X-tron: An Incremental Connectionist Model for Category Perception

Jayanta Basak and Sankar K. Pal, *Fellow, IEEE*

Abstract— **A connectionist model for categorization (self-organization) even in the presence of multiple or mixed patterns has been presented. During self-organization, the network automatically adjusts the number of nodes in the hidden and output layers, depending on the complexity or nature of overlap between the patterns. An ambiguity measure is given based on how well the features are being interpreted by the network. From the ambiguity measure a certainty factor about the decision of the network is derived. The effect of noise on the certainty factor is investigated. A vigilance threshold is used to decide whether the network's decision is correct or not. Functionally the network consists of two parts, one of them categorizes the incoming patterns and the other monitors the performance of categorization. The characteristics of the model has also been demonstrated experimentally on both one-dimensional binary strings and image patterns even when they are corrupted by additive, subtractive, and mixed noise.**

I. Introduction

ALL biological systems exhibit self-organizing property, where the entities are grouped into categories automatically without the help of any external teacher. Several investigations have been made on the self-organizing behavior of connectionist models. Linsker [1], [2] analyzed the self-organization property and applied it to model the development of early visual processes in retina. Kohonen [3] used the concept of self-organization to produce topologically correct feature maps. Here in the learning process the weights over a neighborhood change. Adaptive resonance theory [4] was developed to categorize incoming patterns where a vigilance factor was defined to monitor the belongingness of an input pattern to a particular category. Both fast and slow learning processes have been incorporated in the self-organizing model. The adaptive resonance theory was extended to categorize analog input patterns to form stable category codes [5]. The characteristics of such networks for additive and subtractive noise have also been studied [6], [7]. Later another model, namely, ARTMAP (adaptive resonance theory-supervised predictive mapping) [8], was developed which uses two different ART modules (one of them acts as a predictive system) linked by an associative learning network. An internal controller ensures autonomous system operation in real time. ARTMAP acts like a self-organizing expert system that calibrates the selectivity of its hypotheses based upon the predictive success. Recently the ARTMAP has been extended to fuzzy ARTMAP [9] to handle the imprecise or vague information in the input patterns.

Amari [10], [11] developed a self-organizing model for concept formation and the corresponding orthogonal and covariance learning techniques. Kosko [12] developed a theoretical groundwork for unsupervised learning under noise. Fukushima [13] has developed an application-specific model, called neocognitron, which can also exhibit the self-organization behavior. Several other application-specific self-organization models have also been developed. Ritter and Kohonen [14] produced self-organizing of abstract data, such as words. The semantic relationships in the data were reflected in these maps. Minnix *et al.* [15] used the underlying concept of neocognitron [13] and developed a self-organizing network model to recognize objects with translation invariance. Marshall [16] used self-organization for adapting the network to long-term characteristics of the visual motion of an object.

The concept of self-organization in connectionist models is analogous to the idea of clustering in pattern recognition [17]. In general, the idea is to select a seed for some category so that its distance (which can be Euclidian or Mahalanobis or city-block or any other suitable measure) from any incoming pattern can be measured. Some objective function based on this distance is then used to decide whether the pattern belongs to that category or not. This very concept was used in most of the self-organizing networks mentioned before. In our problem more than one category (in mixed form) can be present at a time in a given input. In that case, it would be rather befooling to compare any single category with the incoming pattern when a particular combination of more than one category is comparable (similar) to the input. Therefore, instead of using the conventional concept of clustering in pattern recognition, the concept of similarity-based induction as posed by Stanfill and Waltz [18] can be used to measure how well the input feature train is being interpreted with the existing set of categories (either individually or in a combination).

In literature there exist several investigations for mixed category recognition. Peng and Reggia [19] developed a model for prediction of multiple disorders for a given set of manifestations. In that model, however, no learning scheme was proposed. Later Cho and Reggia [20], [21] developed a learning scheme through competition and cooperation process for this kind of tasks. They mathematically formulated the competitive and cooperative process and then derived an error backpropagation learning rule for such tasks. The learning rule, however, operates in supervised mode. Marshall [22]–[24] developed a scheme for mixed category recognition. He confined

Manuscript received Feb. 23, 1993; revised March 3, 1994.

The authors are with the Machine Intelligence Unit, Indian Statistical Institute, Calcutta 700 035 India.

IEEE Log Number 9409358.

the competition process within the nodes of similar nature, i.e., getting activations from inputs which have sufficient overlap. Cohen and Grossberg [25], [26] used the concept of masking field for the recognition of mixed categories. The use of masking field enables the system to find out embedded patterns. It also allows multiple nonoverlapping patterns to be classified simultaneously. Later, Nigrin developed a neural network model, namely, SONNET [27]–[30] which is able to self-organize in presence of mixed categories and form stable codes. He introduced a concept of competition between the links for associating a category with a feature.

The present paper describes a three-layered connectionist model for exhibiting self-organizing behavior even in the presence of mixed patterns (mixed pattern may result from presence of more than one objects or occluded objects). The model is based on the network [31] developed for recognizing multiple objects with supervised mode of learning. The underlying concept of the present self-organizing system is as follows. Whenever a feature train appears at the input the network is allowed to settle with its existing links. After settling, the output categories feed back the activation to the input features. For each feature some kind of ambiguity is then measured depending on the external input and the amount of feedback. On that basis, a certainty factor is derived to monitor if the incoming pattern is a new category or a combination of the existing categories.

In the process of self-organization, whenever a new category is detected an output node is allocated for the category. Moreover, to incorporate the knowledge about input–output association of the new category, some hidden nodes are also added to the network. As a result, the network incrementally changes the size of the layers (hidden and output) to accommodate the new unknown categories. The network is named as "X-tron" where X stands for unknown category.

X-tron achieves the desirable property of recognizing mixed patterns. It is able to self-organize and form stable codes. The problem we considered is not finding out a pattern embedded in different patterns, as performed by Cohen and Grossberg [25], [26] or Nigrin [27]–[30]. Rather, the network would be able to decide if more than one learned category is present in the input even when there is a significant amount of overlapping between the patterns. For example, suppose that the model has learned two categories CAB and ABD. In that case, if a new pattern CABD is presented to the network, it would be able to decide that the new pattern is a mixture of these two learned categories. X-tron is able to continually form stable category codes. It accepts confidence values about the features and produces the confidence values about the respective categories. Here, the confidence value means the level of firmness about the presence of a feature or category (object). For example, if the confidence of an entity (feature or category) is 0.0, then the entity is absent. If the confidence is 1.0, then it is present. If the confidence is 0.5, then no decision can be taken about its presence or absence. This indicates that the system can handle imprecise or vague information (which is also incorporated in fuzzy ARTMAP, but here it is dealt in a different way). This capability also enables the system to handle noisy (additive and subtractive) patterns. Moreover, X-tron incrementally adjusts its size (number of nodes and links) depending on the patterns and the nature of their overlap.

The paper is organized as follows: A brief description of a network architecture [31] for supervised learning is presented in Section II. In Section III, some properties of the network are analyzed which is a must for formulating the methodology and designing the architecture for performing the self-organizing tasks. Section IV presents the proposed architecture and the categorization technique including noise characteristics, certainty factor, merits, and stability issues. Simulation results are presented in Section V. Section VI presents the conclusion.

II. Network Architecture for Supervised Learning

The network which works under supervised mode is presented in Fig 1. It has three layers, namely, input, hidden, and output. The input layer accepts patterns in the form of numerical values in [0, 1] indicating the confidence levels regarding the presence of features; the output layer represents the confidence levels regarding the presence of output categories while the hidden layer represents the association between the input and output nodes. With each input node a set of hidden nodes is connected. On the other hand, a hidden node can be connected only to a single input and a single output node. Each hidden node represents the association of the input and output nodes to which it is connected. Input nodes are connected to the hidden nodes by links of fixed weights. On the other hand, each hidden node is connected to a particular output node by bottom-up and top-down links. The bottom-up link carries the activation from the hidden node to the output node, and the top-down link carries the activation from the output node back to the hidden node. Each hidden node consists of two parts. One of them holds the feedback activation coming from the output layer through the top-down links. The other competes with the neighboring hidden nodes which are connected to the same input node.

The input layer accepts the confidence values of the features, and the output layer produces the confidence values of the categories (or the objects). The term confidence value indicates the extent of firmness regarding the presence of feature or category (object). Whenever an input pattern is presented to the network, it propagates to the hidden layer and then through the bottom-up links the activations reach the output layer. This is the initial set of outputs (O). The output layer, in turn, feeds back the activations to the hidden layer. Each hidden node is activated for a particular input–output combination. The hidden nodes connected to a common input node compete with each other. The winner-take-all node physically represents the strongest possible hypothesis supporting that particular input feature. The difference in activations (e) of the input node and the WTA hidden node is again propagated through the bottom-up link connected to the WTA node. Each output node has a negative self-feedback. The feedback ensures that in the process of updating, the activation of an output node automatically decreases if it does not get support from its constituent features. If an output node gets proper support from its constituent features (i.e., the bottom-up differential

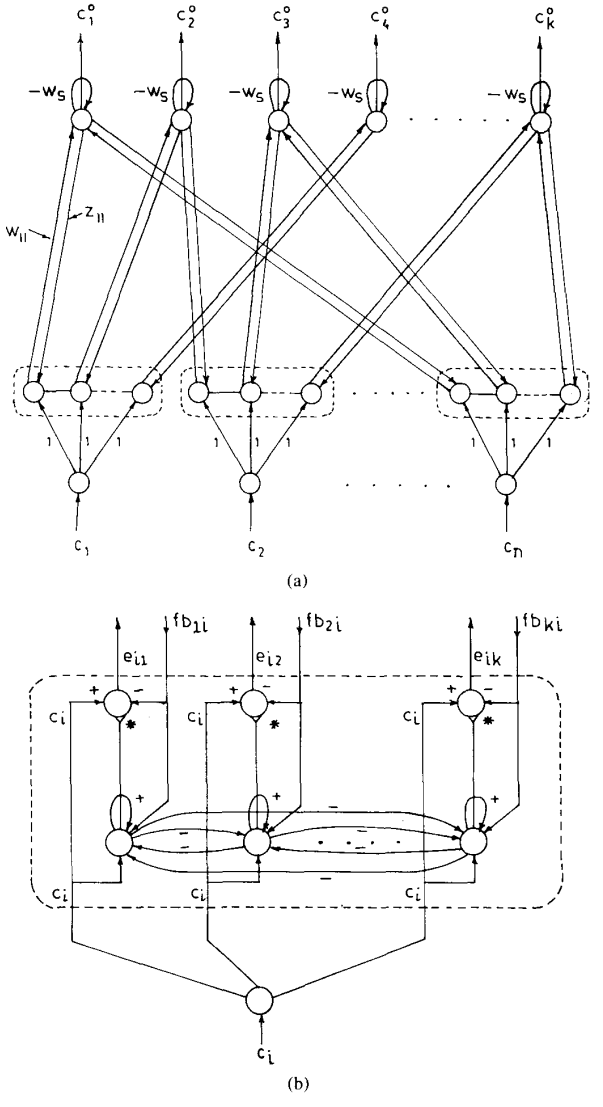

Fig. 1. (a) Structure of the connectionist model for supervised learning and recognition. The hidden nodes connected to the same input node are enclosed in a dashed box. (b) Structure and connections of the hidden nodes in the network. fb_{li} is the feedback signal to (i,l)th hidden node from the lth output node (i.e., $fb_{li} = z_{li}v_l$). (Note that the index of hidden nodes is varied from one to k for the sake of simplicity in representation. Actually, only those hidden nodes are present for which the corresponding objects have the feature i.) e_{il} represents the differential support which is either zero or $c_i - b_i$, where $b_i = \max_l(fb_{li})$. The symbol * denotes the modulating signal appearing through the corresponding link. Each hidden node is basically a conjugation of two nodes (one in the lower level and other in the higher level). In the lower level within the box, the nodes compete and the output of each node modulates the activation of the corresponding node in the higher level within the box. e_{il} is zero if the lower part of the (i,l)th is a loser one, and it will be $c_i - b_i$ if the lower part is a winner. The weights for competitive connections are not shown here.

