NEUCOM 1966
pp: 1--13 (col.fig.: Nil)
PROD. TYPE: COM
ED: Sushma
PAGN: Vidya – SCAN: Raghu

ARTICLE IN PRESS

Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Neurocomputing ▮▮▮ (▮▮▮▮) ▮▮▮–▮▮▮
NEUROCOMPUTING
www.elsevier.com/locate/neucom

A color pattern recognition problem based on the multiple classes random neural network model

Jose Aguilar*

CEMISID. Departmento de Computación, Facultad de Ingeniería, Universidad de los Andes, Av. Tulio Febres. Mérida Venezuela, Venezuela

Abstract

Gelenbe has modeled the neural network using an analogy with the queuing theory. Recently, Fourneau and Gelenbe have proposed an extension of this model, called multiple classes random neural network (RNN) model. The purpose of this paper is to describe the use of the multiple classes RNN model to recognize patterns having different colors. We propose a learning algorithm for the recognition of color patterns based upon the non-linear equations of the multiple classes RNN model using gradient descent of a quadratic error function. In addition, we propose a progressive retrieval process with adaptive threshold value.

Keywords: Multiple classes random neural network; Color pattern recognition; Learning algorithm; Image processing; Retrieval process

1. Introduction

Humans use color, shape and texture to understand and recollect the contents of a pattern. Therefore, it is natural to use features based on these attributes for pattern recognition [8,9,16,17]. In [15] is demonstrated the effectiveness of using simple color features for pattern recognition. Colombo et al. [8] described a system for pictorial content representation and recognition based on color distribution features. They described the distribution of chromatic content in a pattern through a set of color histograms and a pattern-matching strategy using these sets. Recently, Mojsilovic et al. [17] determined the basic categories (vocabulary) used by humans in judging similarity of color

* Tel.: +58-74-402914; fax: +58-74-402872.
E-mail address: aguilar@ula.ve (J. Aguilar).

0925-2312/$ - see front matter © 2004 Published by Elsevier B.V.
doi:10.1016/j.neucom.2004.03.005

patterns, their relative importance and relationships, as well as the hierarchy of rules (grammar).

Coming up with effective learning algorithms for recurrent networks is a current and legitimate scientific subject in neural network theory [10]. There are numerous examples where recurrent networks constitute a natural approach to problems: image processing, pattern analysis and recognition, etc., where local interactions between picture elements lead to mutual interactions between neighboring neurons which are naturally represented by recurrent networks. In such cases, it is clear that effective-learning algorithms enhance the value of neural network methodology.

The problem addressed in this paper concerns the proposition of a color pattern recognition approach composed by a learning algorithm and a retrieval procedure for the multiple classes random neural network (RNN). We use each class to model a color. We present a backpropagation type learning algorithm for the recurrent multiple classes RNN model using gradient descent of a quadratic error function when a set of input–output pairs is presented to the network. Our model is defined for nC parameters for the whole network, where C is the number of primary colors, n is the number of pixels of the image, and each neuron is used to obtain the color value of each pixel in the bit map plane. The primary colors create different colors according to the RGB model. Thus, our learning algorithm requires the solution of a system of nC non-linear equations each time the n-neurons network learns a new input–output pair (n-pixels image with C primary colors). In addition, we propose a progressive retrieval process with adaptive threshold value.

The RNN has been proposed by Gelenbe in 1989 [11–13]. This model does not use a dynamic equation, but uses a scheme of interaction among neurons. It calculates the probability of activation of neurons in the network. Signals in this model take the form of impulses that mimic what is presently known as inter-neural signals in biophysical neural networks. The RNN has been used to solving optimization [1,2,4] and pattern recognition problems [3,6,7]. Gelenbe has considered learning algorithm for the recurrent RNN model [14]. We have proposed modifications of this algorithm for combinatorial optimization problems [4] and an evolutionary learning for combinatorial optimization and recognition problems [1,6]. Fourneau et al. have proposed an extension of the RNN, called multiple classes RNN model [10].

This work is organized as follows: in Section 2 we present the multiple classes RNN, Section 3 presents our recognition algorithm (learning and retrieval processes) for multiple classes RNN, and Section 4, we presents applications. Remarks concerning future work and conclusions are provided in Section 5.

