (27) (among them) is maximum.

We pursue the path from neuron i_1 to find the *ambiguous* input feature vector component x_{k_1} with the greatest absolute influence on neuron i_1. For this, we select $k = k_1$ such that

$$|m_{i_1 k_1} * x_{k_1}| = \max_k |m_{i_1 k} * x_k| \quad \text{where } ininf_k = 1. \tag{32}$$

Here $ininf_k$ is obtained as explained in Section III-A and $1 \leq k_1 \leq 3n + l$. The model queries the user for the value of the corresponding input feature j_1 such that

$$j_1 = (k_1 - 1) \bmod 3 + 1 \tag{33}$$

where $1 \leq j_1 \leq n$ and n is the dimension of the input pattern vector. Note that here the $3n$-dimensional input feature information vector $\mathbf{x}'$ of (6) is under consideration.

When the user is asked for the value of a *missing* variable, she can respond in any of the forms given in Section III-A. Note that if a *missing* input variable by (23) is found to be missing once again, we now tag it as *unobtainable*. This implies that the value of this variable will not be available for the remainder of this session. The inferencing mechanism treats such variables as *known* with values $x_{k_1} = x_{k_1+1} = x_{k_1+2} = 0.5$ but with $ininf_{k_1} = ininf_{k_1+1} = ininf_{k_1+2} = 0$ such that $k_1 = (j_1 - 1) * 3 + 1$. We now use

$$information \equiv \left\{ \frac{0.5}{L}, \frac{0.5}{M}, \frac{0.5}{H} \right\}. \tag{34}$$

Note the difference from (23) in the value of $ininf_k$ and its effect in the confidence estimation by (24)–(27). The response from an *unobtainable* input variable might allow the neuron activations to be inferred, unlike that of a *missing* variable.

Besides, a *missing* variable has a temporary value of 0.5 that may be changed later in the session, whereas an *unobtainable* variable has a *known final* value of 0.5. Here again, the use of the more conventional values of $x_{k_1} = x_{k_1+1} = x_{k_1+2} = 0.2$ in (34) yields no appreciable differences in the results.

Once the requested input variable is supplied by the user, the procedure in Section III-B is followed to determine whether to infer a decision or again continue with further querying. On completion of this phase we have $noinf_{p_1} = noinf_{p_2} = 0$ by (26).

D. Justification

The user can ask the system why it inferred a particular conclusion, say, at neuron p_1 or p_2. The system answers with an *If-Then* rule applicable to the case at hand. It is to be noted that these *If-Then* rules are not represented explicitly in the knowledge base; they are generated by the *inferencing system* from the connection weights when needed for explanations. As the model has already inferred a conclusion (in this stage), we take a subset of the currently known information to justify this decision.

1) Path Selection: Let the user ask justification for a conclusion regarding Class C_{k_1} at neuron p_1. Starting from neuron p_1, the process reasons *backward* to the input vector along the *maximum weighted* paths. We select those input feature vector components x_k that have a significant positive impact on the conclusion reached at neuron p_1.

We choose input feature vector component x_k if

$$x_k > 0.5 \tag{35}$$

where $0 \le k \le 3n$. Let the set of h components so chosen be $\{x_{k_1}, x_{k_2}, \ldots, x_{k_h}\}$ and their corresponding link weights to neuron p_1 be $\{m_{p_1 k_1}, m_{p_1 k_2}, \ldots, m_{p_1 k_h}\}$, respectively. This implies choosing a path that is currently active for deciding the conclusion that is being justified. In other words, this helps selecting paths along which the corresponding input vector component has a significant positive correlation(influence) with(on) the neuron under consideration.

We arrange the chosen input components in the decreasing order of their *net impacts*, where we define the net impact for x_k, as

$$net\ impact_k = x_k * m_{p_1 k}.$$

Then we generate clauses for an *If-Then* rule from this ordered list until

$$\sum_{k_s} m_{p_1 k_s} > 2 \sum_{k_n} m_{p_1 k_n} \tag{36}$$

where k_s indicates the input components selected for the clauses and k_n denotes the input components remaining from the set $\{x_{k_1}, x_{k_2}, \ldots, x_{k_h}\}$ such that

$$|k_s| + |k_n| = h$$

and $|k_s|, |k_n|$ refer, respectively, to the number of components selected and remaining from the said set. This heuristic allows selection of those *currently active* $|k_s|$ input vector constituents contributing *the most* to the final conclusion (among those lying along the maximum weighted paths to the output node p_1) as the clauses of the antecedent part of a rule. It also enables the *currently active* test pattern inputs (current evidence) to influence the generated *knowledge base* (connection weights learned during training) in producing a rule to justify the *current* inference.