support (e) and negative self-feedback cancel each other) then its activation will stabilize to some nonzero value. In Section IV-B we will discuss how the activation level depends on the amount of self-feedback. The network is said to have recognize an object only when the activation level of the corresponding output node, after stabilization, is greater than some predefined threshold (this is referred to in Section III). Now onwards, recognition of some object will mean only recognition after stabilization.

Let us now describe the STM and LTM equations. The details of convergence of STM and LTM equations have been reported earlier [31]. For the convenience of the reader, however, we are presenting it below in brief.

Equations for STM: The states of the output nodes are updated according to the differential equation given as

$$\frac{du_l}{dt} = \sum_{i=1}^{n} w_{il}e_{il} - w_s v_l \tag{1}$$

where u_l and v_l are the total input to and output of the lth output node. w_{il} is the weight of the bottom-up link from the (i,l)th hidden node (connecting ith input node and lth output node) to the lth output node. w_s is the weight of the self-feedback in the output layer. e_{il} (differential support) measures the difference of the input activation from ith input node and the feedback from lth output node, provided the (i,l)th hidden node is enabled (winner-take-all node). Mathematically

$$e_{il} = \begin{cases} c_i - z_{li}v_l & \text{if } z_{li}v_l \geq z_{mi}v_m \\ 0 & \text{otherwise} \end{cases} \qquad \forall m \neq l$$

where c_i is the activation at the ith input node (i.e., confidence value about the presence of ith feature). z_{li} is the weight of the top-down link from the lth output node to the (i,l)th hidden node. The output v_l is related to the instantaneous input u_l by a semilinear nondecreasing gain function $g(\cdot)$ (chosen as an S-function [31]).

Let an energy function $\mathcal{E}(t)$ be defined as

$$\mathcal{E}(t) = \tfrac{1}{2}\sum_{i=1}^{n} \lambda_i'(\max_{l=1,\cdots,k} z_{li}v_l - c_i)^2 + \tfrac{1}{2}w_s\sum_{l=1}^{m} v_l^2 \tag{2}$$

where n is the number of input nodes, and m is the number of output nodes. λ_i' is a multiplication factor such that

$$\lambda_i' = w_{is}/z_{si} \quad \text{if} \quad z_{si}v_s = \max_{l=1,\cdots,m} z_{li}v_l.$$

The rate of change of energy can be written as

$$\frac{d\mathcal{E}}{dt} = \sum_{l=1}^{m} \frac{\partial \mathcal{E}}{\partial v_l}\frac{dv_l}{dt}. \tag{3}$$

But

$$\frac{\partial \mathcal{E}}{\partial v_l} = -\sum_{i=1}^{n} z_{il}\lambda_{il}\delta_{il} - w_s v_l$$

i.e.,

$$\frac{\partial \mathcal{E}}{\partial v_l} = -\frac{du_l}{dt}.$$

Therefore (3) can be written as

$$\frac{d\mathcal{E}}{dt} = -\sum_{l=1}^{k} g^{-1'}(v_l)\left(\frac{dv_l}{dt}\right)^2. \tag{4}$$

From (4) it is evident that $d\mathcal{E}/dt \leq 0$ for all $t \geq 0$. As $t \to \infty, d\mathcal{E}/dt \to 0$, and thereby the system converges to an energy minima (local).

The energy function (2) reveals the fact that the system always tries to minimize the error of mismatch between the input confidence values and the interpreted confidence values of the output layer. The weights of the top-down links are set in such a way that z_{li} is zero if ith feature do not belong to object, and close to unity if it belongs to. As a result, the first part shows the similarity of the problem to the "set covering" problem. The second part of the energy function ensures the fact that the feature train presented to the network should be interpreted by the minimum possible number of objects. This enables the network to reduce the number of redundant objects or the chance of false alarming.

Equations for LTM: The learning rules or the rules for iterative adjustment of the weights are set in such a way that the weight of each link asymptotically reach a predefined measure. The measure for each weight is defined in such a way that it achieves the ability to capture the relative frequency of appearances of the corresponding feature object pairs. From the expression of the energy function (2), it is intuitively seen that the weight of a top-down link (z_{li}) should be proportional to the probability of appearance of the corresponding feature (i) with respect to the corresponding object (l). On the other hand, the value of λ_i' should be proportional to the probability of appearance of the object (corresponding to the winner) with respect to feature i. Therefore, the asymptotic values can be given as

$$z_{li} = p(f_i|o_l) \tag{5}$$

and

$$\lambda_{il} = p(o_l|f_i). \tag{6}$$

Note that we define a quantity λ_{il} to represent λ_i' for all links. The weight of bottom-up links should take a form

$$w_{il} \propto \lambda_{il} z_{li}. \tag{7}$$

In the present work, since the transfer function of the output nodes are chosen as an S-function, the output values always saturate if the total input activations exceed unity. The weights of bottom-up links are iterated in such a way that the total activation reaching an output node is always less than unity. Therefore, an additional constraint is imposed on the weights of the bottom-up links which is

$$\sum_i w_{il} \leq 1. \tag{8}$$

Moreover, if two objects (say A and B) are such that the feature set of one object (say A) is a subset of another one (say B) and the probabilities of appearances of both the objects are the same then both A and B would be fully active if the larger feature set (corresponding to B) is presented to the network. In that case, it would not be possible to decide that a single object has been presented to the network. This problem can be taken into account by using Weber's law (as presented in adaptive resonance theory [4]). Considering (7) and (8) and Weber's law, the asymptotic measure for the weights of bottom-up links becomes

$$w_{il} = \frac{p(o_l|f_i)p(f_i|o_l)}{\gamma + \sum_{i=1}^{n} p(o_l|f_i)p(f_i|o_l)}. \tag{9}$$

The constant γ is used to get the effect of Weber's law.

The weights of the links in the network are changed in such a way that they become equal to the measures after sufficient number of learning trials. In other words, the learning rules should be such that the weights of the bottom-up and top-down links asymptotically reach the measures (5) and (9). The conditional probability values are approximated by the ratio of the number of appearance of the features and the objects. The detailed derivation of the learning rules is presented in [31]. The learning rules are

$$\frac{dw_{il}}{dt} = \left(\alpha_i \delta_l z_{li} + \alpha_l^o \left(\frac{w_{il}}{z_{li}}\right)\right) c_i y_l - (\alpha_i c_i + \alpha_l^o y_l) w_{il} \tag{10}$$

$$\frac{dz_{li}}{dt} = \alpha_l^o y_l (c_i - z_{li}) \tag{11}$$

where α_i and α_l^o are the agility factors of the ith input node and the lth output node. The agility factor determines the capability of learning of the links connected to that node. The higher the agility factor, the higher will be the rate of learning and vice versa. Initially, the agility factor of all nodes are set to unity and they are decreased with the learning trials. The agility factor of a hidden node is the same as that of the input node connected to it. The agility factors are changed according to the following rules

$$\frac{d\alpha_i}{dt} = -\alpha_i^2 c_i \tag{12}$$

$$\frac{d\alpha_l^o}{dt} = -{\alpha_l^o}^2 y_l. \tag{13}$$

The value of y_l is the desired output value at the lth output node. The desired output is determined by the monitor network (if supervised mode of learning is used then it is supplied externally. In the present work, however, it is determined by the monitor network). If the monitor network finds a pattern already known to the network, it sets the desired output of the corresponding node(s) to be unity and all other nodes to zero. If it finds a new pattern then sets the desired output for the newly created node to unity and all other nodes to zero. The value of δ_l is given as

$$\delta_l = \frac{\varepsilon_l}{\gamma g'(u_l)} \tag{14}$$

where $\varepsilon_l = y_l - o_l$ measures the difference of the actual output at the lth output node from its desired value (as determined by monitor network). γ is the constant used in the asymptotic measure (9) which also controls the rate of learning (as expressed in the learning rules). The LTM equations for top-down links are similar to those presented by Grossberg [4], [32].

The weights of the bottom-up links are initially set to zero. The agility factors of all input and output nodes are initially set to unity. Note that the agility factor of a hidden node is

the same as that of the corresponding input node to which it is connected. Whenever a top-down link is created (to connect a newly created hidden node with the corresponding output node) its weight is set equal to the activation level of the corresponding input node (to which the hidden node is connected) i.e., the confidence value of the corresponding feature. With these initial conditions the weights are iteratively changed (in the learning process) so that they reach the asymptotic measures. In the learning process, the weights are updated whenever a pattern appears at the input (and no batch mode learning is followed). The agility factors control the rate of learning. Initially the rate is high so that the network can quickly guess the exemplar code of a category. The learning rate decreases with time, as the age of the node increases (i.e., the network has already gained the knowledge about the exemplar pattern).

Some Remarks: The asymptotic values of the bottom-up links takes care of Weber's law. (The way of implementing Weber's law has been explained by Carpenter and Grossberg in the adaptive resonance theory [4]). Therefore, if two categories are such that the feature set of one category is a proper subset of the other, then also the network would be able to identify these two categories separately. This phenomenon will be explained in detail in Section IV-F.

As mentioned before, the network is able to recognize mixed patterns. It also tries to reduce the redundancy in the decision. For example, assume that the network has learned three patterns say, ab, cd, and $abcd$. In that case, if $abcd$ is presented as an input pattern, only the node corresponding to $abcd$ will be activated (though the other patterns ab and cd are proper subset of $abcd$, nodes corresponding to them will not be activated). The reason is as follows.

