ELSEVIER Journal of Information Sciences 109 (1998) 165–184

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

Simulated annealing based pattern classification

Sanghamitra Bandyopadhyay [a], Sankar K. Pal [a,*], C.A. Murthy [b,1]

[a] *Machine Intelligence Unit, Indian Statistical Institute, 203 Barrackpore Trunk Road, Calcutta 700 035, India*

[b] *Department of Statistics, 326, Thomas Building, Pennsylvania State University, University Park, PA-16802-111, USA*

Received 11 February 1997; received in revised form 20 September 1997; accepted 18 November 1997

Abstract

A method is described for finding decision boundaries, approximated by piecewise linear segments, for classifying patterns in $\mathbb{R}^N, N \geqslant 2$, using Simulated Annealing (SA). It involves generation and placement of a set of hyperplanes (represented by strings) in the feature space that yields minimum misclassification. Theoretical analysis shows that as the size of the training data set approaches infinity, the boundary provided by the SA based classifier will approach the Bayes boundary. The effectiveness of the classification methodology, along with the generalization ability of the decision boundary, is demonstrated for both artificial data and real life data sets having non-linear/overlapping class boundaries. Results are compared extensively with those of the Bayes classifier, k-NN rule and multilayer perceptron, and Genetic Algorithms, another popular evolutionary technique. Empirical verification of the theoretical claim is also provided. © 1998 Elsevier Science Inc. All rights reserved.

Keywords: Pattern recognition; Evolutionary computation; Genetic algorithm; Bayes error probability

[*] Corresponding author. Tel.: +91 33 577 8085 3100; fax: +91 33 556 6680/6925; e-mail: sankar@isical.ac.in.

[1] On leave from Indian Statistical Institute.

PII: S0020-0255(98)00017-6

1. Introduction

Simulated Annealing (SA) [1–4] belongs to a class of local search algorithm. It utilizes the principles of statistical mechanics, regarding the behaviour of a large number of atoms at low temperature, for finding minimal cost solutions to large optimization problems by minimizing the associated energy. Let $E(q,T)$ be the energy at temperature T when the system is in the state q. Let a new state s be generated. Then state s is accepted in favour of state q with a probability

$$p_{qs} = \frac{1}{1 + \exp\left(\frac{(E(q,T)-E(s,T))}{T}\right)}.$$

In statistical mechanics investigating the ground states or low energy states of matter is of fundamental importance. These states are achieved at very low temperature. However, it is not sufficient to lower the temperature alone since this results in unstable states. In the annealing process, the temperature is first raised, then decreased gradually to a very low value (T_{min}), while ensuring that one spends sufficient time at each temperature value. This process yields stable low energy states.

Pattern classification can be viewed as a problem of search and placement of a number, H, of hyperplanes (fixed a priori) which can model the decision boundary of the given data set appropriately. The criterion to be minimized is the number of samples of the given training data that are misclassified for a particular arrangement of the H hyperplanes. The arrangement of hyperplanes that minimizes the number of misclassified data points is considered to provide the decision boundary of the given training data set.

The present article describes a methodology demonstrating the searching ability of SA for finding an appropriate arrangement of H hyperplanes that minimizes the number of misclassified points. The effectiveness of the classifier has been adequately established for several artificial and real life data sets with both overlapping and non-overlapping class boundaries. The results are also compared with a similar approach [5] based on genetic algorithms (GA) [4,6], Bayes maximum likelihood classifier, k-NN rule [7] and multilayered perceptron (MLP) [8].

Besides, a theoretical analysis alongwith an empirical verification is presented which shows that for the size of the training data set going to infinity, the SA based classifier (or *SA classifier*) will provide an error rate of the training data which is less than or equal to the Bayes error probability. (In this regard it may be mentioned here that *Bayes maximum likelihood classifier* [7] is one of the most widely used statistical pattern classifiers which provides optimal performance from the standpoint of error probabilities in a statistical framework. It is known to be the best classifier when the class distributions and the a priori probabilities are known. Consequently, the desirable property of any classifier is that it should approximate or approach the Bayes classifier under limiting conditions.)

A brief discussion on the principles of SA is first presented in Section 2. This is followed by a detailed description of the SA classifier. The theoretical analysis is provided in Section 4 followed by the implementation results in Section 5. Finally, the discussion and conclusions are presented the last section.

2. Simulated annealing: Basic principles

In the recent past, application of techniques having physical or natural correspondence for solving difficult optimization problems has received widespread attention. It has been found that these techniques consistently outperform classical methods like gradient descent search when the search space is large, complex and multimodal. SA is one such paradigm having its foundation in statistical mechanics, which studies the behaviour of a very large system of interacting components in thermal equilibrium.

In statistical mechanics, if the system is in thermal equilibrium, the probability $\pi_T(s)$ that the system is in state s, $s \in S$, S being the state space, at temperature T, is given by

$$\pi_T(s) = \frac{\mathrm{e}^{-E(s)/kT}}{\sum_{w \in S} \mathrm{e}^{-E(w)/kT}}, \tag{1}$$

where k is the Boltzmann's constant and $E(s)$ is the energy of the system in state s.

Metropolis et al. [9] developed a technique to simulate the behaviour of the system in thermal equilibrium at temperature T as follows: Let the system be in state q at time t. Then the probability p that it will be in state s at time $t+1$ is given by the equation

$$p = \frac{\pi_T(s)}{\pi_T(q)} = \exp\left(\frac{-(E(s) - E(q))}{kT}\right). \tag{2}$$

If the energy of the system in state s is less than that in state q, then $p > 1$ and the state s is automatically accepted. Otherwise it is accepted with probability p. Thus it is also possible to attain states with higher energy values. It can be shown that for $t \to \infty$, the probability that the system is in state s is given by $\pi_T(s)$ irrespective of the starting configuration [10].