2) Clause Generation: For an input component $x_{k_{s_1}}$, selected for clause generation, the corresponding input feature j_{s_1} is obtained as in (33) as

$$j_{s_1} = (k_{s_1} - 1) \bmod 3 + 1$$

where $1 \le j_{s_1} \le n$ and $1 \le k_{s_1} \le 3n$. The antecedent of the rule is given in linguistic form with the linguistic property being determined as

$$prop = \begin{cases} low, & \text{if } k_{s_1} - 3(j_{s_1} - 1) = 1 \\ medium, & \text{if } k_{s_1} - 3(j_{s_1} - 1) = 2 \\ high, & \text{otherwise.} \end{cases} \tag{37}$$

A linguistic hedge *very*, *more or less* or *not* may be attached to the linguistic property in the antecedent part, if necessary. We use the mean square distance $d(j_{s_1}, pr_m)$ between the 3-D input feature values corresponding to feature j_{s_1} and the linguistic property *prop* by (37), with or without modifiers, represented as pr_m. The corresponding 3-D values for pr_m corresponding to *prop* are given by (5) (with no modifiers) and by (16)–(18) with the modifiers *very*, *more or less* and *not*, respectively. The pr_m for which $d(j_{s_1}, pr_m)$ is the *minimum* is selected as the antecedent clause corresponding to feature j_{s_1} (or input component $x_{k_{s_1}}$) for the rule justifying the conclusion at output neuron p_1.

This procedure is repeated for all the $|k_s|$ neurons selected by (36) to generate a set of conjunctive antecedent clauses for the rule regarding inference at node p_1.

3) Consequent Deduction: The consequent part of the rule can be stated in quantitative form as membership value $\mu_{k_1}(y)$ to Class C_{k_1} by (29)–(30). However a more *natural* form of decision can also be provided considering the value of $bel_{p_1}^{k_1}$ of (31). For the linguistic output form we use

1) *very likely* for $0.8 \le bel_{p_1}^{k_1} \le 1$
2) *likely* for $0.6 \le bel_{p_1}^{k_1} < 0.8$
3) *more or less likely* for $0.4 \le bel_{p_1}^{k_1} < 0.6$
4) *not unlikely* for $0.1 \le bel_{p_1}^{k_1} < 0.4$
5) *unable to recognize* for $bel_{p_1}^{k_1} < 0.1$

Note that here the belief is converted to linguistic classes to enable the expression of the consequent part of the rule in a *normal* form. However, one may keep or use the actual belief values also.

In principle it should be possible to examine a connectionist network and produce every such *If-Then* rule, using the inferred class memberships for the various test patterns in conjunction with the *learned* connection weights.

IV. Implementation and Results

The above-mentioned algorithm was first tested on a set of 871 Indian Telugu vowel sounds collected by trained personnel [23]. These were uttered in a Consonant-Vowel-Consonant context by three male speakers in the age group

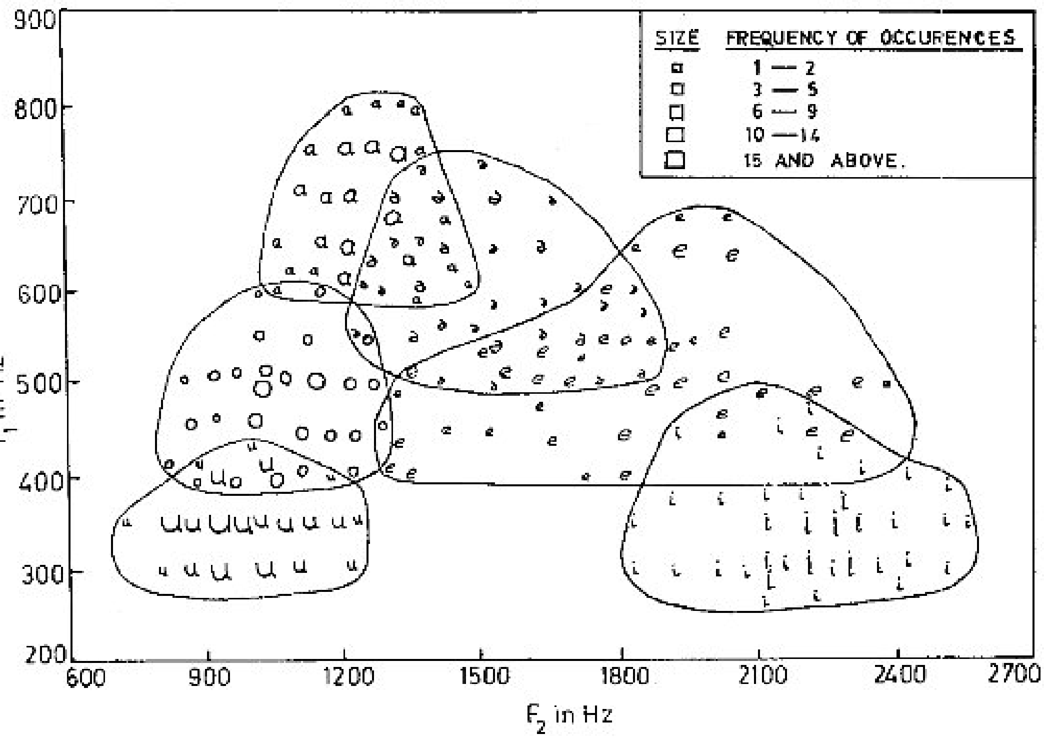

Fig. 2. Vowel diagram in the $F_1 - F_2$ plane.