Whenever $abcd$ is presented to the network, the initial activations of all three nodes will be high (the rest of the output nodes will have low activations considering that no other category has overlap with $abcd$). But as the Weber's law is considered in the asymptotic measures of the weights of the links, the initial activation value of the node for $abcd$ will be higher than the remaining two (i.e., $O_{abcd} > O_{ab}$ and $O_{abcd} > O_{cd}$). The output nodes feed these activation values to the hidden layer. The hidden nodes which are connected to the same input node compete between themselves, i.e., competition occurs for associating a feature with certain category and not between the categories themselves. Therefore, the hidden nodes connected to the node for $abcd$ will receive higher feedback than the hidden node receiving feedback from ab or cd. The competition in the hidden layer occurs only in the feedback activations [this is also clear from the expression $\max(\,)$ in (2)]. As a result, the hidden nodes connected to the output node for $abcd$ will win. In the settling process since an input node sends activation only through the winner-take-all hidden node, output node corresponding to $abcd$ will receive support from input. The other two nodes (ab and cd) will not receive any support because they have only loser hidden nodes (1). Since the output nodes are also associated with a negative feedback, the activation values of ab and cd will gradually diminish while that of $abcd$ will reach a stable condition.

Note that the network is not designed to learn the embedded patterns which is performed in SONNET [28]. Rather, it takes the advantage of dissociating the competition process from the activation values in the output layer. The competition takes place for associating a feature with some object which is performed in the hidden layer. This enables the network to recognize patterns even under high amount of overlapping. For example, let us consider that the network has learned two categories abc and bcd. Then if the network is presented with a pattern $abcd$ then also it will be able to recognize them separately. Of course, the common subset of features bc will be associated to one of the categories (say, bcd) (due to competition in the hidden layer), but there would be no inhibition from any of the hidden nodes. Due to the differential support from the rest of the features (i.e., a) the activation level of the category abc will stabilize to a high activation value.

Let us now consider another example where the network has learned only three categories say, ab, abc, and cd. If a new pattern $abcd$ is presented to the network then it will recognize it as a combination of abc and cd. This is due to the fact that initial activation of abc would be higher than that of ab and consequently ab will have only loser hidden nodes in the settling process. (Here the present system differs from the system developed by Marshall [22], [23] where ab and cd would be recognized.) The expression for the activation level in the output layer is presented in Section IV-B.

III. Properties of the Network

In this section some new properties of the aforementioned network have been established, based on which the proposed theory of categorization (or self-organization) has been developed. Before proceeding to the logical formulation of these properties, a couple of definitions are provided. Before proceeding to the analysis of the behavior of the network a couple of definitions are to be considered.

Definition 1: The network is said to have learned an object class p if and only if the change of weights of the bottom-up and top-down links for the corresponding input feature set is less than some threshold ϵ, whatever small it may be.

Definition 2: The network is said to have recognized an object class p if and only if the output of the corresponding node in the output layer is greater than some threshold T.

The way of selection of T will be discussed in Sections IV and V.

Before proceeding to the design of actual categorization (or self-organization) methodology, we must ensure that the network is able to recognize an object if it is claimed that the network has learned that object. Because only in that case the network would be able to proceed into further categorization correctly. Since the present work is aimed at categorization of more than one objects simultaneously (or mixed patterns), we must, at first, ensure that the network is able to correctly recognize the individual categories constituting the mixed patterns. The following theorems reveal some properties related to the recognition ability of the network in presence of mixed patterns. We have tried to prove the ability of recognition by inductive reasoning.

Several notations are consistently followed in these theorems. An object class is denoted by C. The feature sets are denoted by F while the individual features have been denoted by f. The initial outputs (before stabilization of the network) are denoted by O. Note that if a network learns some object class, then the initial output of the node corresponding to that class will be maximum among all the output nodes whenever a pattern from that class is presented to the network. This is due to the fact that the dot product of the pattern vector and the corresponding weight vector will be maximum (this was the basis for designing the learning rules).

Theorem 1: If the network is presented with one of its learned templates at the input layer, then it will recognize only the category corresponding to that template.

Proof: Let us assume that the network has learned object class C_p. The value of ϵ has been chosen sufficiently small. Let a sample from the object class C_p be now presented to the network. The initial output of the node corresponding to C_p is greater than the output of any other node. Mathematically, $O_p > O_i, \forall i \neq p. O_p$ denotes the initial output of the corresponding output node before stabilization.

Let an arbitrary input feature f_k belong to the object p. In the set of hidden nodes which are connected to the node corresponding to feature f_k, the node connected to pth output will have maximum feedback (provided the feature is not noisy, i.e., the weight of the top-down link is very close to unity). Therefore the particular hidden node wins over the other hidden nodes in the competition. In the recognition process the winner-take-all hidden node will send differential activation (e) to object p only. All other objects sharing the feature f_k will not get any active support from it. Similar phenomena will occur for all other features belonging to object p. By this process the outputs of objects other than p will decrease more and more due to self-negation and absence of any active support from the feature level. Moreover, the features which do not belong to the object p will negate the activations of spurious objects. Only the object p has both self-negation and some positive activation from feature level. The other objects have only self-negation. As a result, the response of object p will stabilize to some positive value while the responses of other objects will come down to zero. Therefore, if the threshold T is selected suitably then the object p can be claimed to have been recognized. □

Theorem 2: If the network is presented with two of its learned templates then it will recognize at least two categories.

Proof: Suppose two objects a and b have been presented to the network. Let F_a and F_b be the respective feature sets of the objects classes a and b. There can be three different cases:

- There does not exist any other object whose feature set is a subset of $F_a \cup F_b$.
- There exists no other object c for which $O_c > O_a$ or O_b.
- There exists at least one object c such that $F_c \subset F_a \cup F_b$ and $O_c > O_a$ or O_b.

In the first case, it is evident that the outputs corresponding to a and b will be stable to some positive value. But since there exists no other object dependent only on the subset of the features, the outputs of other objects will get some negation from the feature level itself. Therefore the outputs of the spurious objects will reduce to zero.

In the second case, even if there exists some object which is dependent entirely on a subset of the features present at the input, its initial output is less than those of the true objects present.[1] Therefore the spurious objects will share no active feature and consequently, the support from the feature level will be zero. Due to self negation, the outputs of the spurious objects will decrease and eventually reduce to zero.

In the third case, the initial output of the spurious object c is greater than those of the true objects. Therefore the spurious object will share some active features. Again, the true objects will have some active features. Therefore the true objects and the spurious object(s) will coexist in the final output. □

Now we will consider a property regarding the recognition of mixed objects.

Theorem 3: Suppose, a network can recognize a set of n objects simultaneously (i.e., input pattern is the overlapped feature set corresponding to this n objects). If another object (i.e., the $(n+1)$th object), now presented to the network, is such that

- It has some features not belonging to either of the n objects, and
- There exists no other object having a feature set which is a subset of the union of all the features of $(n+1)$ objects and whose initial output is greater than any of those of the $(n+1)$ objects,

then the network will be able to recognize only these $(n+1)$ objects simultaneously.

Proof: The network is capable of recognizing any set of n objects perfectly. Therefore there does not exist any other object p such that F_p is a subset of $F = \cup_{i=1}^{n} F_i$ and $O_p > O_1$ or O_2 or O_3 or $\cdots O_n$. Because in that case the object p will share some of the active features and p may also coexist. The system will not then recognize exactly n objects.

Again there cannot exist any other object p such that F_p is a subset of the union of F_{n+1} and the feature sets of any other $n-1$ or less number of objects, and its initial output is greater than that of any of these objects. Because in that case if only those objects were presented to the network (number of objects in that case is less than or equal to n) the network could not be able to eliminate the object p, which violates the basic condition.

There can be another case. The object p is such that $F_p \not\subset F$ but $F_p \subset F \cup F_{n+1}$, and $O_p > O_1$ or O_2 or O_3 or $\cdots O_{n+1}$. In that case the object p will share some of the active features and in the stable state the object p will coexist. On the other hand, if the initial response of the object p is less than those of the other objects present, then no spurious objects will be detected. □

Theorem 4: The network is capable of recognizing any set of n objects exactly if the objects and the corresponding feature sets are such that there exist no other object p for which

- $F_p \subset \cup F_{k_i}$ where k_i can be anything in the entire object set and
- $O_p > O_{k_i}$ for some value of k_i

[1] The word "true" means only those objects which are presented to the network. In the present case "true" objects are a and b.

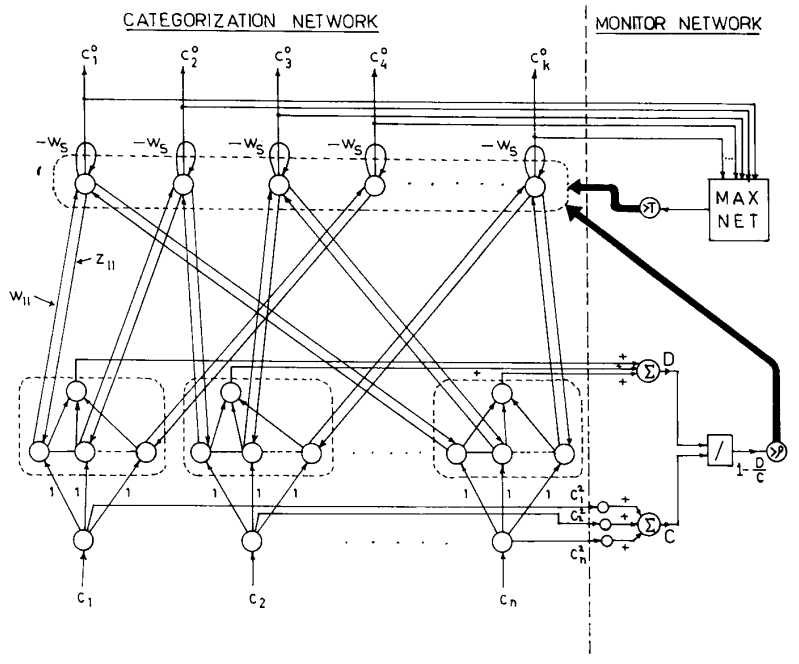

Fig. 2. (a) Structure of the connectionist model for self-organization. Bold lines represent the control paths from the monitor network to the categorization network. Another node is added in each box of hidden nodes which measures the ambiguity at the corresponding feature (considering the set of hidden nodes). Two adders are used in conjunction with hidden and input layers to compute D and C respectively. The node denoted as "/" computes the certainty factor $1 - D/C$. (b) Structure and connections of the hidden nodes in the network. The symbols have similar meanings as in Fig. 1(b). $c_i e_{il}$ measures the ambiguity for an individual hidden node which is either zero (if e_{il} is zero, i.e., loser hidden node in the competition) or equal to $c_i\,(c_i - b_i)$ (only for the winner-take-all hidden node). Note that the output of adder will also be $c_i\,(c_i - b_i)$.