When dealing with a system of particles, it is important to investigate very low energy states, which predominate at extremely low temperatures. To achieve such states, it is not sufficient to lower the temperature. An annealing schedule is used, where the temperature is first increased and then decreased gradually, spending enough time at each temperature in order to reach thermal equilibrium.

In this article we have used the annealing process of the Boltzmann machine, which is a variant of the Metropolis algorithm. Here, at a given temperature T, the new state is chosen with a probability

$$p_{qs} = \frac{1}{1 + \exp\left(\frac{(-E(q,T) - E(s,T))}{T}\right)}.$$

The parameters of the search space is encoded in the form of a bit string of a fixed length. The objective value associated with the string is computed and mapped to its energy. The string with the minimum energy value provides the solution to the problem. The initial string (say q) of 0s and 1s is generated randomly and its energy value is computed. Keeping the initial temperature high (say $T = T_{\max}$), a neighbour of the string (say s) is generated by randomly flipping one bit. The energy of the new string is computed and it is accepted in favour of q with a probability p_{qs} mentioned earlier. This process is repeated a number of times (say k) keeping the temperature constant. Then the temperature is decreased using the equation $T = rT$, where $0 < r < 1$, and the k loops, as earlier, are executed. This process is continued till a minimum temperature (say $T_{\min}$) is attained. The simulated annealing steps are shown in Fig. 1.

3. Description of the SA *classifier*

The correspondence between the physical aspect of SA and an optimization problem is as follows: the parameters of the search space (in this case the H

```
Begin
    generate the initial string randomly = $q$
    $T$ = $T_{max}$
    Let $E(q,T)$ be the associated energy
    while ($T \geq T_{min}$)
      for $i$ = 1 to $k$
        Mutate (flip) a random position in $q$ to yield $s$
        Let $E(s,T)$ be the associated energy
        Set $q \leftarrow s$ with probability $\frac{1}{1+e^{-(E(q,T)-E(s,T))/T}}$
      end for
      $T = rT$
    end while
    Decode the string $q$ to provide the solution of the problem.
End
```

Fig. 1. Steps of simulated annealing.

hyperplanes), are encoded in strings (usually binary) and these represent the different states; low energy states correspond to near optimal solutions (or an arrangement of the hyperplanes that provide minimum misclassification); the energy corresponds to objective function (or the number of misclassified samples), and temperature is a controlling parameter of the system. The important tasks here are to establish a way of representing and generating different configurations (or states) of the problem and an annealing schedule. These are now discussed in details.

3.1. State/hyperplane representation

In this article, binary string of length l is used to encode the parameters of the H hyperplanes. From elementary geometry, the equation of a hyperplane in N-dimensional space $(X_1 - X_2 - \cdots - X_N)$ is given by

$$x_N \cos \alpha_{N-1} + \beta_{N-1} \sin \alpha_{N-1} = d, \tag{3}$$

where $\beta_{N-1} = x_{N-1} \cos \alpha_{N-2} + \beta_{N-2} \sin \alpha_{N-2}$,

$$\beta_{N-2} = x_{N-2} \cos \alpha_{N-3} + \beta_{N-3} \sin \alpha_{N-3},$$

$$\vdots$$

$$\beta_1 = x_1 \cos \alpha_0 + \beta_0 \sin \alpha_0.$$

The various parameters are as follows: X_i is the ith feature of the training points; $(x_1, x_2, \ldots, x_N)$ a point on the hyperplane; α_{N-1} the angle that the unit normal to the hyperplane makes with the X_N axis; α_{N-2} the angle that the projection of the normal in the $(X_1 - X_2 - \cdots - X_{N-1})$ space makes with the X_{N-1} axis; and so on, α_1 the angle that the projection of the normal in the $(X_1 - X_2)$ plane makes with the X_2 axis; α_0 the angle that the projection of the normal in the (X_1) plane makes with the X_1 axis $= 0$. Hence, $\beta_0 \sin \alpha_0 = 0$; and d the perpendicular distance of the hyperplane from the origin. Thus the N tuple $\langle \alpha_1, \alpha_2, \ldots, \alpha_{N-1}, d \rangle$ specifies a hyperplane in N-dimensional space.

Each angle α_j, $j = 1, 2, \ldots, N-1$ is allowed to vary in the range of $[0, 2\pi]$. If b_1 bits are used to represent an angle, then the possible values of α_j are

$$0, \delta\, 2\pi, 2\delta\, 2\pi, 3\delta\, 2\pi, \ldots, (2^{b_1} - 1)\delta\, 2\pi,$$

where $\delta = 1/2^{b_1}$. Consequently, if the b_1 bits contain a binary string having the decimal value v_1, then the angle is given by $v_1\, \delta\, 2\pi$.

Once the angles are fixed, the orientation of the hyperplane becomes fixed. Now only d must be specified in order to specify the hyperplane. For this purpose the hyper rectangle enclosing the training points is considered. Let $(x_i^{\min}, x_i^{\max})$ be the minimum and maximum values of feature X_i as obtained from the training points. Then the vertices of the enclosing hyperrectangle are given by

$$(x_1^{ch_1}, x_2^{ch_2}, \ldots, x_N^{ch_N}),$$

where each ch_i, $i = 1, 2, \ldots, N$ can be either max or min. (Note that there will be 2^N vertices.) Let diag be the length of the diagonal of this hyperrectangle given by

$$\text{diag} = \sqrt{(x_1^{\max} - x_1^{\min})^2 + (x_2^{\max} - x_2^{\min})^2 + \cdots + (x_N^{\max} - x_N^{\min})^2}.$$

A hyperplane is designated as the *base hyperplane* with respect to a given orientation (i.e., for some $\alpha_1, \alpha_2, \ldots, \alpha_{N-1}$) if:

(i) it has the same orientation,

(ii) it passes through one of the vertices of the enclosing rectangle,

(iii) its perpendicular distance from the origin is minimum (among the hyperplanes passing through the other vertices). Let this distance be $d_{\min}$.