2. The random neural model

2.1. General properties of the random neural network model

The RNN model has been introduced by Gelenbe [11–13] in 1989. This model has a remarkable property called "product form" which allows the computation of joint-probability distributions of neurons of the network. The model consists of a network of n neurons in which positive and negative signals circulate. Each neuron accumulates signals as they arrive, and can fire if its total signal count at a given instant of time is positive. Firing then occurs at random according to an exponential distribution of constant rate, and signals are sent out to other neurons or to the outside of the network. Each neuron i of the network is represented at any time t by its input signal potential $k_i(t)$.

Positive and negative signals have different roles in the network. A negative signal reduces by 1 the potential of the neuron to which it arrives (inhibition) or has no effect on the signal potential if it is already zero; while an arriving positive signal adds 1 to neuron potential. Signals can either arrive to a neuron from outside of the network or from other neurons. Each time a neuron fires, a signal leaves it, depleting the total input-potential of the neuron. A signal which leaves neuron i heads for neuron j with probability $p^+(i,j)$ as a positive signal (excitation), or as negative signal with probability $p^-(i,j)$ (inhibition), or it departs from the network with probability $d(i)$. Clearly, we shall have:

$$\sum_{j=1}^{n} [p^+(i,j) + p^-(i,j)] + d(i) = 1 \quad \text{for } 1 \leqslant i \leqslant n.$$

Positive signals arrive to the ith neuron according to a Poisson process of rate $\Lambda(i)$ (external excitation signals). Negative signals arrive to the ith neuron according to a Poisson process of rate $\lambda(i)$ (external inhibition signals). The rate at which neuron i fires is $r(i)$. The main property of this model is the excitation probability of a neuron i, $q(i)$, which satisfies a non-linear equation:

$$q(i) = \frac{\lambda^+(i)}{r(i) + \lambda^-(i)}, \tag{1}$$

where

$$\lambda^+(i) = \sum_{j=1}^{n} q(j)r(j)p^+(j,i) + \Lambda(i),$$

$$\lambda^-(i) = \sum_{j=1}^{n} q(j)r(j)p^-(j,i) + \lambda(i).$$

The synaptic weights for positive ($w^+(i,j)$) and negative ($w^-(i,j)$) signals are defined as

$$w^+(i,j) = r(i)p^+(i,j), \quad w^-(i,j) = r(i)p^-(i,j)$$

and

$$r(i) = \sum_{j=1}^{n} [w^+(i,j) + w^-(i,j)].$$

If a unique non-negative solution exists in Eq. (1) such that each $q(i) \leqslant 1$, then the stationary probability distribution is

$$p(k) = \prod_{i=1}^{n} (1 - q(i))q(i)^{k(i)}, \quad k(t) \text{ is the vector of signal potentials at time } t.$$

To guarantee stability of the RNN, the following equation is a sufficient condition for the existence and uniqueness of the solution in Eq. (1)

$$\lambda^+(i) < [r(i) + \lambda^-(i)].$$

Note that the model is based on rates at which natural neural systems operate. Thus, this is a "frequency modulated" model, which translates rates of signal emission into excitation probabilities via Eq. (1). For instance $q(j)r(j)p^+(j,i)$ denotes the rate at which neuron j excites neuron i. Eq. (1) can also be translated into a special form of sigmoid that treats excitation (in the numerator) asymmetrically with respect to inhibition (in the denominator).

2.2. The multiple classes random network model

We now describe the multiple classes random network model introduced in [10]. The neural network is composed of n neurons and receives exogenous positive (excitatory) and negative (inhibitory) signals as well as endogenous signals exchanged by the neurons. As in the classical model 1989 [11–13], excitatory and inhibitory signals are sent by neurons when they fire, to the other neurons in the network or to the outside world. In this model, positive signals may belong to several classes and the potential at a neuron is represented by the vector $K_i = (K_{i1}, \ldots, K_{iC})$, where K_{ic} is the value of the "class c potential" of neuron i, or its "excitation level in terms of class c signals", and negative signals only belong to a single class. The total potential of neuron i is $K_i = \sum_{c=1}^{C} K_{ic}$. The arrival of an excitatory signal of some class increases the corresponding potential of a neuron by 1, while an inhibitory signal's arrival decreases it by 1. That is, when a positive signal of class c arrives at a neuron, it merely increases K_{ic} by 1, and when a negative signal arrives at it, if $K_i > 0$, the potential is reduced by 1, and the class of the potential to be reduced is chosen randomly with probability K_{ic}/K_i for any $c = 1, \ldots, C$. A negative signal arriving at a neuron whose potential is zero has no effect on its potential.