of 30 to 35 years old. The data set has three features; F_1, F_2, and F_3 corresponding to the first, second and third vowel formant frequencies obtained through spectrum analysis of speech data. Fig. 2 shows a 2-D projection of the 3-D feature space of the six vowel classes $(\partial, a, i, u, e, o)$ in the $F_1 - F_2$ plane (for ease of depiction). The dimension of the input vector in (6) for the proposed model is 15. Note that the boundaries of the classes in the given data set are seen to be ill-defined (fuzzy). The training data has the complete set of input feature information along with the contextual class membership components. During self-organization, *perc*% samples were randomly chosen from each representative pattern class. The remaining (100 – *perc*)% samples from the original data set were used as the test set. We selected $F_d = 5$ and $F_e = 1$ in (7), $s = 0.2$ in (8), $kn = 5$ in (12) and $\delta = 0.0001$ in (13) after several experiments. The test set uses complete/partial sets of inputs and the appropriate classification is inferred by the trained neural model along with a measure of certainty. Querying regarding unknown input feature values is resorted to in case of some partial input sets. Justification in *If-Then* rule form, regarding a condition, is also obtained when desired.

The model has also been used on two sets (A, B) of artificially generated linearly nonseparable, nonconvex pattern classes represented in the 2-D feature space F_1-F_2, each set consisting of 880 pattern points. These are given in Figs. 3 and 4. The training set consists of the complete pattern vectors in the 9-dimensional (9-D) form of (6) with the appropriate contextual class information. Note that *missing* and *unobtainable* refer to the conditions given in (23) and (34), respectively, $bel_{p_l}^{k_1}$ is obtained from (31) and $\mu_{k_1}(\eta)$; $\mu_{k_2}(\eta)$ are computed from (29) and (30).

TABLE I

COMPARISON OF RECOGNITION SCORE (%) BETWEEN BAYES' CLASSIFIER, STANDARD SUPERVISED FUZZY CLASSIFIER AND THE FUZZY NEURAL NET MODEL ON THE VOWEL DATA. NEURAL NETWORK IS OF SIZE 10×10 WITH $epoch = 20$

Class	Bayes' classifier	Standard fuzzy classifier	Fuzzy neural model
∂	44.6	51.4	23.0
a	83.9	81.7	97.5
i	81.9	73.0	74.8
u	88.9	67.6	73.5
e	82.3	77.7	85.7
o	77.7	78.8	92.6
Overall	79.6	73.4	79.6

A. Vowel Data

The details regarding the classification performance on various training and test sets as well as the choice of parameters for the said model (on the vowel data) have already been reported in [8]. Table I compares the recognition score (on test set) of the fuzzy neural net model (trained using 10% samples from each representative vowel class) to that of the Bayes' classifier [24], [25], and the standard fully supervised fuzzy approach [23]. We have used the Bayes' classifier for multivariate normal patterns with the *a priori* probabilities $p_i = \frac{|C_i|}{N}$, where $|C_i|$ indicates the number of patterns in class C_i and N is the total number of pattern points. The dispersion matrices are different for each pattern class. The overall performance of the model is found to be quite satisfactory. Next we

TABLE II
INFERRED OUTPUT RESPONSES AND CERTAINTY MEASURES FOR A SET OF VOWEL DATA PRESENTED TO THE *trained* NEURAL NETWORK MODEL

Serial	Input features			First choice		Second choice		Certainty
No.	F_1	F_2	F_3	C_{k_1}	$\mu_{k_1}(n)$	C_{k_2}	$\mu_{k_2}(n)$	$bel_{P_1}^{k_1}$
1	700	1000	*missing*	a	0.84	a	0.81	0.80
2	700	1000	2600	a	0.83	a	0.67	0.45
3	700	*missing*	*unobtainable*	a	0.66	∂	0.38	-0.55
4	400	*unobtainable*	*missing*	e	0.59	e	0.48	0.30
5	300	900	*missing*	u	0.87	u	0.76	0.65
6	450	2400	*missing*	e	0.92	e	0.91	0.85
7	700	2300	*missing*	ε	0.89	ε	0.75	0.65
8	900	1400	*missing*	a	0.78	a	0.67	0.45
9	*high*	*Mol low*	*missing*	a	0.80	a	0.67	0.45
10	*between* 500 *&* 600	1600	*missing*	∂	0.62	ε	0.48	0.30
11	*greater than* 650	*high*	*missing*	ε	0.75	∂	0.37	0.65

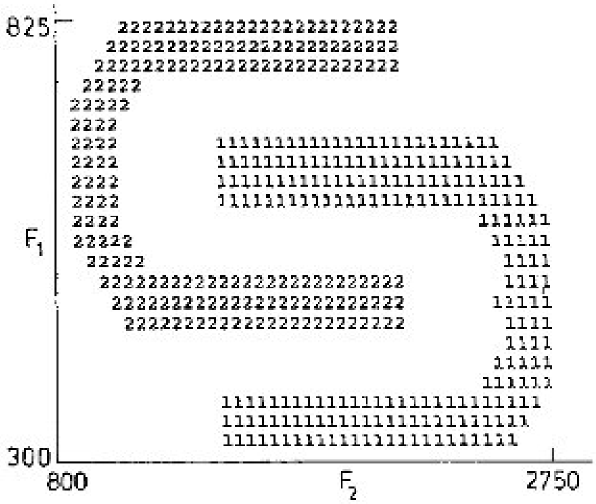

Fig. 3. Pattern set A in the F_1-F_2 plane.