Proof: The statement is true for two objects (Theorem 2). Since this is true for two objects it is true for three objects (Theorem 3) and so on. It is, therefore, true for any set of n objects. □

IV. Categorization

Depending on the properties of the network described in Section III, the theory of categorization (self-organization) and the corresponding architecture have been developed here.

A. Overall Methodology and Categorization Architecture

The underlying concept of the proposed categorization technique is as follows: Whenever a pattern (either single category or mixed categories) appears, it produces output

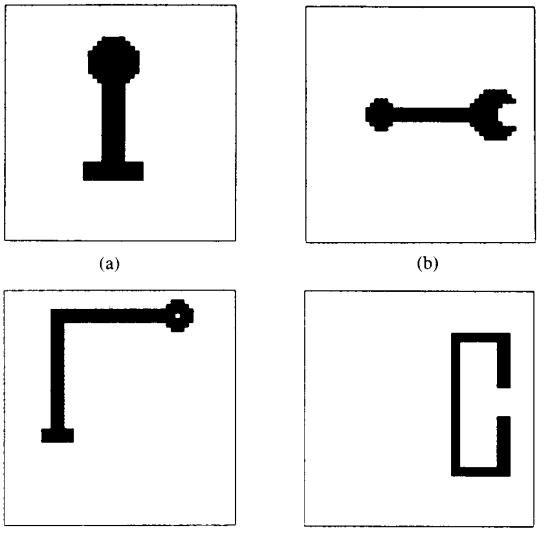

Fig. 3. (a)–(d) represent the objects 1, 2, 3, and 4, respectively, used as visual input pattern.

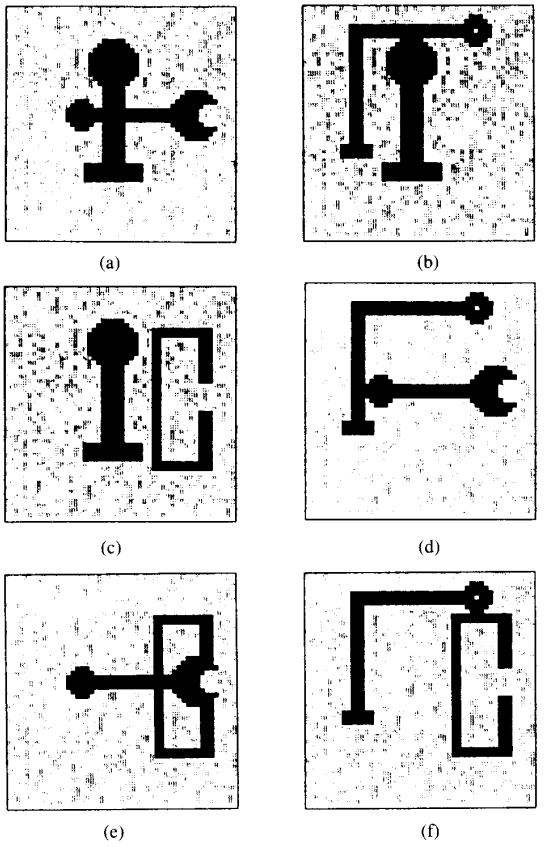

Fig. 4. Mixed patterns with 20% noise level. (a)–(f) represent the mixture of objects 1 and 2, 1 and 3, 1 and 4, 2 and 3, 2 and 4, and 3 and 4, respectively.

responses for the categories which are already learned. These outputs actually yield a measure indicating how well the feature set is being interpreted by the network. If the activation to a particular input node does not match with the support received from the output categories, then there will be an ambiguity corresponding to that feature. If a single pattern or a set of mixed patterns from a new class appears, the features will not be fully interpreted by the learned categories. If the total ambiguity for all the features is greater than some threshold, then the pattern presented to the network is considered to belong to a new category. In designing the architecture for such self-organization (categorization), a portion of the network must therefore be able to monitor the performance of categorization and to determine the ambiguity value.

In the architecture for categorization, an extra network is added to monitor the performance of the network for some given pattern. The network architecture is shown in Fig. 2. With each hidden node there is another node which computes the ambiguity at that particular node. These ambiguities and the input activations are propagated to the monitor where the certainty factor is calculated. The certainty factor is considered to have a linear relation with the total ambiguity. If the certainty factor is greater than some threshold, then the pattern is considered to be already present, and the weights of the links are iterated. If the certainty factor is less than the threshold, then a new output node is allocated for the input pattern. This threshold is called the vigilance threshold (ρ) which is supplied externally. This vigilance threshold is analogous to the vigilance factor used in adaptive resonance theory [4].

In the monitor network there is another part which checks the maximum activation present at the output layer. The maximum activation is checked if it is greater than a threshold T. If the condition holds true then only the network is allowed to learn the pattern present at the input. Otherwise, even if the certainty factor is greater than the vigilance threshold the learning is not allowed. Whenever a new output node is allocated in the output layer the upper part of the monitor network is activated and the network learns the new pattern.

The method of self-organization works as follows.

Step 1) Present a new pattern. The pattern may be representative of a single category or may be caused by the presence of more than one category.

Step 2) Measure the certainty factor (CF). Measure the maximum output (x) in the output layer. Note that there does not exist any output node initially. In that case, CF and x are considered as zero. The monitor network measures these two factors.

Step 3) If CF $> \rho$ and $x < T$, then goto Step 7).

If CF $> \rho$ and $x > T$, then make all the output nodes whose activations are greater than T fully active, and goto Step 6).

Step 4) Allocate a new output node in the output layer. If the total number of output nodes is greater than the capacity of the network, then exit.

Step 5) Allocate hidden nodes for having input–output associations corresponding to the features which are present at the input. Connect each newly created hidden node to the corresponding input node (with a link of unit weight) and to

the newly created output node (with bottom-up and top-down links). Initialize the weights of the newly created links i.e., weights of the bottom-up links are set either to zero or to a small value. The weights of the top-down links are set equal to the activation level of the corresponding input node (i.e., $z_{li} = c_i$ where l is newly created output node and i is the input node). Make the newly created output node fully active. (The allocation of nodes enables the network to incrementally adjust its size i.e., number of nodes and links depending on the input train of patterns. The dynamic allocation of nodes has some similarity with the concept introduced by Platt [33].)

Goto Step 7).

Step 6) If a single node is activated in the output layer, check if there exists any input node for which there is no highly activated hidden node (which indicates that the corresponding feature did not appear in the previous examples). If any such input node exists then create a hidden node to represent the association between the input node and the activated output node. Connect the hidden node with the input node (with a link of unit weight), to other hidden nodes connected to the same input node (with inhibitory links), and to the activated output node (with bottom-up and top-down links). Initialize the weight of the bottom-up link to zero or a very small value. Initialize the weight of the top-down link equal to the activation value of the input node.

Step 7) Learn the weights of the links, i.e., iterate the weights till they converge (become less than some threshold ϵ(say)).

Step 8) Present another new pattern. Goto Step 2).

The categorization process always takes an input pattern and self-organize accordingly. Therefore, it is able to do its task with as many number of input patterns so long as there is no restriction about the size of the network. It always self-organizes in on-line mode and does not consider any batch-mode operation.

B. Output Response

The response of an output node is now considered. Let each input feature be represented as a tuple (c_i, w_i, z_i). Here c_i is the confidence value of the feature. w_i and z_i represent the weights of the bottom-up and top-down links from the ith feature to the particular output node and from that output node to the ith feature, respectively (here we suppress the subscript for the output node because an arbitrary node response has been considered). Suppose the output of the node stabilizes to a value x. Let the negative self-feedback of each output node be w_s. Without loss of generality, we can consider all the features to be active, i.e., sending differential activation (e) to the output node. If some features are shared by other objects and they really do not send any activation to the output nodes, then that triplet can be omitted from the set. Under stable condition, the positive and negative signals at the output node will cancel each other. Mathematically

$$w_s x = \sum_{i=0}^{n} (c_i - z_i x) w_i. \tag{15}$$

By simple algebraic manipulation it can be shown that

$$x = \frac{\sum_{i=0}^{n} w_i c_i}{w_s + \sum_{i=0}^{n} w_i z_i}. \tag{16}$$

When the network has learned a particular category, the value of $\sum_{i=0}^{n} w_i z_i$ is nearly unity (the learning rules are designed in such a way). If w_s is small enough and all c_i values are nearly unity, then the output x will reach unity. Depending on the value of self inhibition w_s the value of T can be selected.

Example 1: Consider an object for which all w_is (at some stage of learning) are equal to w. Let the object have k features each of equal importance to the objec, and the weights of all top-down links be unity. Then the output is given as

$$x = \frac{kw}{w_s + kw}.$$

If only k_1 features are active and the rest are shared by others so that they cannot send any active signal, the output becomes

$$x_1 = \frac{k_1 w}{w_s + k_1 w}.$$

To recognize the object, even for k_1 features being present at the input, we must have

$$\frac{k_1 w}{w_s + k_1 w} > T$$

or

$$w_s < k_1 w \left(\frac{1}{T} - 1 \right).$$

Consider $kw = K$, then the above equation becomes

$$w_s < \frac{k_1}{k} \left(\frac{1}{T} - 1 \right) K.$$

If the network has to recognize the object even for 10% of the active features then (considering $K = 1.0$) w_s becomes

$$w_s < 0.1 \left(\frac{1}{T} - 1 \right)$$

or

$$T < \frac{1}{1 + 10 w_s}.$$

If $w_s = 0.02$ then $T < 0.83$, i.e., a threshold of 0.8 will work well enough to recognize objects under heavy occlusion.

Note that, in the network design the value of w_s should be selected in such a way that the network does not take long time to stabilize and the output threshold T is not too low.

C. Principles of Ambiguity Measure

It is clear from the discussion made so far that the network will produce some stable output for a set of active features. For a particular output whether the object will be considered to have been recognized or not will depend on the design of the system. But if an object is present in the scene and the network has learned the object, then the features must get proper support from the output layer. This is the very basic concept over which the categorization (self-organization) strategy has been developed.

For example, consider the previous case (Example 1) where an object has k features each connected to the corresponding output node by a bottom-up link of weight w and a top-down link of unit weight. Then the output will be $kw/(w_s + kw)$. Therefore each feature will get the same support. If w_s is small enough the input confidence value will be almost equal to that of the top-down support (since output response and, therefore, the top-down support are very near to unity, and the input confidence value is considered to be unity). If some features are shared by some other objects the mismatch between the input confidence and the top-down support will increase. Ideally, in a noiseless scene there should be no mismatch between the input confidence and the top-down support corresponding to a feature. Due to the presence of negative self-feedback, top-down feedback will be slightly reduced even for a perfect feature. Moreover, if more than one object is present in the scene and they share common subsets of features then, as seen before, the outputs will degrade gracefully. In that case also, the mismatch between the input confidence and the top-down feedback at the feature level slightly increases.