Exogenous positive signals of class c arrive at neuron i in a Poisson stream of rate $\Lambda(i,c)$, while exogenous negative signals arrive at it according to a Poisson process of rate $\lambda(i)$. A neuron is excited if its potential is positive. It then fires, at exponentially distributed intervals, sends excitatory signals of different classes, or inhibitory signals, to other neurons or to the outside of the network. That is, the neuron i can fire when its potential is positive ($K_i > 0$). The neuron i sends excitatory signals of class c at rate $r(i,c) > 0$, with probability K_{ic}/K_i. When the neuron fires at rate $r(i,c)$, it deletes by 1 its class c potential and sends to neuron j a class φ positive signal with probability

$$r(i,c)[K_{ic}/K_i]\,p^+(i,c;j,\varphi),$$

or a negative signal with probability

$$r(i,c)[K_{ic}/K_i]\,p^-(i,c;j)\,p^-(i,c;j).$$

On the other hand, the probability that the deleted signal is sent out of the network, or that it is "lost", is

$$r(i,c)[K_{ic}/K_i]d(i,c).$$

Clearly, we shall have

$$\sum_{(j,\varphi)} p^{+}(i,c;j,\varphi) + \sum_{j} p^{-}(i,c;j) + d(i,c) = 1 \quad \forall i = 1,n \text{ and } c = 1,C.$$

Let $K(t)$ be the vector representing the state of the neural network at time t and $K = (K_1, \ldots, K_n)$ be a particular value of the vector. We shall denote by $p(K,t) = Pr[K(t) = K]$ the probability distribution of its state. The main property of this model is the excitation probability of the "class φ potential" of neuron j, $q(j,\varphi)$, with $0 < q(j,\varphi) < 1$, which satisfies the non-linear equation [10]:

$$q(j,\varphi) = \frac{\lambda^{+}(j,\varphi)}{r(j,\varphi) + \lambda^{-}(j)}, \tag{2}$$

where

$$\lambda^{+}(j,\varphi) = \sum_{(i,c)} q(i,c) r(i,c) p^{+}(i,c;j,\varphi) + \Lambda(j,\varphi),$$

$$\lambda^{-}(j) = \sum_{(i,c)} q(i,c) r(i,c) p^{-}(i,c;j) + \lambda(j).$$

Thus, $p(K,t)$, the stationary probability distribution of network state, satisfies

$$p(K,t) = \prod_{i=1}^{n} \prod_{c=1}^{C} (1 - q(i,c)) q(i,c)^{k_{ic}}.$$

The synaptic weights for positive $(w^{+}(i,c;j,\varphi))$ and negative $(w^{-}(i,c;j))$ signals are defined as

$$w^{+}(i,c;j,\varphi) = r(i,c) p^{+}(i,c;j,\varphi), \quad w^{-}(i,c;j) = r(i,c) p^{-}(i,c;j),$$

and, if $d(i,c) = 0$, the fire rate $r(i,c)$ will be

$$r(i,c) = \left[\sum_{(j,\varphi)} w^{+}(i,c;j,\varphi) + \sum_{(j)} w^{-}(i,c;j) \right].$$

3. Color pattern recognition algorithm on the multiple classes random neural network model

We now show how the multiple classes RNN can be used to solve the color pattern recognition problem. The recognition procedure is based on an associative memory technique [3,5]. In our approach, a "signal class" represents each color distinctly. To design such a memory, we have used a single-layer RNN of n fully interconnected neurons. For every neuron i the probability of emitting signals from the network is $d(i,c) = 0$. We suppose a pattern composes by n points (m,k) in the plane (for $m = 1, \ldots, J$ and $k = 1, \ldots, K$). We associate a neuron $N(i)$ to each point (m,k) in the plane (for $i = 1, \ldots, n$; $m = 1, \ldots, J$ and $k = 1, \ldots, K$). The state of $N(i)$ can be interpreted as the color intensity value of the pixel (m, k). That is, each pixel is represented by a neuron (p.e, pixel (1,1) is neuron $N(1)$, and pixel (J, K) is neuron $N(n)$).