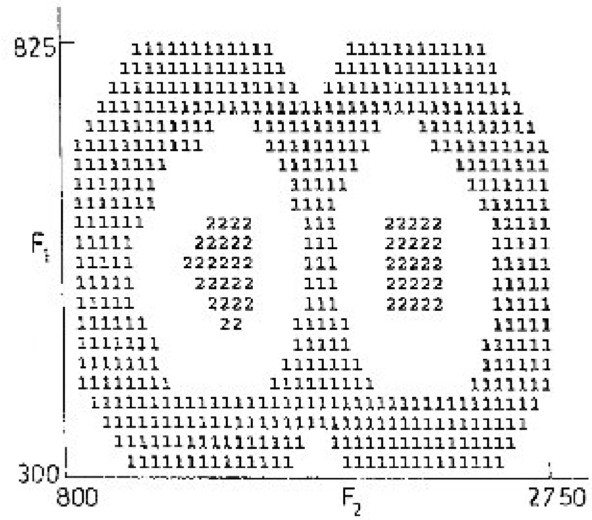

Fig. 4. Pattern set B in the F_1-F_2 plane.

TABLE III
QUERYING MADE BY THE NEURAL NETWORK MODEL WHEN PRESENTED WITH A SET OF *partial* PATTERN VECTORS FOR VOWEL DATA

Serial	Input features			Query
No.	F_1	F_2	F_3	for
1	700	*missing*	*missing*	F_3
2	*about* 350	*missing*	*missing*	F_2
3	400	*missing*	*missing*	F_2
4	*missing*	1000	*missing*	F_1

demonstrate a sample of the inferencing ability of a *trained* 10×10 *neural model* (with $cdenom = 100$) that functions as a *knowledge base* for the vowel recognition problem. The results are demonstrated in Tables II–IV.

Table II illustrates the inferred output response of the proposed model on a set of partial and complete input feature vectors. It is observed that often the two features F_1 and F_2 are sufficient for reaching a conclusion. This may easily be verified from the 2-D representation of the vowel data in Fig. 2. Let us consider the first three entries in the table. When only F_1 is specified (Entry 3), we have a horizontal line across Fig. 2 at this F_1 value. For this ambiguous case both classes a and ∂ register positive belongingness, although class a is the *more likely* winner. Note that F_3 = *unobtainable* is generated by (34). The specification of F_2 and F_3 is seen to further consolidate this decision. The 4th entry corresponds to pattern class e, owing to its highest horizontal coverage along this F_1 value, although the ambiguity in the decision is evident from the value of the certainty measure. The 8th entry is in favor of class a owing to its proximity to this class. The 10th entry corresponds to a strip of area at $F_2 = 1600$ between $F_1 = 500$ and 600 in Fig. 2. This region corresponds to both classes ∂ and ε and the ambiguity of the point is evident in the value of the certainty measure.

In Table III we demonstrate a sample of the partial input feature combinations that are insufficient for inferring any particular decision. The *more essential* of the feature value(s) is queried for by (32), (33). Table IV shows the rules generated from the *knowledge base* by presenting a sample set of test patterns. The antecedent parts are obtained using (35)–(37) while the consequent parts are deduced from the values of

TABLE IV
RULES GENERATED BY THE NEURAL NETWORK MODEL TO JUSTIFY ITS INFERRED DECISIONS FOR A SET OF PATTERN VECTORS FOR VOWEL DATA

Sr. No.	Input features F_1	F_2	F_3	Justification / Rule generation: *If* clause	*Then* conclusion
1	300	900	*missing*	F_2 is *very low*	*likely* class *u*
2	700	1000	2600	F_2 is *very low* and F_1 is *Mol high*	*more or less likely* class *a*
3	700	2300	*missing*	F_2 is *very high* and F_1 is *very medium*	*likely* class *e*
4	450	2400	*missing*	F_2 is *very high* and F_1 is *very medium*	*very likely* class *e*
5	900	1400	*missing*	F_2 is *Mol low* and F_1 is *very high*	*more or less likely* class *a*
6	*high*	*Mol low*	*missing*	F_2 is *Mol low* and F_1 is *high*	*more or less likely* class *a*
7	*between* 500 & 600	1600	*missing*	F_2 is *very medium* and F_1 is *very medium*	*not unlikely* class ∂
8	*greater than* 650	*high*	*missing*	F_1 is *very medium* and F_2 is *high*	*likely* class *e*