There can also be other cases. Suppose a feature has suffered from noise or some kind of degradation, and its confidence value is less than unity. In that case the top-down feedback may be greater than the input confidence. It may be noted here that the importance of the mismatch is not the same as the importance of mismatch for a perfect feature. The entire ambiguity in the task of recognition depends on how far the features are being interpreted (or explained or emphasized) by the network. If the recognition is perfect, then all the features will be well interpreted. The term "interpretation" refers to the relationship between the input confidence and the top-down support. If the top-down support is less than the input confidence, then the feature is not properly interpreted by the network. If the top-down support is more than the input confidence, then the feature is over-emphasized. Perfect interpretation means a perfect matching. If the average interpretation over the entire feature set can be quantified (i.e., a measure of average ambiguity can be provided) then it will certainly yield a quantitative index for the recognition criterion.

Suppose the network is presented with a set of n features with input confidence values given by $[c_1, c_2, \cdots, c_n]$. Let, after the network has stabilized, the top-down feedback corresponding to these features be $[b_1, b_2, \cdots, b_n]$. Then the total ambiguity D corresponding to the entire feature set can be defined as

$$D = \sum_{i=1}^{n} c_i(c_i - b_i). \tag{17}$$

Note that the mismatch between the input confidence c_i and the top-down feedback b_i in each feature is modulated by the confidence value of the feature itself, thereby setting the relative importance of the mismatch. If $(c_i - b_i)$ increases, the value of D also increases, and vice-versa. Note that $D \geq 0$ and possess a maximum value of $\Sigma_{i=1}^{n} c_i^2 = C$ (say).

The normalized ambiguity measure is given by

$$\mu = \frac{D}{C}. \tag{18}$$

D. Principles of Categorization

In the previous section it has been discussed that the recognition criteria can be evaluated on the basis of the average ambiguity measure μ of a feature (normalized ambiguity). It is clear that if the average ambiguity is high then the recognition is very poor and vice-versa. Therefore the certainty of a decision based on the interpretation of a set of features may be quantified as

$$\mathrm{CF} = 1 - \mu.$$

A high certainty factor (CF) means the system is more confident about its decision, while a small value of CF indicates that the system is less confident or confused in making a decision. (Note that there are similar concepts in pattern recognition problem e.g., see [34] where certainty factor indicates the degree of firmness i.e., meaningfulness or validity of a decision regarding belonging of a feature set to a class. On the other hand, the certainty factor in the present case refers to decision regarding interpretation of a feature set for single/mixed category. In the case of single category, however, both these concepts become analogous.)

Consider the case in the previous example where the network is presented with n features with confidence values $[c_1, c_2, \cdots, c_n]$. Let us assume that all these features belong to a single object. In that case all the features will be active for that particular object. The corresponding output after stabilization will be

$$x = \frac{\sum_{i=1}^{n} w_i c_i}{w_s + \sum_{i=1}^{n} w_i z_i}. \tag{19}$$

Therefore the total ambiguity is given by

$$D = \sum_{i=1}^{n} c_i(c_i - z_i x). \tag{20}$$

Substituting the value of x

$$D = \frac{(w_s + A)\sum_{i=1}^{n} c_i^2}{w_s + \sum_{i=1}^{n} w_i z_i} \tag{21}$$

where

$$A = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(w_j z_j c_i^2 + w_i z_i c_j^2 - (w_i z_j + w_j z_i)c_i c_j)}{2\sum_{i=1}^{n} c_i^2}.$$

The average ambiguity is given by

$$\mu = \frac{w_s + A}{w_s + \sum_{i=1}^{n} w_i z_i}. \tag{22}$$

Let

$$W = \sum_{i=1}^{n} w_i z_i.$$

The certainty factor is, therefore

$$\mathrm{CF} = 1 - \frac{w_s + A}{w_s + W}$$

or

$$\mathrm{CF} = \frac{W - A}{w_s + W}$$

or

$$\mathrm{CF} = \frac{W}{w_s + W}\left(1 - \frac{A}{W}\right). \tag{23}$$

Note that if a perfect pattern from a learned category is presented to the network, then there will be no ambiguity which means that the value of A will be zero. In that case also the certainty factor will be less than unity. This is due to the fact that some negative feedback is present at the output layer for which the output, even for a perfect pattern, will be less than unity under stable condition. Mathematically

$$\mathrm{CF} = \rho_0 cf \tag{24}$$

where

$$\rho_0 = \frac{W}{w_s + W} \tag{25}$$

is the certainty factor for a perfect pattern, and

$$cf = 1 - \frac{A}{W} \tag{26}$$

is the measure of degradation in certainty factor, whenever some noisy or unknown pattern is presented to the network.

Case 1: Suppose an object has k_1 features. Under learned condition all the bottom-up links have weights w and the top-down links have weights unity. Let a new pattern with $k_1 + k_2$ features be presented to the network. The new feature set consists of all the k_1 features of the object. It is considered that the weights of all the bottom-up links from these extra features to the output layer are zero, i.e., these extra features are not learned by the network. In that case the features will receive no top-down feedback. Here, the value of A is given as

$$A = \frac{k_1 k_2 w}{k_1 + k_2}.$$

The value of W is

$$W = k_1 w.$$

Therefore, from (26)

$$cf = 1 - \frac{k_1 k_2 w}{(k_1 + k_2)k_1 w}$$

or

$$cf = \frac{k_1}{k_1 + k_2}.$$

The certainty factor (from (24)) can be written as

$$\mathrm{CF}_1 = \rho_0 \frac{k}{k_a} \tag{27}$$

where k is the number of features of a perfect pattern in the category. In this case this is the same as k_1. k_a is the number of features of the pattern when some extra features are added to it.

Case 2: Suppose the network has learned an object with $k_1 + k_3$ features. That is, the object is fully activated when all the $k_1 + k_3$ features are presented at the input. Now if only k_1 features are presented to the input, then the question is: what will be the certainty of decision.

In this case the value of A is given by

$$A = \frac{k_1 k_3 w}{k_1}$$

i.e.,

$$A = k_3 w.$$

Now

$$W = (k_1 + k_3)w.$$

From (26), the value of cf is given as

$$cf = 1 - \frac{k_3 w}{(k_1 + k_3)w}$$

or

$$cf = \frac{k_1}{k_1 + k_3}.$$

Therefore, from (26), the certainty factor is given as

$$\mathrm{CF}_2 = \rho_0 \frac{k_s}{k} \tag{28}$$

where k has the same meaning as in Case 1. k_s is the total number of features when some features are deleted from the feature set of the perfect pattern.

Case 3: Consider a case where both the situations cited above occur simultaneously. That is, an object consists of $k_1 + k_3$ features out of which only k_1 features have been presented. Moreover, a set of k_2 features is presented to the network which is not at all learned by the network. In that case what will be the certainty of the decision.

The value of A is given as

$$A = \frac{(k_1k_2 + k_2k_3 + k_3k_1)w}{k_1 + k_2}.$$

In this case

$$W = (k_1 + k_3)w.$$

From (26), the value of cf is given as

$$cf = 1 - \frac{(k_1k_2 + k_2k_3 + k_3k_1)w}{(k_1 + k_2)(k_1 + k_3)w}$$

or

$$cf = \frac{k_1^2}{(k_1 + k_2)(k_1 + k_3)}.$$

In this case $k_1 = k_s, k = k_1 + k_3$, and $k_1 + k_2 + k_3 = k_a$. Therefore, the certainty factor is given as [from (24)]

$$\mathrm{CF}_3 = \rho_0 \frac{k_s^2}{k(k_s + k_a - k)}. \tag{29}$$

Some logical reasoning can be derived from the examples cited before. In the first case, some extra features have been presented. Therefore, the entire feature set either collectively corresponds to some other object or it can correspond to more than one object. But the network is only certain about the learned feature set. Therefore, the certainty factor is really the ratio of the learned feature set and the new input feature set [which is substantiated by (27)]. The addition of extra features may also be caused by the presence of some kind of additive noise. For example, in a visual recognition system, some extra (undesirable) features may be detected in the preprocessing stage due to noisy environment. In that case also the network will behave in the same way as cited in Case 1.[2]

In the second case, some features have been taken out from the feature set of a perfect pattern. Therefore, the features which were present in the pattern but not in the new input will be over-emphasized (this is due to the fact that no external input is present for these features but the network will try to support them). Therefore the discarded features will negate the output activation, but the activation will never reduce to zero due to the presence of other features. As a result, there will always be a confusion or ambiguity in the discarded features. Consequently, the certainty of the decision will decrease. Note that this phenomenon must not be confused with the case of sharing of features. If a feature is shared then the feature may not send activation to some of the output nodes, but the presence of the feature would possibly be perfectly interpreted. On the other hand, in Case 2, some features are lost altogether. The decrease in certainty about the decision intuitively happens to be the ratio of the size of the new input feature set to the size of the stored template which is substantiated by (28). It is to be noted that the loss of features may be due to the presence of a pattern from a new category. It can also happen due to the presence of subtractive noise. For example, in the case of visual pattern recognition, some features may not be detected in the preprocessing task due to presence of noise.

In the third case, some features have been discarded, as well as some extra features have been added. This simultaneous loss of features and presence of extra features may be caused by the presence of a pattern from a new category or due to the presence of both additive and subtractive noise.[3] Equation (29) illustrates the effect of such kind of mixed noise.

From the above discussion it is clear that the categorization process (whether a new category code has to be formed or not for a given pattern) should be guided by the fact that how well the features are being interpreted i.e., on the certainty factor of the decision regarding the feature set. If a set of features is presented at the input of a network and it is found that the certainty factor is low enough, then it can be inferred that the network is not able to interpret (explain) the set of features with the current templates stored in its links. The set of features then can be considered to represent a new object class altogether. In this network whenever a feature set is presented, the certainty factor, after stabilization, is compared with some threshold (ρ) called the vigilance threshold. The way of selecting the threshold will be discussed in the next paragraph. Note that the vigilance threshold has similar effect as that of the vigilance factor in the ART [4].

E. Vigilance Threshold and Noise Characteristics

The principle of categorization has been discussed logically. To select the exact vigilance threshold certain characteristics of categorization need to be investigated. The noise tolerance of the categorization will be determined by the vigilance threshold. If the vigilance threshold ρ is very high then the system will be very prone to noise, and due to presence of noise a large number of categories will be formed for the same class of patterns. On the other hand, if ρ is very low then more than one class may fall into the same category, and therefore it will affect the weights of the links also, which eventually may give rise to oscillation among categories for the same class of patterns.