If b_2 bits are used to represent d, then a value of v_2 in these bits represents a hyperplane with the given orientation and for which d is given by $d_{\min} + (\text{diag}/2^{b_2})\, v_2$. Thus a string is of a fixed length of $l = H((N-1)\, b_1 + b2)$, where H = the number of hyperplanes. The initial string is generated randomly. Note that we have used this recursive form of representation over the classical one viz. $l_1x_1 + l_2x_2 + \cdots + l_Nx_N = d$, where $l_1, \ldots, l_N$ are known as the direction cosines. The latter representation involves a constraint equation, $l_1^2 + l_2^2 + \cdots + l_N^2 = 1$. This, in turn, leads to the complicated issue of getting invalid or unacceptable solutions when the constraint equation is violated. However, the representation that we have chosen avoids this problem by being unconstrained in nature.

3.2. Energy/objective value computation

A string encodes the parameters of H hyperplanes as described earlier. Using these parameters, the region in which each training pattern point lies is determined from Eq. (3). A region is said to provide the demarcation for class i, if maximum number of points that lie in this region belong to class i. Other points that lie in this region are considered to be misclassified. The misclassifications associated with all the regions (for these H hyperplanes) are summed up to provide the total misclassification, miss, for the string, which represents its energy.

3.3. New state generation process and annealing schedule

For generating a new configuration, one (or more) random position(s) in the bit string is chosen and flipped. This provides a new string, whose energy is computed in the above mentioned manner.

As already mentioned, the crucial task over here is the attainment of low energy states, obtained at very low temperatures. If the temperature is decreased quickly, then the low energy states tend to be unstable. In order to reach stable states, the temperature must be initially increased, and then decreased gradual-

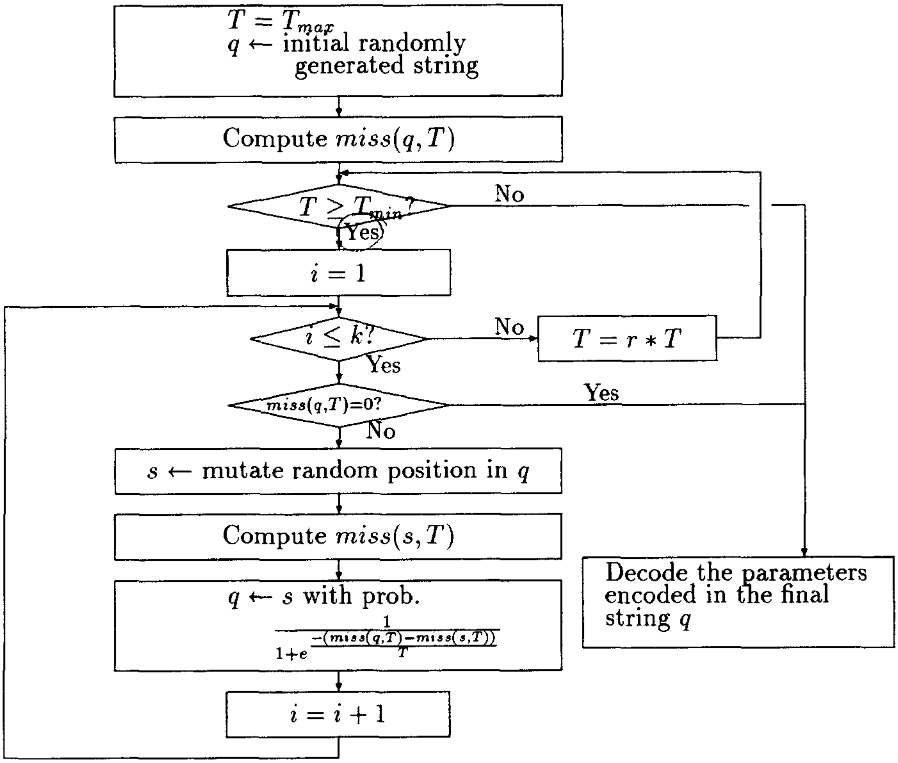
$T = T_{max}$
$q \leftarrow$ initial randomly generated string
Compute $miss(q,T)$
$T \geq T_{min}$?
No
Yes
$i = 1$
$i \leq k$?
No
$T = r * T$
Yes
$miss(q,T)=0$?
Yes
No
$s \leftarrow$ mutate random position in q
Compute $miss(s,T)$
$q \leftarrow s$ with prob. $\frac{1}{1+e^{\frac{-(miss(q,T)-miss(s,T))}{T}}}$
$i = i + 1$
Decode the parameters encoded in the final string q

Fig. 2. Steps of the SA classifier.

ly allowing sufficient time at each temperature. This process is known as *annealing*. In order to simulate this method, initially the temperature is kept high ($= T_{\max}$). A parameter k is used to control the time spent at each temperature value. The temperature is decreased according to the formula $T = rT$, where $0 < r < 1$. Higher value of r indicates a more gradual annealing schedule. The different steps of the SA classifier are shown in Fig. 2. The process continues until either a string with no misclassified points is obtained (miss $= 0$) or an user specified minimum temperature value ($= T_{\min}$) is attained. The final string q at termination provides the solution to the problem.