In the other hand, we suppose that three classes represent the primary colors (red, green, and blue), according to the RGB model. The RGB model creates different colors with the combination of different intensities of primary colors. For example, to represent a pixel with red color the neuron value is $(1, 0, 0)$, the black color is $(1, 1, 1)$, the pink color is $(0.5, 0, 0)$, etc. We suppose the values to be equal to 0, 0.5 and 1 for each class on every neuron. In this way, we can represent geometric figures with different combinations of colors. We have used this model because it agrees better with human chromatic perception [8], but our approach can use another model like this one to represent the colors of a given pattern. In order to select a color, we need to take into account that each primary color will be a class, which increases the complexity of the system. The parameters of the neural network will be chosen as follows:

(a) $p^+(j,\varphi;i,c) = p^+(i,c;j,\varphi)$ $p^-(i,c;j) = p^-(j,c;i)$ for any $i,j = 1,\ldots,n$ and $c,\varphi = 1,\ldots,C$.

(b) $\Lambda(i,c) = L_{ic}$ and $\lambda(i) = 0$, where L_{ic} is a constant for the class c of the neuron i. The choice of the value L_{ic} for each color can made as follows. Since the network parameters are homogeneous, the equations will lead to a single point $q(i,c)$ for any i for a color. If we call this value $q(c)$, then we can write:

$$q(j,\varphi) = \frac{\alpha(c)q(c) + L_{ic}}{r(c) + bq(c)}$$

yielding, $L_{ic} = q(c)(bq(c) + r(c) - \alpha(c))$

and $\alpha(c) = \sum_{(j,\varphi,t)} w^+(j,\varphi;i,c)$

$q(c)$ can be interpreted as the average intensity of the color c for the image which will be recognized, given a real number between 0 and 1; thus, L_{ic} must be chosen so as to bring $q(c)$ to the desired value.

3.1. Learning algorithm

Now, we search to define a learning algorithm for the multiple classes RNN model. We propose a gradient descent algorithm for choosing the set of network parameters $w^+(j,z;i,c)$ and $w^-(j,z;i)$ in order to learn a given set of m input–output pairs (X, Y) where the set of successive inputs is denoted by

$X = \{X_1,\ldots,X_m\}$ where $X_k = \{X_k(1,1),\ldots,X_k(n,C)\}$, and $X_k(i,c)$ is the cth class on the neuron i for the patron k

$$X_k(i,c) = \{\Lambda_k(i,c),\ \lambda_k(i)\}$$

and the successive desired outputs are the vector

$$Y = \{Y_1,\ldots,Y_m\},\quad \text{where } Y_k = \{Y_k(1,1),\ldots,Y_k(n,C)\},\ \text{and } Y_k(1,1) = \{0, 0.5, 1\}.$$

The values $\Lambda_k(i,c)$ and $\lambda_k(i)$ provide network stability. Particularly, in our models $\Lambda_k(i,c)$ and $\lambda_k(i)$ are initialized as defined previously. Normally, arrival rates of exogenous signals are chosen as follows:

$$Y_k(i,c) > 0 \Rightarrow X_k(i,c) = (\Lambda_k(i,c), \lambda_k(i)) = (L_{ic}, 0),$$

$$Y_{ik}(i,c) = 0 \Rightarrow (\Lambda_k(i,c), \lambda_k(i)) = (0,0).$$

The network approximates the set of desired output vectors in a manner that minimizes a cost function E_k:

$$E_k = \frac{1}{2}\sum_{i=1}^{n}\sum_{c=1}^{C}[q_k(i,c) - Y_k(i,c)]^2.$$

The rule to update the weights may be written as

$$\begin{aligned} w_k^+(u,p;v,z) &= w_{k-1}^+(u,p;v,c) - \mu\sum_{i=1}^{n}\sum_{c=1}^{C} \\ &\quad \times[q_k(i,c) - Y_k(i,c)][\partial q(i,c)/\partial w^+(u,p;v,z)]_k, \\ w_k^-(u,p;v) &= w_{k-1}^-(u,p;v) - \mu\sum_{i=1}^{n}\sum_{c=1}^{C}[q_k(i,c) - Y_k(i,c)] \\ &\quad \times[\partial q(i,c)/\partial w^-(u,p;v)]_k, \end{aligned} \tag{3}$$

where $\mu > 0$ is the learning rate (some constant),

$q_k(i)$ is calculated using X_k, $w_k^+(u,p;v,z) = w_{k-1}^+(u,p;v,z)$ and

$w_k^-(u,p;v) = w_{k-1}^-(u,p;v)$ in (2)