TABLE V
COMPARISON BETWEEN RECOGNITION SCORES FOR VARIOUS SIZES OF NEURAL NET ARRAYS WITH *cdenom* = 100, USING DIFFERENT VALUES OF *perc* ON PATTERN SET *A*

Size	14 × 14		16 × 16		18 × 18
perc	50	10	50	10	50
1	63.4	56.7	53.0	64.2	51.7
2	50.5	30.8	55.5	30.2	40.2
none	50.4	78.1	65.6	63.5	56.0
Overall	53.7	62.8	62.6	56.4	51.6

TABLE VI
COMPARISON BETWEEN RECOGNITION SCORES FOR VARIOUS SIZES OF NEURAL NET ARRAYS WITH *cdenom* = 100, USING DIFFERENT VALUES OF *perc* ON PATTERN SET *B*

Size	10 × 10	14 × 14		16 × 16		18 × 18
perc	10	50	10	50	10	50
1	43.3	77.3	50.6	82.7	51.7	72.7
2	39.1	53.8	58.7	19.2	84.7	65.3
none	81.7	44.5	51.6	55.4	35.8	53.5
Overall	56.6	64.4	51.5	69.4	48.0	65.5

the certainty measure $\omega_{p_1}^{k_1}$. The rules obtained may be verified by comparing with Fig. 2. Note that Entries 3 and 4 generate slightly different consequent parts for a rule with the same antecedent clauses. This is because different pattern points are used to obtain the two justifications. Entries 5 and 6 generate the same rules from numeric and linguistic input specifications, respectively.

B. Artificially Generated Data

The model was next trained on the two sets of linearly nonseparable, nonconvex pattern classes in succession, using various sizes of neural network arrays. Two nonseparable pattern classes (1 and 2) were considered in each case. The region of *no pattern points* was modeled as class *none (no class)*. Tables V and VI are used to compare the performance on test set (both classwise and overall) of different sizes of neural net arrays on the two sets of nonseparable patterns *A* and *B*, respectively. Various training set sizes *perc* were chosen from each representative pattern class in both the cases. The patterns were best classified by using neural arrays of size 16 × 16 (with 50% of the data used as training set in each case).

TABLE VII
INFERRED OUTPUT RESPONSES AND CERTAINTY MEASURES FOR A SAMPLE OF PATTERN SET *A* DATA PRESENTED TO THE *trained* NEURAL NETWORK MODEL

Serial No.	Input features F_1	F_2	First choice C_{k_1}	$\nu_{k_1}(\eta)$	Second choice C_{k_2}	$\nu_{k_2}(\eta)$	Certainty $bel_{y_j}^{k_1}$ or $bel_{y_j}^{k_2}$
1	*Mol low*	*missing*	*none*	0.81	2	0.65	(0.30)
2	*medium*	*missing*	*none*	0.78	*none*	0.67	0.56
3	*Mol high*	*missing*	1	0.99	1	0.99	0.99
4	*low*	*medium*	2	0.86	2	0.82	0.73
5	*medium*	*medium*	*none*	0.78	*none*	0.67	0.34
6	*high*	*high*	1	0.89	*none*	0.77	0.54

In Tables VII and X we demonstrate the inferred output responses of the neural net model on some partial and complete input feature vectors for the two pattern sets. Tables VIII and XI show the querying phase where in some cases the missing feature information is *necessary* for inferring a decision and hence queried for. Tables IX and XII illustrate the generation of a few rules from the two *knowledge bases*. Verification regarding these tables may be made by examining the original patterns given in Figs. 3 and 4.

Entries 1, 2, and 3 in Table VII correspond to horizontal bands across Fig. 3, illustrating Pattern Set *B*, around the given F_1 values. The certainty value in brackets (case 1) indicates belief more in favor of the decision of second choice for Class C_{k_2} as compared to Class C_{k_1}. This is obtained from the connection weight values as given in (31). In the 1st entry the decision is *ambiguous* and in favor of Class 2, as observed from the certainty measure (although Class *none* generates the highest response followed by Class 2). As F_1 changes to *medium* (in Entry 2), the decision becomes *more certain* in favor of Class *none* while for F_1 = *Mol high* (case

TABLE VIII
QUERYING PHASE IN THE NEURAL NETWORK MODEL WHEN PRESENTED WITH A SAMPLE OF *partial* INPUT VECTORS FOR PATTERN SET A DATA

Serial	Input features		Query
No.	F_1	F_2	for
1	*medium*	*missing*	-
2	*not medium*	*missing*	F_2
3	*Mol high*	*missing*	-
4	*not high*	*missing*	-

TABLE IX
RULES GENERATED BY THE NEURAL NETWORK MODEL TO JUSTIFY ITS INFERRED DECISIONS FOR A SAMPLE OF INPUT VECTORS FOR PATTERN SET A DATA