4. Relationship with Bayes error probability

In this section we study the theoretical relationship between the SA classifier and Bayes classifier in terms of the error probabilities. The mathematical notations and preliminary definitions are described first. This is followed by the claim that for $n \to \infty$ the performance of the SA classifier will no way be worse than that of Bayes classifier. Finally some critical comments about the proof are mentioned.

Let there be k classes $C_1, C_2, \ldots, C_k$ with a priori probabilities $P_1, P_2, \ldots, P_k$ and class conditional densities $p_1(x), p_2(x), \ldots, p_k(x)$. Let the mixture density be

$$p(x) = \sum_{i=1}^{k} P_i p_i(x). \tag{4}$$

Let $X_1, X_2, \ldots, X_n, \ldots$ be independent and identically distributed (i.i.d) N-dimensional random vectors with density $p(x)$. This indicates that there is a probability space $(\Omega, \mathscr{F}, Q)$, where $\mathscr{F}$ is a σ field of subsets of Ω, Q is a probability measure on $\mathscr{F}$, and

$$X_i\colon\ (\Omega, \mathscr{F}, Q) \to (\mathbb{R}^N, B(\mathbb{R}^N), P) \quad \forall i = 1, 2, \ldots$$

such that

$$P(A) = Q(X_i^{-1}(A)) = \int_A p(x)\,\mathrm{d}x$$

$\forall A \in B(\mathbb{R}^N)$ and $\forall i = 1, 2, \ldots$ Here $B(\mathbb{R}^N)$ is the Borel σ field of $\mathbb{R}^N$.

Let

$$\mathscr{E} = \{E\colon\ E = (S_1, S_2, \ldots, S_k), S_i \subseteq \mathbb{R}^N, S_i \neq \emptyset$$
$$\forall i = 1, \ldots, k, \bigcup_{i=1}^{k} S_i = \mathbb{R}^N, S_i \bigcap S_j = \emptyset, \forall i \neq j\}.$$

$\mathscr{E}$ provides the set of all partitions of $\mathbb{R}^N$ into k sets as well as their permutations, i.e.,

$$E_1 = (S_1, S_2, S_3 \ldots, S_k) \in \mathscr{E},$$
$$E_2 = (S_2, S_1, S_3, \ldots, S_k) \in \mathscr{E},$$

then $E_1 \neq E_2$. Note that $E = (S_{i_1}, S_{i_2}, \ldots, S_{i_k})$ implies that each S_{i_j}, $1 \leqslant j \leqslant k$, is the region corresponding to class C_j.

Let $E_0 = (S_{01}, S_{02}, \ldots, S_{0k}) \in \mathscr{E}$ be such that each S_{0i} is the region corresponding to the class C_i in $\mathbb{R}^N$ and these are obtained by using Bayes decision rule. Then

$$a = \sum_{i=1}^{k} P_i \int_{S_{0i}^c} p_i(x)\mathrm{d}x \leqslant \sum_{i=1}^{k} P_i \int_{S_{1i}^c} p_i(x)\mathrm{d}x \tag{5}$$

$\forall E_1 = (S_{11}, S_{12}, \ldots, S_{1k}) \in \mathscr{E}$. Here a is the error probability obtained using the Bayes decision rule.

It is known from the literature that such an E_0 exists and it belongs to $\mathscr{E}$ because Bayes decision rule provides an optimal partition of $\mathbb{R}^N$ and for every such $E_1 = (S_{11}, S_{12}, \dots, S_{1k}) \in \mathscr{E}, \sum_{i=1}^{k} P_i \int_{S_{1i}^c} p_i(x)\mathrm{d}x$ provides the error probability for $E_1 \in \mathscr{E}$. Note that E_0 need not be unique.

Assumptions 1. Let H_0 be a positive integer and let there exist H_0 hyperplanes in $\mathbb{R}^N$ which can provide the regions $S_{01}, S_{02}, \dots, S_{0k}$. Let H_0 be known a priori. Let the algorithm for generation of class boundaries using H_0 hyperplanes be allowed to be executed for a sufficiently large number of iterations in each temperature value and for sufficiently low temperatures. Let the number of strings be t with misclassification values $\mathrm{miss}_1, \mathrm{miss}_2, \dots, \mathrm{miss}_t$ where $0 \leqslant \mathrm{miss}_1 \leqslant \mathrm{miss}_2 \leqslant \cdots \leqslant \mathrm{miss}_t$. Let $p_{i,j}^{(n_1)}(T)$ denote the probability of going from string i to string j in n_1 steps with the temperature value T. It is known in the literature that for the adopted SA algorithm

$$\lim_{n_1 \to \infty} p_{i,j}^{(n_1)}(T) = p_{i,j}(T),$$

where $p_{i,j}(T) = \mathrm{e}^{-\mathrm{miss}_j/T} / \sum_{k=1}^{t} \mathrm{e}^{-\mathrm{miss}_k/T}$. It follows that

$$\begin{aligned} \lim_{T \to 0^+} p_{i,j}(T) &= 1 \quad \text{for } j = 1, \\ &= 0 \quad \text{for } j \neq 1. \end{aligned}$$

Thus it is known that using SA technique of (i) making $n_1 \to \infty$ and (ii) making $T \to 0^+$, one can get the optimal string and its value.