$[\delta q(i,c)/\delta w^+(u,p;v,z)]_k$ and $[\delta q(i,c)/\delta w^-(u,p;v)]_k$ are evaluated using the values

$q(i,c) = q_k(i,c), w_k^+(u,p;v,z) = w_{k-1}^+(u,p;v,z)$ and

$w_k^-(u,p;v) = w_{k-1}^-(u,p;v)$ in (3).

The complete learning algorithm for the network is

- *Initiate the matrices W_0^+ and W_0^- in some appropriate manner. Choose a value of μ in* (3).
- *For each successive value of m*:
- *Set the input–output pair* (X_k, Y_k)
- *Repeat*
- *Solve Eq.* (2) *with these values*
- *Using* (3) *and the previous results update the matrices W_k^+ and W_k^-*

Until the change in the new values of the weights is smaller than some predetermined valued.

For more details about this learning algorithm, see [5].

3.2. Retrieval procedure

Once the learning phase is completed, the network must perform the completion of noisy versions of the training vectors as well as possible. In this case, we propose a progressive retrieval process with adaptive threshold value. Let $X' = \{X'(1,1), \ldots, X'(n,C)\}$ be any input vector with values equal to 0, 0.5 or 1, for each $X'(i,c)$, $i = 1, \ldots, n$ and $c = 1, \ldots, C$. In order to determine the corresponding output vector $Y = \{Y(1,1), \ldots, Y(n,C)\}$, we first compute the vector of probabilities $Q = (q(1,1), \ldots, q(n,C))$. We consider that $q(i,c)$ values such that $1 - T < q(i,c) < T/2$ or $1 - T/2 < q(i,c) < T$, with for instance $T = 0.8$, belong to the uncertainty interval Z. When the network stabilizes to an attractor state, the number NB_Z of neurons whose $q(i,c) \in Z$, is equal to 0. Hence, we first treat the neurons whose state is considered certain to obtain the output vector $Y^{(1)} = (Y^{(1)}_{(1,1)}, \ldots, Y^{(1)}_{(n,C)})$, with

$$Y^{(1)}_{(i,c)} = F_z(q(i,c)) = \begin{cases} 1 & \text{if } q(i,c) > T, \\ 0 & \text{if } q(i,c) < 1 - T, \\ 0.5 & \text{if } T/2 \Leftarrow q(i,c) \Leftarrow 1 - T/2, \\ x'_i & \text{otherwise}, \end{cases}$$

where F_z is the thresholding function by intervals. If NB_Z $= 0$, this phase is terminated and the output vector is $Y = Y^{(1)}$. Otherwise, Y is obtained after applying the thresholding function f_β as follows:

$$Y(i,c) = f_\beta(q(i,c)) = \begin{cases} 1 & \text{if } q(i,c) > \beta, \\ 0.5 & \text{if } \beta/2 < q(i,c) < \beta, \\ 0 & \text{otherwise}, \end{cases}$$

where β is the selected threshold. Each value $q(i,c) \in Z$ is considered as a potential threshold. That is, for each $q(i,c) \in Z$:

$$\beta = \begin{cases} q(i,c) & \text{if } q(i,c) > 0.666, \\ 1 - q(i,c) & \text{otherwise}. \end{cases}$$

Eventually, Z can be reduced by decreasing T (for $T > 0.666$). For each potential value of β, we present to the network the vector $X'^{(1)}(\beta) = f_\beta(Q)$. Then, we compute the new vector of probabilities $Q^{(1)}(\beta)$ and the output vector $Y^{(2)}(\beta) = F_z(Q^{(1)}(\beta))$. We keep the cases where NB_Z $= 0$ and $X'^{(1)}(\beta) = Y^{(2)}(\beta)$. If these two conditions are never satisfied, the initial X' is considered too different from any training vector. If several thresholds are candidates, we choose the one which provides the minimal error (difference between $q(i,c)$ and $Y(i,c)$, for $i = 1, n$ and $c = 1, \ldots, C$):

$$E(\beta) = \frac{1}{2} \sum_{i=1}^{n} [q(i,c)^{(1)}(\beta) - Y(i,c)^{(1)}(\alpha)]^2.$$