Suppose a pattern is contaminated with the same level of different types of noise (additive, subtractive, or mixed). Same level of noise means that the number of extra features added to a pattern in case of additive noise, the number of features discarded in case of subtractive noise, and the total number of features which are discarded and added in case of mixed noise are same. In that case

$$\mathrm{CF}_1 > \mathrm{CF}_3 > \mathrm{CF}_2. \tag{30}$$

[2]Note that, the structural information necessary for visual recognition is not embedded here. The example is presented only to establish the effect of noise.

[3]Now onwards the presence of both additive and subtractive noise will be referred as the presence of mixed noise.

Proof: Suppose the noise level is $l(<1)$ and the total number of features is k. Then the total number of features added or discarded (or both) will be kl. Therefore the CF values in the first two cases can be written as [from (27) and (28)]

$$\mathrm{CF}_1 = \rho_0 \frac{1}{1+l} \tag{31}$$

and

$$\mathrm{CF}_2 = \rho_0(1-l). \tag{32}$$

In the third case, let the number of features discarded be pkl. Then the number of extra features added is $(1-pl)k$, since the sum of the numbers of features discarded and added is kl. Therefore the noise level can be written as [from (29)]

$$\mathrm{CF}_3 = \rho_0 \frac{(1-pl)^2}{1+(1-2p)l}. \tag{33}$$

To prove $\mathrm{CF}_3 > \mathrm{CF}_2$, we have to prove

$$(1-l)(1+l(1-2p)) < (1-pl)^2.$$

By algebraic manipulation this reduces to

$$(1-p)^2 l^2 > 0$$

which is true for any real value of p.

Similarly, to prove $\mathrm{CF}_3 < \mathrm{CF}_1$ we have to prove

$$(1+l)(1-pl)^2 < 1+(1-2p)l.$$

By algebraic manipulation this reduces to

$$p^2 - 2p + p^2 l < 0$$

or

$$p(1+l) < 2.$$

Since $l<1$, and $p<1$, the expression holds true. Hence proved. □

Suppose, a single pattern with k perfect features each of equal importance has been presented to the network. Then according to (25)

$$\rho_0 = \frac{kw}{w_s + kw} > 1 - \frac{w_s}{kw}.$$

This gives an idea about the maximum value of ρ (or ρ_0) that can be allowed for categorizing a perfect pattern. For example, if we consider $kw = 1$ and $w_s = 0.05$ then the value of ρ_0 is less than 0.95.

In Case 1, the certainty factor of a decision reduces due to the presence of some extra features. These extra features represent some kind of additive noise in a pattern. The value of certainty factor in the case of additive noise is, in fact

$$\mathrm{CF} = \rho_0 cf.$$

Suppose the system tolerates 20% additive noise. In that case, value of cf will be [from (27)] 1/1.2. Therefore, ρ should be selected less than 0.95/1.2 or 0.792.

In Case 2, the certainty factor reduces due to the absence of some features which can be viewed as a sort of subtractive noise. Let the system be able to tolerate 20% subtractive noise. From (28) it is clear that cf is equal to 0.8. Therefore, the value of ρ should be less than 0.95×0.8 or 0.76.

In Case 3, the certainty factor reduces due to presence of both additive and subtractive noise. If the pattern suffers 10% additive and 10% subtractive noise then the value of cf becomes $0.9 \times 0.9/(1.1 + 0.9 - 1)$ or 0.81. Therefore, ρ has to be selected less than 0.95×0.81 or 0.769 which is in between the other two values.

If the system has to tolerate 20% noise in any form then ρ has to be selected nearly 0.76.

Again, the value of ρ determines the capability of the network in distinguishing two different patterns. For low value of ρ, the network may decide two different patterns be from the same category. Therefore the noise tolerance and the distinction capability set the upper and lower bounds for the selection of ρ. If two patterns are sufficiently overlapped then high noise tolerance cannot be achieved which is a common problem in pattern recognition also.

If multiple or mixed (occluded) patterns appear in the scene during categorization then, as seen before, the output will reduce gracefully for some objects. But the decrease in output is not as severe as the decrease in output due to the presence of noise. Therefore if several objects appear together they are checked in the same way; and if the certainty value is less than ρ then the entire feature set is considered to represent a pattern from a new object class.

F. Issues of Stability

X-tron is able to form stable category codes even under the presence of mixed categories. Unlike SONNET [28], it is not designed to form stable codes for embedded patterns. It is able, however, to operate under high amount of overlapping and also under noisy environment.

The learning rules (Section II) reveal the fact that the initial rate of learning for a new category is very high (because ε_l is nearly unity and $g'(u_l)$ is very low). This enables the network to code the incoming pattern at a faster rate. But with several presentations the agility factor of the node decreases and learning rate becomes slow for that particular node. In this model, however, there is no such separate scheme for fast and slow learning; the same learning rule behaves differently for different nodes according to the age of that node.

The categorization technique is based on the assumption that the templates presented to the network are perfectly learned. Therefore whenever a new object class is detected by the network, it is iterated for a number of learning trials until the rate of change of weights becomes less than a small quantity ϵ (say). When the change falls below ϵ, the network is ready to accept a new pattern. Learning a particular category for several trials ensures stability in the process of categorization.

This is also necessary because no order search, as done in ART [4], is performed here. The certainty factor is determined completely on the basis of output response and therefore the network must ensure that the output responses produced are

nearly perfect for learned set of categories. If the weights are not iterated for several trials then patterns from a single class may fall to different categories at different times making the system virtually unstable or indecisive. The system is able to handle both the additive and the subtractive noise.

The stability of categorization process can be explained as follows. If an existing pattern is presented to the network, the corresponding output node becomes highly active. An output node gets activated only when the weights are learned properly. If the weights are learned properly then the output category would be able to interpret the features also. If the features are properly interpreted then the ambiguity in the hidden layer would be low and no new category will be formed.

In the case of mixed categories also, if the pattern corresponds to more than one category and the activation values of the corresponding output nodes are high (i.e., exemplar patterns have been learned) then they will be able to interpret the features. The ambiguity is measured based on the maximum feedback that a set of hidden nodes (for a single input node) received. Therefore the feature will be interpreted by any one of the object categories to which it belongs. As a result, the ambiguity will be low and no new category will be formed.

Let us now consider that the network is presented with a pattern which do not corresponds to any single or any combination of the learned exemplar patterns. In that case no single output node will be highly activated because some of its features would be absent and they would inhibit its activation value. Therefore, the maximum feedback from the output layer to a set of hidden nodes (connected to a single input node) would not be very high. Moreover, there may be some features (in the new pattern) which do not belong to any object. For these features there will be no feedback from output layer (since the connections had not been formed). As a result, there will be a high amount of ambiguity in the interpretation of the features, i.e., certainty factor would be low. In that case the network is able to decide that a new pattern has appeared and consequently it allocates a new output node. This enables the network to have its plasticity property.

The maximum output value is compared with a threshold T as an auxiliary condition. This prevents the network from getting confused by ghost patterns. For example, consider that the network is presented with a learned exemplar pattern with very low feature confidence values (say, a pattern 1 1 0 0 1 is presented in the form 0.3 0.3 0 0 0.3). In that case the output of the corresponding node will be low, but the outputs of other nodes will be even lower. When the output is fed back to the hidden layer, the mismatch between the input confidence value and the feedback will also be low and as a result, the ambiguity will be low. In that case the network may confuse the pattern as a case of the exemplar and flag that it has recognized it, but this is not desirable. This can be prevented (as mentioned in Section IV) whenever the maximum output is compared with the threshold T.

The most appealing part of the system is that it is able to stably categorize patterns even when they appear in mixed form (i.e., mixed set of patterns). This is because of the fact that no single object is flagged as winner in the output layer. Rather, some of the hidden nodes representing the input–output associations are flagged as winners.

TABLE I
PATTERNS ARE PRESENTED AGAINST FEATURES. FULL CONFIDENCE IS REPRESENTED BY 1 AND NO CONFIDENCE AS 0

	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}
pat_1	0	0	0	1	0	0	0	1	0	0	0	1	1	1
pat_2	1	1	0	0	1	1	1	0	1	0	0	0	1	0
pat_3	1	0	0	0	0	1	0	0	1	1	1	0	0	0
pat_4	0	1	1	0	1	0	1	0	0	1	0	0	1	1
pat_5	1	1	0	1	1	0	0	0	0	0	0	0	0	1
pat_6	1	1	0	1	1	0	0	1	1	0	0	0	0	0

TABLE II
WEIGHTS ARE PRESENTED AGAINST FEATURES AND CLASSES. A ZERO WEIGHT INDICATES THAT THE CORRESPONDING LINK IS ABSENT IN THE NETWORK

	C_1	C_2	C_3	C_4	C_5	C_6
f_1	0.0	0.003	0.087	0.0	0.162	0.121
f_2	0.0	0.003	0.0	0.069	0.170	0.127
f_3	0.0	0.0	0.0	0.316	0.0	0.0
f_4	0.117	0.0	0.0	0.0	0.20	0.149
f_5	0.0	0.003	0.0	0.069	0.170	0.127
f_6	0.0	0.007	0.195	0.0	0.0	0.0
f_7	0.0	0.007	0.0	0.164	0.0	0.0
f_8	0.184	0.0	0.0	0.0	0.0	0.235
f_9	0.0	0.004	0.124	0.0	0.0	0.172
f_{10}	0.0	0.0	0.194	0.146	0.0	0.0
f_{11}	0.0	0.0	0.344	0.0	0.0	0.0
f_{12}	0.366	0.0	0.0	0.0	0.0	0.0
f_{13}	0.146	0.005	0.0	0.101	0.0	0.0
f_{14}	0.126	0.0	0.0	0.088	0.216	0.0

V. RESULTS AND ANALYSIS

The connectionist model was simulated on SUN 3/60 workstations using sequential codes. Both binary pattern and visual pattern were considered as input. The purpose of the present investigation is to establish the power of the network in categorizing these inputs. Note that, the application of the network in practical domain like two-dimensional object recognition or medical diagnosis etc., needs domain specific knowledge and that is not incorporated here.

A. Binary Strings

The set of binary patterns which was presented to the network is shown in Table I. The patterns are chosen arbitrarily. At first, the patterns are presented to the network without any noise. The self-feedback was chosen to be 0.05. The value of γ was selected as 0.15. The constant γ was used to realize Weber's law (as done in ART [4]) in the weight updating rules of bottom-up links. The effect of γ is discussed in detail in [31]. The threshold T, above which an output category would be considered to be present, was selected as 0.8.

The categorization (self-organization) property of the network was tested for different vigilance thresholds like 0.8, 0.85, and 0.9. Since the patterns were perfect in all cases they were categorized correctly. As an illustration, the weights of the links attained after 600 presentations are shown in Table II. The weights of the bottom-up links represent the relative importance of the features with respect to output categories. Note that, the weights of the top-down links, created in the network, were unity since the patterns were perfect. Therefore the weights of the top-down links are not presented separately.