Let $\mathscr{A} = \{\mathscr{A}\colon \mathscr{A}$ is a set consisting of H_0 hyperplanes in $\mathbb{R}^N\}$. Let $A_0 \in \mathscr{A}$ be such that it provides the regions $S_{01}, S_{02}, \dots, S_{0k}$ in $\mathbb{R}^N$ i.e., A_0 provides the regions which are also obtained using the Bayer decision rule. Note that each $A \in \mathscr{A}$ generates several elements of $\mathscr{E}$. Let $\mathscr{E}_A \subseteq \mathscr{E}$ denote all possible $E = (S_1, S_2, \dots, S_k) \in \mathscr{E}$ that can be generated from A.

Let $G = \bigcup_{A \in \mathscr{A}} \mathscr{E}_A$. Let

$$Z_{iE}(\omega) = \begin{cases} 1 & \text{if } X_i(\omega) \text{ is misclassified when } E \text{ is used as a decision} \\ & \text{rule where } E \in G, \forall \omega \in \Omega. \\ 0 & \text{otherwise.} \end{cases}$$

Let $f_{nE}(\omega) = (1/n) \sum_{i=1}^{n} Z_{iE}(\omega)$, when $E \in G$ is used as a decision rule. Let $f_n(\omega) = \mathrm{Inf}\{f_{nE}(\omega)\colon E \in G\}$.

It is to be noted that the pattern classification algorithm mentioned in Section 2 uses $n \times f_{nE}(\omega)$, the total number of misclassified samples, as the objective function which it attempts to minimize. This is equivalent to searching for a suitable $E \in G$ such that the term $f_{nE}(\omega)$ is minimized, i.e., for which $f_{nE}(\omega) = f_n(\omega)$. As already mentioned, it is known that for infinitely many iterations the Elitist model of GAs will certainly be able to obtain such an E.

Theorem 2. *For sufficiently large n, $f_n(\omega) \not> a$, (i.e., for sufficiently large n, $f_n(\omega)$ cannot be greater than a) almost everywhere.*

Proof. Let

$$\begin{aligned} Y_i(\omega) &= 1 \quad \text{if } X_i(\omega) \text{ is misclassified according to Bayes rule } \forall \omega \in \Omega. \\ &= 0 \quad \text{otherwise.} \end{aligned}$$

Note that $Y_1, Y_2, \ldots, Y_n, \ldots$ are i.i.d random variables. Now

$$\begin{aligned} \text{Prob}(Y_i = 1) &= \sum_{j=1}^{k} \text{Prob}(Y_i = 1/X_i \text{ is in } C_j) P(X_i \text{ is in } C_j) \\ &= \sum_{j=1}^{k} P_j \, \text{Prob}(\omega\text{: } X_i(\omega) \in S^c_{0j} \text{ given that } \omega \in C_j) \\ &= \sum_{j=1}^{k} P_j \int_{S^c_{0j}} p_j(x) \, \mathrm{d}x = a. \end{aligned}$$

Hence the expectation of Y_i, $E(Y_i)$ is given by

$$E(Y_i) = a \quad \forall i.$$

Then by using Strong Law of Large Numbers [11], $(1/n)\sum_{i=1}^{n} Y_i \to a$ almost everywhere, i.e.,

$$P\left(\omega\text{: } \frac{1}{n}\sum_{i=1}^{n} Y_i(\omega) \to a\right) = 0.$$

Let

$$B = \left\{\omega\text{: } \frac{1}{n}\sum_{i=1}^{n} Y_i(\omega) \to a\right\} \subseteq \Omega.$$

Then $Q(B) = 1$. Note that $f_n(\omega) \leqslant (1/n)\sum_{i=1}^{n} Y_i(\omega)$, $\forall n$ and $\forall \omega$, since the set of regions $(S_{01}, S_{02}, \ldots, S_{0k})$ obtained by the Bayes decision rule is also provided by some $A \in \mathscr{A}$ and consequently it will be included in G. Note that $0 \leqslant f_n(\omega) \leqslant 1$, $\forall n$ and $\forall \omega$. Let $\omega \in B$. For every $\omega \in B$, $U(\omega) = \{f_n(\omega); n = 1, 2, \ldots\}$ is a bounded, infinite set. Then by Bolzano–Weierstrass theorem [12], there exists an accumulation point of $U(\omega)$. Let $y = \text{Sup}\{y_0\text{: } y_0 \text{ is an accumulation point of } U(\omega)\}$. From elementary mathematical analysis we can conclude that $y \leqslant a$, since $(1/n)\sum_{i=1}^{n} Y_i(\omega) \to a$ almost everywhere and $f_n(\omega) \leqslant (1/n)\sum_{i=1}^{n} Y_i(\omega)$. Thus it is proved that for sufficiently large n, $f_n(\omega)$ cannot be greater than a for $\omega \in B$. □

It is to be mentioned that the theorem proved earlier indicates that as the size of the training data set is increased, the performance of the SA classifier will approach that of the Bayes classifier. The fact that $f_n(\omega) < a$ is true for only a finite number of sample points, since many distributions can generate these points. However, as the size of the data set goes to infinity, only one distribution can possibly generate all the points [13]. Also, since we know that Bayes classifier is the optimal one in a statistical framework, and there can be no better classifier, the above mentioned claim (that $f_n(\omega) \leqslant a$) can only indicate that $f_n(\omega) = a$; or in other words, the performance of the SA classifier will tend to that of the Bayes classifier in the limiting case. This, in turn indicates that under limiting conditions, the boundary provided by the SA classifier will approach the bayes boundary. This is experimentally demonstrated in Section 5.4.

Note: The term 'sufficiently large' is borrowed from statistics books and indicates mathematical term '$\rightarrow \infty$'.