4. Experimental results

4.1. Problem definition

In this section, we present several examples to compare the quality of our recognition algorithm for different pattern types. We will input various geometric figures to a multiple classes RNN and train the network to recognize these as separate categories. To evaluate our approach, we use three figure groups: for the first set of figures (group A), we use the set of figures shown in Fig. 1, where blackened boxes represent blue colors, gray boxes represent green colors and white boxes represent red colors. The next group (Group B) is composed for the black and white figures shown in Fig. 2, to compare our approach with the results obtained with the recognition algorithm based on RNN proposed in [3], and the evolutionary learning approach proposed in [5]. The last group is composed by the set of patterns used in [17] (see Fig. 3). For the last case, extended experiments are presented to evaluate the performance of our method according to the relationship between the problem features (number of patterns, pixels and colors), recognition rates and processing time. Each pixel is represented by a neuron and we suppose that three classes represent the primary colors (red, green, and blue) according to the RGB model.

For the first and second case, each figure is represented by a $6*6$ grid of pixels. For example, to represent the seventh geometric figure of Fig. 2, we must use the pattern

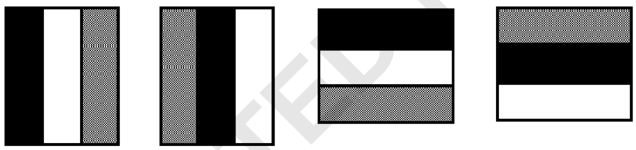


Fig. 1. Geometric figures with three colors (Group A).

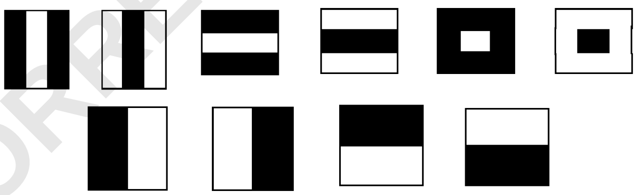
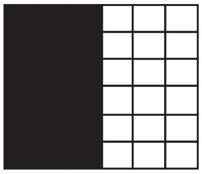


Fig. 2. Geometric figures with two colors (Group B).

Fig. 3. Pattern set used in the last experiment (Group C).



=>

111 111 111 000 000 000
111 111 111 000 000 000
111 111 111 000 000 000
111 111 111 000 000 000
111 111 111 000 000 000
111 111 111 000 000 000

Fig. 4. Representation of a geometric figure with a $6 * 6$ pattern.

Table 1
Recognition rate of noisy versions of Group A

Noisy rate	0%			10%			20%			30%		
Number of figures	4	6	10	4	6	10	4	6	10	4	6	10
Group A	99%	99%	97%	94%	93%	89%	83%	82%	81%	73%	71%	67%

Table 2
Recognition rate of noisy versions of Group B

Noisy rate	0%			10%			20%			30%		
Number of figures	4	6	10	4	6	10	4	6	10	4	6	10
Cl	99%	95%	96%	93%	91%	85%	83%	80%	77%	68%	66%	62%
Evol	99%	99%	99%	96%	95%	92%	87%	85%	81%	75%	74%	72%
Mult	99%	99%	99%	95%	94%	90%	85%	84%	81%	73%	72%	70%

shown in Fig. 4. According to the RGB model, the blackened boxes are represented as $(1, 1, 1)$, while white boxes are represented as $(0, 0, 0)$. In this way, we can represent geometric figures with different combinations of colors (for example, for Fig. 2 if we suppose blackened boxes correspond to red colors, and white boxes to blue colors, neurons for blackened boxes are equal to $(1, 0, 0)$ and for white boxes to $(0, 0, 1)$). Thus, for these cases we use a single-layer multiple classes RNN composed by 36 neurons ($n = 36$) and 3 classes ($C = 3$).