TABLE III
CERTAINTY FACTORS FOR DIFFERENT NOISE LEVELS ON THE PATTERNS

case	pattern													certainty factor
1	0	0	0	1	0	0	0	1	0	0	0	1	1	1	0.9494
2	0	0	0	0.9	0	0	0	0.9	0	0	0	0.9	0.9	0.9	0.9494
3	0.1	0.1	0.1	1	0.1	0.1	0.1	1	0.1	0.1	0.1	1	1	1	0.9492
4	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.9	0.9	0.9	0.9491
5	1	0	0	1	0	0	0	1	0	0	0	1	1	1	0.8401
6	0	0	0	1	0	0	0	1	0	0	0	1	1	0	0.8220
7	1	0	0	1	0	0	0	1	0	0	0	1	1	0	0.7163
8	1	0	0	0	0	1	0	0	1	1	1	0	0	0	0.9497
9	0.9	0	0	0	0	0.9	0	0	0.9	0.9	0.9	0	0	0	0.9497
10	1	0.1	0.1	0.1	0.1	1	0.1	0.1	1	1	1	0.1	0.1	0.1	0.9493
11	0.9	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.9	0.9	0.9	0.1	0.1	0.1	0.9492
12	1	0	0	0	0	1	0	0	1	1	1	0	0	1	0.8361
13	1	0	0	0	0	1	0	0	0	1	1	0	0	0	0.8249
14	1	0	0	0	0	1	0	0	0	1	1	0	0	1	0.7136

1) Effect of Noise on CF: In the next phase of experiment the effect of noise on a particular pattern was studied. For this purpose input patterns were contaminated with various levels of additive and subtractive noise. The effect of noise on the certainty factor for the concerned pattern is presented in Table III. From this table it is clear that when a perfect pattern (Case 1 in Table III) from the first category of Table I was presented then CF was 0.9494 which is the value of ρ_0 (25). The confidence values of the features were replaced by 0.9, and it was found that CF retains the same value. This is due to the fact that in finding CF, the total ambiguity is always normalized by the total input confidence (26). Therefore, apparently it seems that even if the confidence values of the features, which were present in the perfect pattern, go on decreasing, the value of CF will retain its original value (ρ_0). This is the basic reason for which the value of output threshold was always considered in the process of self-organization. Although CF retains its original value, the maximum output goes on decreasing with the decrease in the confidence values of the features. Since the output decreases proportionately with the confidence values of the features, the average ambiguity remains constant and thereby CF retains its value.

The pattern was again contaminated with some additive noise where all the features which were actually absent have now confidence value equal to 0.1 (Case 3). In that case it (Table III) shows that CF slightly decreases. When both the noise are present simultaneously then also CF decreases slightly. In fact, in all three cases the maximum output in the output layer will determine if the noisy pattern is from the present category or from a new one, during the process self-organization.

We have then checked the performance of the network by adding one extra feature to the pattern, namely the first feature. Results (Case 5) show that CF decreases by a great extent, from 0.94 to 0.84. This is due to the fact that there was no top-down support to that feature from the category concerned. Therefore the ambiguity at that feature is very high and hence the average ambiguity also increases and, as a result, CF decreases. Note that, in Case 4 (Table III), the effect of noise was not very high to any particular feature, and therefore there was a graceful degradation in the performance, which is not true in the present one (Case 5). Case 6 shows the performance when a feature is deleted from the pattern, namely, the last feature. Result indicates a drastic reduction in CF from 0.94 to 0.82. This has also the same explanation as in Case 5. Case 7 presents the result when an extra feature is added and a genuine feature is discarded. In effect, CF reduces from 0.94 to 0.72.

TABLE IV
RESULTS OF CATEGORIZATION WHEN PATTERNS ARE PRESENTED INDIVIDUALLY WITH VIGILANCE FACTOR = 0.9 AND NOISE LEVEL = 0.3

Actual class	category																
1	1	0	0	0	0	0	42	0	0	0	0	0	0	0	0	1	0
2	0	1	0	0	0	0	0	6	0	1	1	26	0	0	8	0	0	1
3	0	0	1	0	0	0	0	0	41	0	0	0	0	0	0	0	1	0
4	0	1	0	39	0	0	0	0	0	0	0	0	1	0	0	1	0	2
5	0	0	2	0	32	0	0	0	0	0	0	0	0	10	0	0	0	0
6	0	0	0	0	0	44	0	0	0	0	0	0	0	0	0	0	0	0

Note that the objective of the investigations performed through Cases 1–7 is to establish the nature of CF and not the nature of maximum output. Because in the categorization process the output threshold is always kept fixed (chosen as 0.8 with negative self-feedback of 0.05), and the categorization property was studied with different vigilance thresholds. Cases 8–14 show the similar effects on another pattern, namely the pattern from third category in Table I.

2) Categorization of Individual Patterns: In this experiment the values of self-feedback and γ are chosen as before. It is found that when the patterns are contaminated with 10% and 20% random noise, the network is able to categorize them perfectly with vigilance thresholds 0.8 and 0.9. Here, 10% noise means the maximum level of noise is 10% of the maximum confidence value that can occur. For example, in the present experiment, the maximum confidence value is unity, and hence maximum noise level (additive or subtractive) can be 0.1. When the patterns were contaminated with 30% noise the network is able to categorize correctly with a vigilance threshold (ρ) equal to 0.8. On the other hand, if ρ is selected as 0.9 then it is seen that a large number of categories are created. Since perfect categorization occurs in the first two cases the results are not presented here. Table IV shows the results of categorization with 30% noise and ρ equal to 0.9. It may be noted that T is always kept fixed at 0.8. From the table it is clear that although quite a few redundant categories have been formed, there are only six distinct categories into which the patterns fall most of the times. This shows that although the network is initially confused with such a high noise level and high value of ρ, it has been able to associate a particular node with a particular category finally.

The network behavior is then studied with very low CF (ρ). The patterns were again contaminated with a high noise level of 30% and ρ was selected to be 0.65. Table V shows the categorization performance of the network. The results show that the network has been able to categorize almost perfectly. Only thing is that initially two classes (1 and 6) fell into the same category, but afterwards, class 1 was allocated to some new category.

3) Categorization of Mixed Patterns: Here the capability of the network to categorize both the individual and mixed patterns simultaneously is studied. The results corresponding

TABLE V
RESULTS OF CATEGORIZATION WHEN PATTERNS ARE PRESENTED INDIVIDUALLY WITH VIGILANCE THRESHOLD = 0.65 AND NOISE LEVEL = 0.3

Actual class	*category*						
1	1	0	0	0	0	49	0
2	0	50	0	0	0	0	0
3	0	0	50	0	0	0	0
4	0	0	0	50	0	0	0
5	0	0	0	0	1	0	49
6	50	0	0	0	0	0	0

TABLE VI
RESULTS OF CATEGORIZATION WHEN INDIVIDUAL AND MIXED PATTERNS APPEAR RANDOMLY TO THE NETWORK. "cls" STANDS FOR CLASS AND "ctg" STANDS FOR CATEGORY

	ctg1	ctg2	ctg3	ctg4	ctg5	ctg6	ctg7
cls1	0	0	0	0	58	0	0
cls2	0	0	0	0	0	63	0
cls3	0	0	60	0	0	0	0
cls4	74	0	0	0	0	0	0
cls5	0	0	0	56	0	0	0
cls6	0	1	0	0	0	0	58
cls1&2	0	0	0	1	6	5	3
cls1&3	0	0	6	0	6	0	0
cls1&4	6	0	0	0	6	0	0
cls1&5	0	0	0	3	3	0	0
cls1&6	0	0	0	1	4	0	3
cls2&3	0	0	4	0	0	4	0
cls2&4	7	0	0	0	0	7	0
cls2&5	0	0	0	6	0	6	0
cls2&6	0	0	0	0	0	7	7
cls3&4	5	0	5	0	0	3	0
cls3&5	0	0	8	8	0	0	0
cls3&6	0	0	3	0	0	0	3
cls4&5	6	0	0	6	0	0	0
cls4&6	2	0	0	0	0	0	2
cls5&6	0	0	0	5	0	0	7

to 10% noise level and 0.8 vigilance threshold are shown in Table VI when the patterns were presented in random order. Note that if a mixed pattern appears it was treated either as a new category or as a superposition of two different categories. In Table VI the output categories are presented against the actual classes.

It is clear from Table VI that whenever the pattern from an individual class appears it has been categorized perfectly. For example, all 74 patterns from class 4 have been identified as the first category. Only one pattern from class 6 fell into second category which is really a redundant one. Whenever mixed patterns appear, they are identified either as a new one or as a mixture of the learned categories. For example, whenever a mixture of patterns from classes 1 and 5 appears, it is identified by the network as a mixture of categories 4 and 5. This is, of course, a correct decision because all patterns from class 1 fell into fifth category and those from class 5 went to fourth category. In the table the most confusing result corresponds to the entries for the mixture of classes 1 and 2. Note that patterns from class 1 and class 2 went to categories 5 and 6, respectively. This reveals the fact that these categories were learned after the first four (one of them is redundant) which correspond to classes 3, 4, and 5. Initially, when a mixture from classes 1 and 2 appears, the network was not able identify them as a mixed category and therefore a new category was allocated (category 4). Later, this category is taken over by the patterns from class 5. Whenever the network becomes able to categorize the patterns from classes 1 and 2, the mixture of 1 and 2 is no more confusing. The network is able to identify the mixture five times (from Table VI). Results show that the network is able to identify the mixtures of other patterns also.

TABLE VII
CONFUSION MATRIX WHEN VISUAL PATTERNS ARE PRESENTED WITH VIGILANCE THRESHOLD = 0.8 AND NOISE LEVEL = 0.2. "cls" STANDS FOR CLASS AND "ctg" STANDS FOR CATEGORY

	ctg1	ctg2	ctg3	ctg4
cls1	0	12	0	0
cls2	23	0	0	0
cls3	0	0	19	0
cls4	0	0	0	18
cls1& 2	7	7	0	0
cls1& 3	0	3	3	0
cls1& 4	0	5	0	5
cls2& 3	4	0	4	0
cls2& 4	2	0	0	2
cls3& 4	0	0	7	7

B. Visual Patterns

Here, the parameters of the network, namely, amount of self-feedback, γ are selected as before. The visual patterns are contaminated by both additive and subtractive noise and then categorized by the network with a vigilance threshold equal to 0.8 and output threshold 0.8. It is found that the network is capable of correctly categorizing the patterns (both individual and mixed) even in the presence of 30% noise. As an illustration, the results for 20% contamination are presented in Table VII.