5. Implementation and results

The three data sets used for demonstrating the effectiveness of the SA classifier are the following.

ADS 1: This two-dimensional artificial data set (Fig. 3) consists of 557 data points belonging to two classes. It is evident that the classes, which are separable, have non-linear class boundary.

Vowel data: This real life speech data consists of 871 Indian Telugu vowel sounds in six classes represented by $\{\delta, a, i, e, o, u\}$ [14]. It has three features corresponding to the first, second and third formant frequencies. Fig. 4 shows the data set in the first and second formant frequency plane.

Iris data: This four-dimensional data set for a specific category of irises has 150 points in three classes [15]. The features correspond to the sepal width and length and petal width and length in centimeters.

Data set 1: This two-dimensional data set, used for verifying the theoretical result in Section 4, is generated using a triangular distribution for the two classes, 1 and 2. The range for class 1 is $[0,2] \times [0,2]$ and that for class 2 is $[1,3] \times [0,2]$ with the corresponding peaks at (1,1) and (2,1). If P_1 is the a priori probability of class 1, then using elementary mathematics, we can show that Bayes classifier will classify a point to class 1 if its X coordinate is less than $1 + P_1$. This indicates that the Bayes decision boundary is given by

$$x = 1 + P_1. \tag{6}$$

5.1. Performance of SA classifier

The parameters of SA are as follows: $T_{\max} = 100, \quad T_{\min} = 0.01, r = 0.9, k = 100$.

Accordingly, the maximum number of iterations will be 8800. In order to generate a new string, one randomly chosen bit is flipped. The results shown are the average values of five different runs of the algorithm.

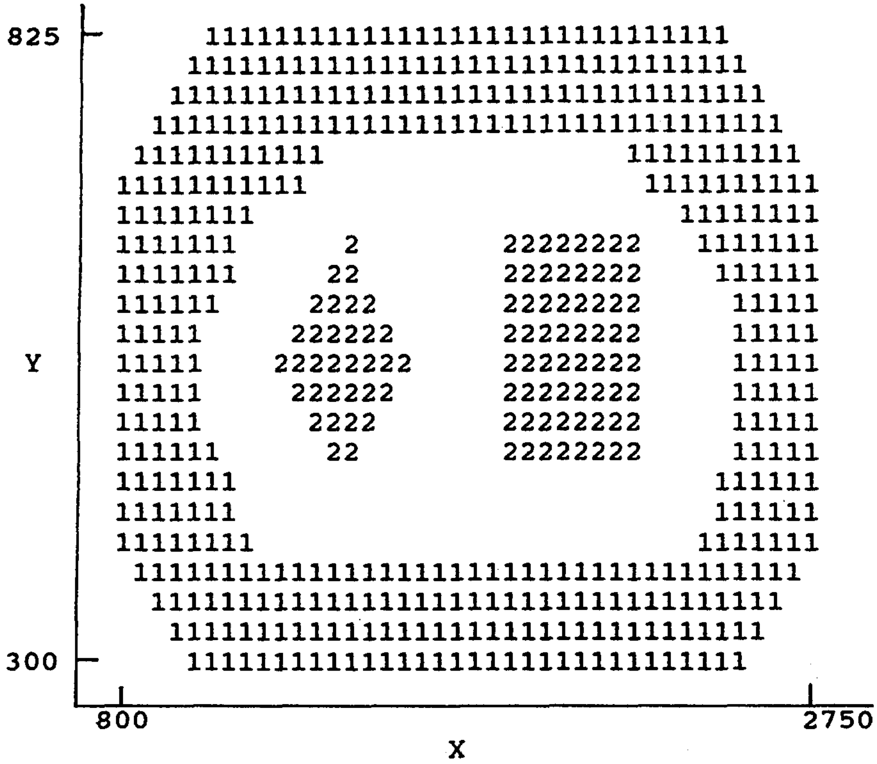

Fig. 3. Artificial data set ADS 1.

Table 1 shows the overall training performance of the SA classifier for data sets ADS 1, Vowel and Iris using five values of H when 10% of the data set is used for training. As expected, the training score generally improves to a maximum of 100% as the number of hyperplanes is increased, since more hyperplanes can readily fit the training data set to reduce the number of misclassified points. Note that because of the considerable amount of overlap, for the *Vowel data*, consideration of even $H = 8$ could not provide zero misclassification.

Tables 2 and 3 show the test results of the SA classifier for these three data sets, for five values of H, when 10% and 30% of the data set are used for training while the remaining 90% and 70% data are used for testing, respectively. Unlike the training performance, the test recognition score improves initially as H is increased upto a specific value, beyond which the score decreases. For example consider $H = 6$ and $H = 8$ of Table 2 for ADS 1 where the score decreases during testing, although it remained constant (at 100%) during training (Table 1). This indicates that $H = 8$ leads to overfitting of the classes