4.2. Results analysis

In order to test associative memories, we have evaluated the recognition rates of distorted versions of training patterns (Tables 1 and 2). We generated 20 noisy images used as inputs, for each training image and for a given distortion rate. The result of the learning stage is used as the initial neural network of this second phase (retrieval stage). We have corrupted them by reasonable noise rates equal to 0%, 10%, 20% and 30% distortion by modifying bit values at random. A pattern is recognized if the residual-error rate is less than 3%. The results we have obtained are presented in Tables 1 and 2. These values represent an average of eight processes for each set S_i of images.

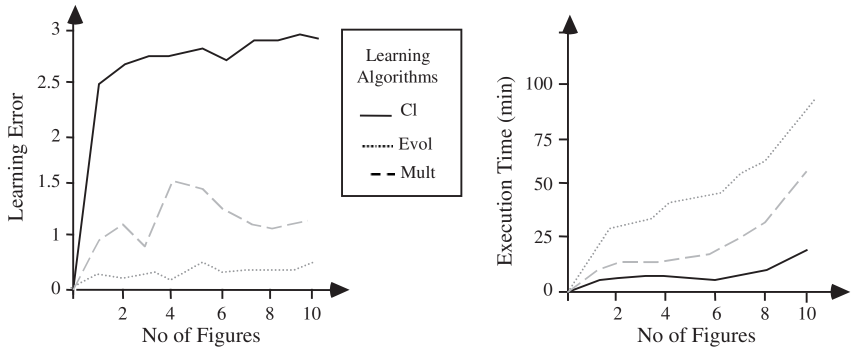


Fig. 5. Learning error and execution time of the learning algorithms for Group B.

The performance results obtained are lower when the noise rate is important (memories are then more discriminating). The results for the first group are presented in Table 1. Our algorithm provides a good recognition rate. Particularly, the recognition rate of the sets S_4 and S_6 remain good for our approach. Concerning S_{10} and 30% of noise rate, recognition-rate decreases.

In Table 2 the recognition rate for the last group of images (Group B) is shown, using the classical gradient recognition algorithm (*Cl*), the hybrid genetic/RNN learning algorithm (*Evol*) and our multiple classes learning algorithm (*Mult*). In general, *Evol* appears to give the best results. The recognition rate remains good for our algorithm (*Mult*). This algorithm provides a better recognition rate that *Cl*. Concerning *Cl*, the recognition rate is bad.

In Fig. 5 the system errors during the learning phase for Group B are shown, using the classical gradient decent learning algorithm (*Cl*), the hybrid genetic/RNN learning algorithm (*Evol*) and our multiple classes learning algorithm (*Mult*). *Evol* gives the best learning rate, but with a substantially large execution time. That is because *Evol* is very slow to converge. The learning error remains good for our learning algorithm (*Mult*). This algorithm provides a better error convergence of the learning phase that *Cl*. As regards *Cl*, error costs are an important reason for the bad recognition rate.

In Table 3 the relationship between the problem features (number of pixels and colors), recognition rates and processing time for the set of figures of the group C is shown. If we increase the number of pixels to describe a pattern, we improve the quality of the retrieval phase, but the execution time increases exponentially. The number of colors is not important because our system does not depend on it. If the RGB model can represent the specific color of a given pattern, our approach can recognize it (see the similarity of the recognition rates for the cases of the Figs. 1–3 and 6–8). When the patterns are different (patterns 6, 7 and 8 of the Fig. 3) the system has a better retrieval rate. Our approach can recognize several patterns, but with a large retrieval time if we like to obtain good retrieval rates.

Our system has a typical associative memory approach problem: low storage capability. If we compare the quality of our results with the method proposed on [8,17], their

Table 3
Performance evaluation of our method with the noisy versions of Group C

Noisy rate	0%				10%				20%			
Figures	1–8	1–3	6–8	1–5	1–8	1–3	6–8	1–5	1–8	1–3	6–8	1–5
Number of pixels	144	144	144	144	144	144	144	144	144	144	144	144
Recognition rates	95%	98%	98%	94%	88%	92%	95%	90%	78	83%	84%	80%
Retrieval time (sec)	240	31	33	120	300	34	43	60	234	33	38	55
Number of pixels	576	576	576	576	576	576	576	576	576	576	576	576
Recognition rates	95%	99%	99%	95%	89%	93%	96%	91%	79%	85%	86%	82%
Retrieval time	720	120	123	212	723	114	132	221	756	134	144	321
Number of pixels	2304	2304	2304	2304	2304	2304	2304	2304	2304	2304	2304	2304
Recognition rates	99%	99%	99%	99%	95%	97%	97%	96%	92%	95%	95%	92%
Retrieval time	9188	1302	1287	2341	8923	1100	1098	2178	9012	1231	1101	2111

approach has a better storage capability (they tested their approach for 30 patterns), but our recognition-rate quality is better ($\geqslant 90\%$ for 20% of noisy rate).