VI. DISCUSSION AND CONCLUSIONS

In this article we have presented a new self-organization technique which is capable of extracting out individual categories even when they appear in mixed form. An ambiguity measure, depending on the interpretation of the input features, is defined, based on which the categorization process has been formulated. The corresponding network architecture (X-tron) is also proposed here. The network automatically adjusts the number of nodes in the hidden and output layers, depending on the complexity or nature of overlap between the patterns. Note that the entire structure of X-tron basically follows from the mathematical formulation (2) which was derived for properly interpreting a feature set.

The effectiveness of the method is demonstrated for both binary and visual patterns in presence of additive, subtractive, and mixed noise. Note that the binary strings considered in the present investigation have a higher degree of overlap between themselves compared to the visual patterns. As a consequence, the categorization of the binary strings (individual or mixed patterns) seemed to be more difficult than the visual patterns. That is why the characteristics of the network has been studied in detail on the binary strings to establish the exactitude of the self-organizing capability. The categorization results for visual inputs also have been provided.

The characteristics of the proposed system well compares with the self-organization property as shown in adaptive resonance theory. First, in ART an order search is necessary among the output categories to find if one of them interprets (explains) the input pattern. In the proposed system no such

search is necessary. This is due to the fact that the ambiguity is in the featural level and the activations of all output nodes are kept in tact; none of them are disabled. Second, ART does not deal with multiple or mixed categories. On the other hand, if a mixture of known categories appears at the input, the present network is able to detect it and no new category is formed. Note that the system provides output in continuum grades (fuzzy). The concept of continuum grades was also considered in ART2, ART3, and fuzzy ARTMAP for forming stable codes for individual categories.

X-tron is able to self-organize in the presence of mixed categories. The model proposed by Cho and Reggia [20] can also do the mixed category recognition, but the learning process is supervised. Moreover, the way X-tron operates is entirely different from that proposed in [20]. The system developed by Marshall [22]–[24] does the task of mixed category perception by limiting the competition process within similar types of nodes, whereas X-tron performs this task (with high overlap between the patterns) by dissociating the competition process from the output layer. SONNET [27], [28], [30] is able to form stable categories for embedded patterns. X-tron forms stable category only when it finds the pattern separately (and the problem of embedded patterns has not been addressed). Note that SONNET uses a concept of competition between the links for associating a feature with an object, whereas the process of competition takes place in the hidden layer of X-tron. Amalgamation of SONNET with X-tron may possibly lead to a more powerful one. Cohen and Grossberg [25], [26] have considered the mixed category recognition problem using masking fields, but their network does not form any stable category.

X-tron also compares with other self-organization models (which do not perform the task of mixed category recognition) in a similar line. For example, in Neocognitron [13], the case where the feature set of a category is a proper subset of the feature set of another category, was solved by employing inhibitory connections. On the other hand, the same phenomenon has been solved here by using Weber's law as in the case of ART. If inhibitory connections were used then the activation of the genuine categories would have reduced if a pattern representing a set of mixed categories is presented to the network. This is due to the fact that the inhibitory connection will ensure that if a redundant feature is presented at the input, it will reduce the output of a category. In the case of a mixed pattern, the features of one pattern would play the role of redundant features with respect to other patterns (provided the feature is not shared). Therefore, the activation of the true categories would decrease to a great extent if the mechanism of inhibitory connections were embedded.

Although the network has been tested on synthetic data (binary and visual patterns), it can be used to solve real life object recognition problem under occlusion, partial information loss or noisy environment. It may be mentioned here that in practical visual recognition problem, the degree of high overlap, as considered for binary strings, may be rarely possible. For example, if we use "corner" as features for object recognition then it is unlikely that a particular corner feature is shared by more than one object in the scene. The results on binary strings indicate the fact that if the structural information from the feature space to the object space can be embedded into the connections of the network, then it can possibly be applied to object recognition tasks.

ACKNOWLEDGMENT

The authors acknowledge Prof. D. D. Majumder for his keen interest in this work and for providing the computing facility of the KBCS center. The authors are also thankful to Dr. C. A. Murthy and Dr. S. Chaudhury for helpful discussions, and S. Chakraborty for preparing diagrams.

REFERENCES

[1] R. Linsker, "From basic network principles to neural architecture: Emergence of spatial opponent cells," in *Proc. Nat. Academy Sci. USA*, vol. 83, 1986, pp. 7508–7512.
[2] ——, "From basic network principles to neural architecture: Emergence of orientation-selective cells," in *Proc. Nat. Academy Sci. USA*, vol. 83, 1986, pp. 8390–8394.
[3] T. Kohonen, *Self-Organization and Associative Memory*. Berlin: Springer-Verlag, 1988.
[4] G. A. Carpenter and S. Grossberg, "A massively parallel architecture for a self-organizing neural pattern recognition machine," *Comput. Vision, Graphics Image Process.*, vol. 37, pp. 34–115, 1987.
[5] ——, "Self-organization of stable category recognition codes for analog input patterns," *Appl. Opt.*, vol. 26, pp. 4919–4930, 1987.
[6] ——, "Art3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures," *Neural Networks*, vol. 3, pp. 129–152, 1990.
[7] G. Carpenter, S. Grossberg, and D. Rosen, "Fuzzy art: Fast stable learning and categorization of analog patterns by an adaptive resonance system," *Neural Networks*, vol. 4, pp. 493–504, 1991.
[8] G. Carpenter, S. Grossberg, and J. Reynolds, "Artmap: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network," *Neural Networks*, vol. 4, pp. 565–588, 1991.
[9] G. Carpenter, S. Grossberg, N. Markuzon, J. Reynolds, and D. Rosen, "Fuzzy artmap: A neural network architecture for incremental supervised learning of analog multidimensional maps," *IEEE Trans. Neural Networks*, vol. 3, pp. 698–713, 1992.
[10] S. Amari, "Neural theory of association and concept formation," *Biological Cybern.*, vol. 26, pp. 175–185, 1977.
[11] S. Amari and K. Maginu, "Statistical neurodynamics of associative memory," *Neural Networks*, vol. 1, pp. 63–74, 1988.
[12] B. Kosko, *Neural Networks and Fuzzy Systems: A Dynamical Approach to Machine Intelligence*. Englewood Cliffs, NJ: Prentice-Hall, 1991.
[13] K. Fukushima, "Neural network model for selective attention in visual pattern recognition and asociative recall," *Appl. Opt.*, vol. 26, pp. 4985–4992, 1987.
[14] H. Ritter and T. Kohonen, "Self-organizing semantic maps," *Biological Cybern.*, vol. 61, pp. 241–254, 1989.
[15] J. Minnix, E. McVey, and R. Iñigo, "A multilayered self-organizing artificial neural network for invariant pattern recognition," *IEEE Trans. Knowledge Data Eng.*, vol. 4, pp. 162–167, 1992.
[16] J. Marshall, "Self-organizing neural networks for perception of visual motion," *Neural Networks*, vol. 3, pp. 45–74, 1990.
[17] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1978.
[18] C. Stanfill and D. Waltz, "Toward memory-based reasoning," *Commun. ACM*, vol. 29, pp. 1213–1228, 1986.
[19] Y. Peng and J. Reggia, "A connectionist model for diagnostic problem solving," *IEEE Syst., Man, Cybern.*, vol. 19, pp. 285–298, 1989.
[20] S. Cho and J. Reggia, "Learning competition and cooperation," *Neural Computa.*, vol. 5, pp. 242–259, 1993.
[21] J. Reggia, C. DAutrechy, G. Sutton, III, and M. Weinrich, "A competitive distribution theory of neocortical dynamics," *Neural Computation*, vol. 4, pp. 287–317, 1992.
[22] J. Marshall, "A self-organizing scale-sensitive network," in *Proc. Int. Joint Conf. Neural Networks*, San Diego, CA, 1990, pp. 649–654.
[23] ——, "Representation of uncertainty in self-organizing neural networks," in *Proc. Int. Neural Networks Conf.*, Paris, France, 1990, pp. 809–812.

[24] ——, "Development of perceptual context-sensitivity in unsupervised neural networks: Parsing, grouping and segmentation," in *Proc. Int. Joint Conf. Neural Networks*, Baltimore, MD, 1992, pp. 315–320.
[25] M. Cohen and S. Grossberg, "Neural dynamics of speech and language coding: Developmental programs, perceptual grouping, and competition for short-term memory," *Human Neurobiology*, vol. 5, pp. 1–22, 1986.
[26] ——, "Masking fields: A massively parallel neural architecture for learning, recognizing, and predicting multiple groupings of data," *Appl. Opt.*, vol. 26, pp. 1866–1891, 1987.
[27] A. Nigrin, "The stable classification of temporal sequences with an adaptive resonance circuit," in *Proc. Int. Joint Conf. Neural Networks*, Washington, DC, 1990, pp. 525–528.
[28] ——, "Sonnet: A self-organizing neural network that classifies multiple patterns simultaneously," in *Int. Joint Conf. Neural Networks*, San Diego, CA, 1990, pp. 313–318.
[29] ——, "The stable learning of temporal patterns with an adaptive resonance circuit", Ph.D. dissertation, Duke Univ, 1990.
[30] ——, "A new architecture for achieving translational invariant recognition of objects," in *Proc. Int. Joint Conf. Neural Networks*, Baltimore, MD, 1992, pp. 683–688.
[31] J. Basak, C. A. Murthy, S. Chaudhury, and D. D. Majumder, "A connetcionist model category perceptron: Theory and implementation," *IEEE Trans. Neural Networks*, vol. 4, pp. 257–269, 1993.
[32] S. Grossberg, *Studies of Mind and Brain*. Boston: Reidel, 1982.
[33] J. Platt, "A resource-allocating network for function interpolation," *Neural Computa.*, vol. 3, pp. 213–225, 1991.
[34] D. P. Mandal, C. A. Murthy, and S. K. Pal, "Formulation of a multivalued recognition system," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, pp. 607–620, 1992.

Jayanta Basak received the bachelor's degree in electronics and telecommunication engineering from Jadavpur University, Calcutta, India, in 1987. He received the M.E. degree in computer science and engineering from the Indian Institute of Science, Bangalore, in 1989.

He served as Computer Engineer in the National Center for Knowledge-Based Computing, Calcutta, from 1989–1992. In April 1992 he joned ECSU, ISI as a Programmer. Presently, he is working in the Machine Intelligence Unit of the same Institute. He visited the Robotics Institute of Carnegie Mellon University, Pittsburgh, PA, in 1991 and 1992 under a UDNP Fellowship. His current research interests include neural networks and computer vision.

Mr. Basak received the Young Scientist Award from the Indian Science Congress Association in 1994.

Sankar K. Pal, for a photograph and biography, see p. 63 of the January issue of this TRANSACTIONS.