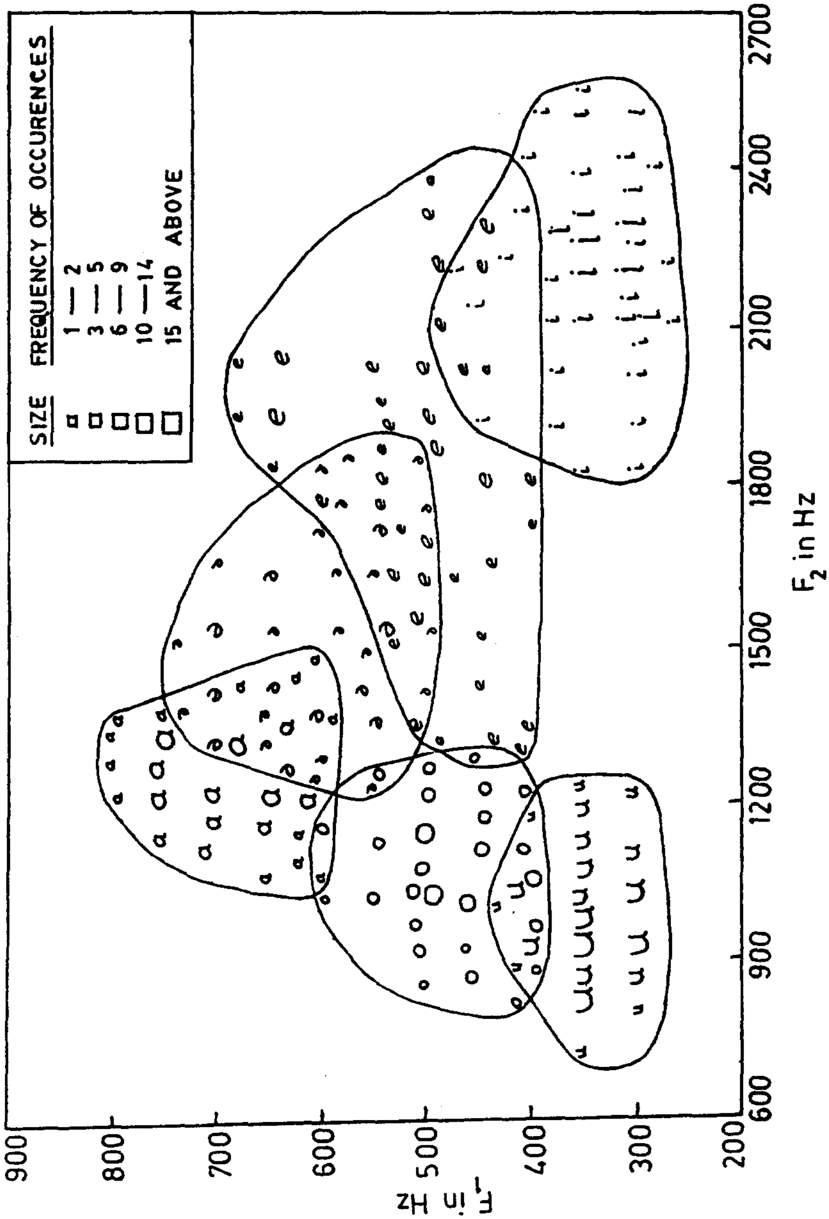

Fig. 4. Real life speech data. Vowel data, in the first and second formant frequency planes.

Table 1
Performance during training of SA classifier for different values of H using 10% training data

Data set	Recognition score				
	$H = 3$	$H = 4$	$H = 5$	$H = 6$	$H = 8$
ADS 1	94.54	98.18	100.0	100.0	100.0
Vowel	52.94	74.71	95.29	96.65	95.29
Iris	100.0	100.0	100.0	100.0	100.0

Table 2
Performance during testing of SA classifier for different values of H for 10% training and 90% test data

Data set	Recognition score				
	$H = 3$	$H = 4$	$H = 5$	$H = 6$	$H = 8$
ADS 1	91.63	92.23	93.02	93.02	88.64
Vowel	63.35	65.60	76.84	74.55	70.73
Iris	89.63	93.33	93.33	93.33	77.78

Table 3
Performance during testing of SA classifier for different values of H for 30% training and 70% test data

Data set	Recognition score				
	$H = 3$	$H = 4$	$H = 5$	$H = 6$	$H = 8$
ADS 1	91.28	96.92	98.72	96.41	96.20
Vowel	65.60	67.48	75.98	75.00	79.90
Iris	93.33	95.23	94.28	91.42	94.28

during training, thereby reducing the generalization capability of the classifier during testing. Similar is the case for $H = 6$ and 8 for ADS 1 in Table 3. As expected, the overall recognition capability of the classifier increases when the size of the training data set is increased from 10% in Table 2 to 30% in Table 3.

5.2. Replacing simulated annealing with genetic algorithm

Genetic Algorithm (GA) [6] is another evolutionary search paradigm, based on the principles of natural genetic systems and *survival of the fittest*. Like SA, GAs also generally work with a binary string encoding of the parameters of the search problem. Instead of dealing with a single string or *chromosome*, it operates on a number of strings termed *population*. A fitness value, which is maximized, is associated with each string which represents the degree of goodness associated with it. Several biologically inspired operators like *selection*, *crossover* and *mutation* are applied iteratively over a number of generations to generate potentially better solutions. Termination is achieved if either a maximum number of iterations has been executed or a user specified criterion is satisfied. Details of the method can be found in [4,6].

The fitness computation method is the same as the process of calculating the energy associated with a string (see Section 3.2). Roulette wheel selection strategy, single point crossover strategy with probability 0.8 and bit wise mutation with a variable mutation probability value in the range [0.015,0.333] [5] for a population size of 20 are chosen for the GA. The maximum number of iterations is fixed at 1500. The comparative performance (in terms of both the test score and number of iterations required for attaining zero misclassification during training) of SA and GA for the classification problem is presented in Table 4, when 10% data are considered for training and the remaining 90% for testing. An entry '-' in iter. field indicates that zero misclassification could not be achieved even after the maximum number of iterations was executed.