5. Conclusions

In this paper, we have proposed a recognition algorithm based on the multiple classes random neural model. The main contribution is the proposition of a learning algorithm and a retrieval procedure for the recognition problem. We have shown that this model can efficiently work as associative memory, and that a backpropagation learning approach is useful for this problem. In general, our approach gives better results than the other mentioned works in our paper, due to its capabilities to describe the image to be recognized. We can recognize arbitrary color images, but the processing time will increase rapidly according to the number of pixels used. The number of neurons is dictated by image resolution, and it has a direct influence on the quality of performance of our approach. During the learning phase, we have met classical problems found in supervised learning approaches like the existence of local minimal and large learning times. At the level of retrieval algorithm, we have obtained good performance but with a large execution time. However, most of the computations are intrinsically parallel and can be implemented on SIMD or MIMD architectures. At this moment we work in a data-parallel version of our approach.

Acknowledgements

This work was partially supported by FONACIT grant AP-97003817, CDCHT-ULA grant I I-621-98-02-A.

References

[1] J. Aguilar, Evolutionary learning on recurrent random neural network, in: Proceedings of the World Congress on Neural Networks, 1995, pp. 232–236.
[2] J. Aguilar, An energy function for the random neural networks, Neural Process. Lett. 4 (1996) 17–27.
[3] J. Aguilar, A recognition algorithm using the random neural network, in: Proceedings of the 3rd International Congress on Computer Science Research, 1996, pp. 15–22.
[4] J. Aguilar, Definition of an energy function for the random neural to solve optimization problems, Neural Networks 11 (1998) 731–738.
[5] J. Aguilar, Learning algorithms for the multiple classes random neural network, Lect. Notes Artif. Intell. 1821 (2000) 561–566.
[6] J. Aguilar, A. Colmenares, Resolution of pattern recognition problems using a hybrid genetic/random neural network learning algorithm, Pattern Anal. Appl. 1 (1998) 52–61.
[7] V. Atalay, E. Gelenbe, N. Yalabik, The random neural network model for texture generation, Int. J. Pattern Recogn. Artif. Intell. 6 (1992) 131–141.
[8] C. Colombo, A. Del Bimbo, Color-induced image representation and retrieval, Pattern Recogn. 32 (1999) 1685–1695.
[9] A. Del Bimbo, P. Pala, Visual image retrieval by elastic matching of user sketches, IEEE Trans. Pattern Anal. Machine Intell. 19 (1997) 223–234.
[10] M. Fourneau, E. Gelenbe, R. Suros, G-networks with multiple classes of negative and positive customers, Theor. Comput. Sci. 155 (1996) 141–156.
[11] E. Gelenbe, Random neural networks with positive and negative signals and product form solution, Neural Comput. 1 (1989) 502–511.
[12] E. Gelenbe, Stability of the random neural networks, Neural Comput. 2 (1990) 239–247.
[13] E. Gelenbe, Theory of the random neural network model, in: E. Gelenbe (Ed.), Neural Networks: Advances and Applications, North-Holland, Pays-Bas, Amsterdam, 1991.
[14] E. Gelenbe, Learning in the recurrent random neural network, Neural Comput. 5 (1993) 376–389.
[15] A. Jain, A. Vailaya, Image retrieval using color and shape, Pattern Recogn. 29 (1996) 1233–1244.
[16] T. Minka, R. Picard, Interactive learning with a society of models, Pattern Recogn. 30 (1997) 1370–1381.
[17] A. Mojsilovic, J. Kovacevic, D. Kall, R. Safranek, K. Ganapathy, The vocabulary and grammar of color patterns, IEEE Trans. Image Process. 9 (2000) 417–431.