Serial	Input features		Justification / Rule generation	
No.	F_1	F_2	*If clause*	*Then conclusion*
1	*Mol low*	*missing*	F_1 is *very medium*	*not unlikely* class 2
2	*medium*	*missing*	F_1 is *medium*	*not unlikely no class*
3	*not medium*	*low*	F_2 is *low*	*not unlikely no class*
4	*Mol high*	*missing*	F_1 is *Mol high*	*very likely* class 1
5	*low*	*medium*	F_2 is *medium* and F_1 is *low*	*likely* class 2
6	*medium*	*low*	F_1 is *medium* and F_2 is *low*	*very likely no class*
7	*medium*	*medium*	F_1 is *medium* and F_2 is *medium*	*not unlikely no class*
8	*high*	*high*	F_2 is *high* and F_1 is *high*	*Mol likely* class 1

3), the decision is *certainly* in favor of Class 1. Entry 5, with F_1 = *medium*, is *less certain* in inferring Class *none* as compared to Entry 4, with F_1 = *low* (both with F_2 = *medium*). The latter yields a *less ambiguous* decision (as observed from the certainty measure) in favor of Class 2. Note that the value of the certainty measures, and not the output membership values, determine the *ambiguity* in a decision. Entry 6 gives a *more or less ambiguous* decision in favor of Class 1 while also producing a significant response for Class *none*. All results of Tables VII and IX may be verified from Fig. 3.

Entries 1, 2, 3, 4, 6, and 7 in Table X correspond to horizontal bands across Fig. 4, illustrating Pattern Set B, around the given F_1 values. Among these, Entries 1, 3, 6, and 7, (with F_1 = *low, not low, high,* and *not high*, respectively,) generate *certain* decisions in favor of Class 1. However, Entries 4 and 2 (with F_1 = *medium* and *Mol low*, respectively) produce *rather ambiguous* decisions (low values of certainty measure) in favor of Classes 1 and *none*, respectively. Comparing Entries 5, 8, and 11, we find that cases 8 and 11 (with F_1 = *low* and *high*, respectively) generate decisions in favor of Class *none* while case 5 (with F_1 = *not medium*) produces a decision in favor of Class 1. From the 9th, 10th, and 12th entries we observe that Entry 12 (F_1 = *high*) produces the *most certain* decision in favor of Class 1 with Entry 9 (F_1 = *low*) following close behind. On the other hand, Entry 10 (with F_1 = *medium*) produces a *more ambiguous (less certain)* decision for Class 1. All results of Tables X–XII may be verified from Fig. 4. In Table XII, Entry 2 (corresponding to the 2nd entry in Table X) is unable to infer any positive decision (*unable to recognize*) due to the extremely low certainty measure generated in this case.

TABLE X
INFERRED OUTPUT RESPONSES AND CERTAINTY MEASURES FOR A SAMPLE OF PATTERN SET B DATA PRESENTED TO THE *trained* NEURAL NETWORK MODEL

Serial	Input features		First choice		Second choice		Certainty
No.	F_1	F_2	C_{k_1}	$\mu_{k_1}(\vec{\eta})$	C_{k_2}	$\mu_{k_2}(\vec{\eta})$	$bel_{\vec{y}}^{k}$
1	*low*	*missing*	1	0.93	1	0.59	0.87
2	*Mol low*	*missing*	*none*	0.67	*none*	0.54	0.07
3	*not low*	*missing*	1	0.99	1	0.98	0.97
4	*medium*	*missing*	1	0.60	1	0.56	0.11
5	*not medium*	*low*	1	1.0	1	0.99	1.0
6	*high*	*missing*	1	0.99	1	0.98	0.97
7	*not high*	*missing*	1	0.94	1	0.58	0.87
8	*low*	*low*	*none*	0.97	*none*	0.76	0.95
9	*low*	*medium*	1	0.94	*none*	0.53	0.87
10	*medium*	*medium*	1	0.95	1	0.60	0.20
11	*high*	*low*	*none*	0.98	*none*	0.95	0.97
12	*high*	*medium*	1	0.99	1	0.98	0.97

TABLE XI
QUERYING PHASE IN THE NEURAL NETWORK MODEL WHEN PRESENTED WITH A SAMPLE OF *partial* INPUT VECTORS FOR PATTERN SET B DATA

Serial	Input features		Query
No.	F_1	F_2	for
1	*low*	*missing*	-
2	*Mol low*	*missing*	-
3	*medium*	*missing*	-
4	*not medium*	*missing*	F_2
5	*high*	*missing*	-
6	*not high*	*missing*	-

V. CONCLUSION AND DISCUSSION

In this work we considered an application of the fuzzy self organizing neural network model [8], based on Kohonen's net, capable of inferencing and rule generation. The connection weights of the neural net (after self-organization and calibration) constituted the *knowledge base* for the problem under consideration. The network was capable of handling uncertainty and/or impreciseness in the input representation provided in quantitative, linguistic and/or set forms. The output class memberships were inferred for the input patterns. The user could be queried for the *more essential* feature information in case of partial inputs, when so required. Justification for the decision reached was generated in rule form. The antecedent and consequent parts of these rules were provided in linguistic and *natural* terms. The magnitudes of the connection weights of the *trained* neural net were used in every stage of the inferencing procedure. A measure of certainty expressing confidence (belief) in an output decision was defined and also used in generating the consequent part of the corresponding justificatory rule. The effectiveness of the model was demonstrated on the vowel recognition problem and on two sets of artificially generated linearly nonseparable, nonconvex pattern classes.

It should be noted that the two given artificially generated data sets (A, B) consist of nonseparable patterns. Hence any evaluation of the performance of the proposed model on these data sets should be made in this context. This accounts for the relatively better inferencing as well as rule generating

TABLE XII
RULES GENERATED BY THE NEURAL NETWORK MODEL TO JUSTIFY ITS INFERRED DECISIONS FOR A SAMPLE OF INPUT VECTORS FOR PATTERN SET B DATA

Serial	Input features		Justification / Rule generation	
No.	F_1	F_2	*If* clause	*Then* conclusion
1	*low*	*missing*	F_1 is *low*	*very likely* class 1
2	*Mol low*	*missing*	F_1 is *Mol low*	*unable to recognize*
3	*not low*	*missing*	F_1 is *very high*	*very likely* class 1
4	*medium*	*missing*	F_1 is *medium*	*Mol likely* class 1
5	*not medium*	*low*	F_2 is *low*	*very likely* class 1
6	*high*	*missing*	F_1 is *high*	*very likely* class 1
7	*not high*	*missing*	F_1 is *very low*	*very likely* class 1
8	*low*	*low*	F_1 is *low* and F_2 is *low*	*very likely no class*
9	*low*	*medium*	F_2 is *medium* and F_1 is *low*	*very likely* class 1
10	*low*	*high*	F_1 is *low* and F_2 is *high*	*very likely no class*
11	*medium*	*low*	F_1 is *medium* and F_2 is *low*	*likely no class*
12	*medium*	*medium*	F_2 is *medium* and F_1 is *medium*	*not unlikely* class 1
13	*high*	*low*	F_1 is *high* and F_2 is *low*	*very likely no class*
14	*high*	*medium*	F_2 is *medium* and F_1 is *high*	*very likely* class 1

capability of this model on the vowel data as compared to that on the two given pattern sets.

It is worth mentioning that a genetic algorithm based approach has been investigated for tuning the output class-membership values of (7). The F_d-F_e pair values thus obtained correspond to the best values that we obtained experimentally in this study. Some other work on the tuning of the linguistic functions at the input have been reported [26], based on automatically determining the centres and radii of the π-functions from the training patterns.

The model described here has been initially trained as a partially supervised fuzzy classifier, and then used for inferencing, querying and rule generation for unknown test patterns. Incorporation of clustering at the input level, for generating initial seeds, could be an interesting area for future investigation.

ACKNOWLEDGMENT

The authors gratefully acknowledge S. Chakraborty for drawing some of the diagrams. This work was done when Sankar K. Pal held the Jawaharlal Nehru Fellowship.

REFERENCES

[1] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-4, pp. 4–22, 1987.
[2] D. E. Rumelhart and J. L. McClelland, Eds.,*Parallel Distributed Processing: Explorations in the Microstructures of Cognition*, vol. 1, Cambridge, MA: MIT Press, 1986.
[3] T. Kohonen, "An introduction to neural computing," *Neural Networks*, vol. 1, pp. 3–16, 1988.
[4] ——, *Self Organization and Associative Memory*. Berlin: Springer-Verlag, 1989.
[5] L. A. Zadeh, "Making computers think like people," *IEEE Spectrum*, pp. 26–32, Aug. 1984.
[6] S. K. Pal and D. Dutta Majumder, *Fuzzy Mathematical Approach to Pattern Recognition*. New York: Wiley, 1986.
[7] G. J. Klir and T. Folger, *Fuzzy Sets, Uncertainty and Information*. Reading, MA: Addison-Wesley, 1989.
[8] S. Mitra and S. K. Pal, "Self-organizing neural network as a fuzzy classifier," *IEEE Trans. Syst., Man, Cybern.*, vol. 24, no. 3, pp. 385–399, 1994.
[9] H. Takagi and I. Hayashi, "Artificial neural network driven fuzzy reasoning," *Int. J. Approx. Reas.*, vol. 5, pp. 191–212, 1991.
[10] J. M. Keller and H. Tahani, "Backpropagation neural networks for fuzzy logic," *Inform. Sci.*, vol. 62, pp. 205–221, 1992.
[11] S. I. Gallant, "Connectionist expert systems," *Commun. Assoc. Comput. Mach.*, vol. 31, pp. 152–169, 1988.
[12] W. Pedrycz, "Neurocomputations in relational systems," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 289–297, 1991.
[13] ——, "Fuzzy neural networks with reference neurons as pattern classifiers," *IEEE Trans. Neural Networks*, vol. 3, pp. 770–775, 1992.
[14] R. Krishnapuram and J. Lee, "Fuzzy-set-based hierarchical networks for information fusion in computer vision," *Neural Networks*, vol. 5, pp. 335–350, 1992.
[15] H. Ishibuchi, R. Fujioka, and H. Tanaka, "Neural networks that learn from fuzzy If-Then rules," *IEEE Trans. Fuzzy Syst.*, vol. 1, pp. 85–97, 1993.
[16] S. K. Pal and S. Mitra, "Multi-layer perceptron, fuzzy sets and classification," *IEEE Trans. Neural Networks*, vol. 3, pp. 683–697, 1992.
[17] S. Mitra and S. K. Pal, "Fuzzy multi-layer perceptron, inferencing and rule generation," *IEEE Trans. Neural Networks*, vol. 6, no. 1, pp. 51–63, 1994.
[18] ——, "Logical operation based fuzzy MLP for classification and rule generation," *Neural Networks*, vol. 7, pp. 353–373, 1994.
[19] T. L. Huntsberger and P. Ajjimarangsee, "Parallel self-organizing feature maps for unsupervised pattern recognition," *Int. J. Gen. Syst.*, vol. 16, pp. 357–372, 1990.
[20] J. C. Bezdek, E. C. Tsao, and N. R. Pal, "Fuzzy Kohonen clustering networks," in *Proc. 1st IEEE Int. Conf. Fuzzy Systems*, San Diego, CA, 1992, pp. 1035–1043.
[21] S. K. Pal and P. K. Pramanik, "Fuzzy measures in determining seed points in clustering," *Pattern Recognit. Lett.*, vol. 4, pp. 159–164, 1986.
[22] S. K. Pal and D. P. Mandal, "Linguistic recognition system based on approximate reasoning," *Inform. Sci.*, vol. 61, pp. 135–161, 1992.
[23] S. K. Pal and D. D. Majumder, "Fuzzy sets and decision making approaches in vowel and speaker recognition," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-7, pp. 625–629, 1977.
[24] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.
[25] J. T. Tou and R. C. Gonzalez, *Pattern Recognition Principles*. London: Addison-Wesley, 1974.
[26] S. Mitra, "Fuzzy MLP based expert system for medical diagnosis," *Fuzzy Sets Syst.*, vol. 65, pp. 285–296, 1994.

Sushmita Mitra received the B.Sc. (Hons.) degree in physics and the B.Tech., and M.Tech. degrees in computer science from the University of Calcutta, India, in 1984, 1987, and 1989, respectively.

From 1989 to 1991, she was a Senior Research Fellow of the Council for Scientific and Industrial Research. Since 1991, she has been with the Indian Statistical Institute, Calcutta, as a Programmer. From 1992 to 1994, she was in the European Laboratory for Intelligent Techniques Engineering, Aachen, as a German Academic Exchange Service Fellowship holder. Her research interests include pattern recognition, fuzzy sets, artificial intelligence, and neural networks.

From 1978 to 1983, Ms. Mitra was a recipient of the National Talent Search Scholarship from the National Council for Educational Research and Training, India. She received the 1994 IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award.

Sankar K. Pal (M'81–SM'84–F'92) received the M.Tech. and Ph.D. degrees in radiophysics and electronics from the University of Calcutta, India, in 1974 and 1979, respectively. In 1982, he received another Ph.D. degree in electrical engineering, along with DIC, from Imperial College, University of London.

In 1986, he was awarded the Fulbright Post-doctoral Visiting Fellowship to work at the University of California, Berkeley, and the University of Maryland, College Park. Currently, he is a Professor and Founding Head of the Machine Intelligence Unit at the Indian Statistical Institute, Calcutta. His research interests mainly include pattern recognition, image processing, neural nets, genetic algorithms, and fuzzy sets and systems.

Dr. Pal received an NRC-NASA Senior Research Award to work at the NASA Johnson Space Center, Houston, TX in 1989. He also received the 1990 Shanti Swarup Bhatnagar Prize in Engineering Sciences, the 1993 Jawaharlal Nehru Fellowship, the 1993 Hari Om Prerit Vikram Sarabhai Research Award, the 1993 NASA Technical Brief Award, the 1994 IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award, and a Patent Award from NASA, in 1995. He is a co-author of the book *Fuzzy Mathematical Approach to Pattern Recognition*, (New York: Wiley, 1986) which received the Best Production Award in the 7th World Book Fair, New Delhi, and a co-editor of the books *Fuzzy Models for Pattern Recognition* (New York: IEEE Press, 1992), and *Genetic Algorithms for Pattern Recognition* (Boca Raton, FL: CRC Press, 1996). He is a Fellow of the Indian National Science Academy, National Academy of Sciences, India, the Indian Academy of Sciences, the Indian National Academy of Engineering, Institute of Engineers, India, and the IETE; an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS, *Pattern Recognition Letters*, *Neurocomputing*, *Applied Intelligence*, *Information Sciences (C)*, *Far-East Journal of Mathematical Sciences*, an Executive Advisory Editor for the IEEE TRANSACTIONS ON FUZZY SYSTEMS, and the *International Journal of Approximate Reasoning*.