As is evident from Table 4, the test recognition scores of both GA and SA based classifiers are comparable. Although, the iterations required to attain zero misclassification for GA is less than that for SA, the number of string evaluations is much more since one iteration of GA corresponds to a maximum of 20 strings, which is the size of the population. On the other hand, exactly one new string is evaluated in each iteration of SA. On this count, GA requires at most 10 240 and 440 string evaluations for ADS 1 and Iris respectively, which is significantly more than that required in SA. However, one must note that of the 10 240 (or 440) strings evaluated by GA for ADS 1 (or Iris) there will be many replications. In fact, only a relatively small fraction of the strings will be unique.

Table 4
Comparative performance of SA and GA for classification for $H = 6$

Data set	GA		SA	
	iter.	score	iter.	score
ADS 1	512	93.22	5815	93.02
Vowel	–	71.99	–	74.55
Iris	22	93.33	97	93.33

5.3. Comparison with other classifiers

The performance of the SA classifier is compared to Bayes maximum likelihood classifier, Multilayered Perceptron (MLP) and k-NN rule. Both MLP (with hard delimiters) and k-NN rule are known to provide piecewise linear boundaries, which is the underlying philosophy of the SA classifier. k-NN algorithm is executed taking k equal to $\sqrt{n}$, where n is the number of training data points. It can be proved that for such a form of k, the error probability of the k-NN rule approaches the Bayes error probability. For the Bayes maximum likelihood classifier, unequal dispersion matrices and unequal a priori probabilities ($= n_i/n$ for n_i patterns from class i), are considered. In each case, we assume a multivariate normal distribution of the samples.

For MLP, learning rate and momentum factor are 0.9 and 0.1, respectively. Online connection weight updation, i.e., updation after the presentation of each training data point, is performed. A maximum of 10 000 iterations are allowed. The network architectures for ADS 1, Vowel and Iris data sets are 2-5-2, 3-8-6 and 4-5-3 respectively, where the first and the last numbers represent the number of nodes in the input and output layers, and the intermediate number(s) represent the number of nodes in the hidden layer(s).

The results in Table 5 show that the SA classifier provides superior performance to all the other classifiers for both ADS 1 (where k-NN is known to perform well) and Iris. For the Vowel Data, the result of the Bayes classifier is the best. In fact, the Bayes classifier is known to perform well for this data [14]. In this case also, the recognition score of the SA classifier is found to be closer to the Bayes score as compared to MLP and k-NN.

5.4. Empirical verification of the theoretical result

As a consequence of Theorem in Section 4, the boundary provided by the SA classifier approaches the Bayes boundary under limiting conditions. Fig. 5(a)–(c) demonstrates that this is indeed the case for the *Data Set 1*. The Bayes boundary is a straight line $x = 1.4$. The SA line is marked with an arrow, Figs. 5(a)–(c) show the SA lines obtained for $n = 100$, 1000 and 4000, respectively. Only 100 data points are plotted in the figures for clarity.

Table 5
Comparative test performance with 10% training data

Data set	SA classifier for $H = 6$	Bayes max. like. class.	MLP	k-NN $k = \sqrt{n}$
ADS 1	93.02	85.65	82.47	90.23
Vowel	74.55	77.73	60.30	70.35
Iris	93.33	83.22	74.81	90.37

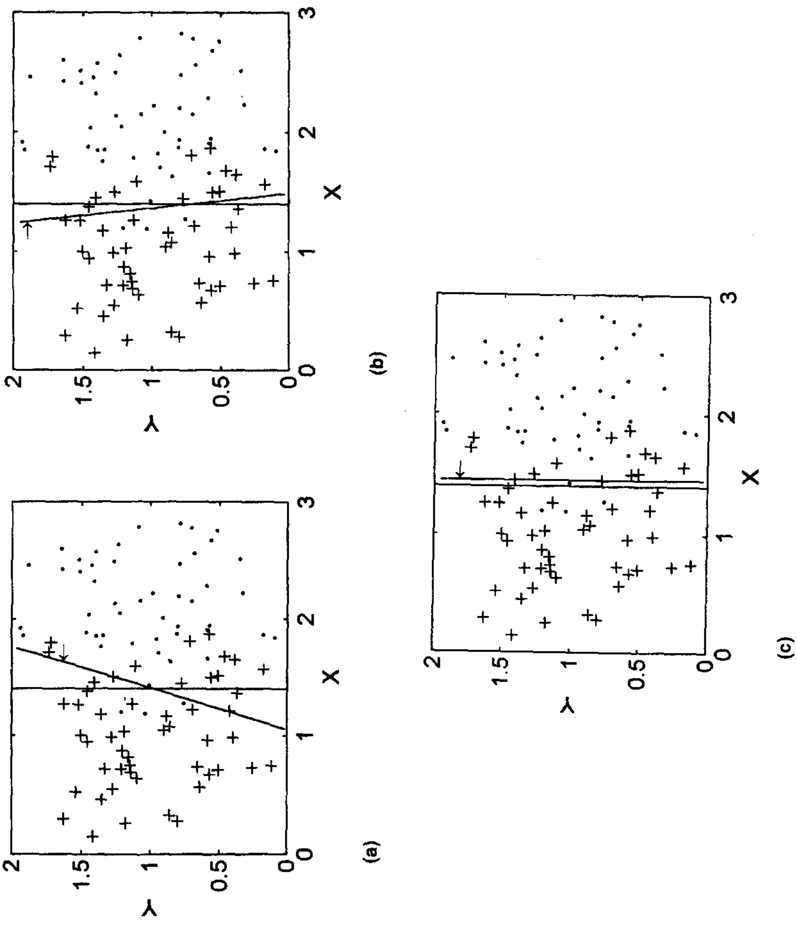

Fig. 5. (a) Data Set 1 for $n = 100$ and the boundary provided by SA classifier (marked with an arrow) along with Bayes decision boundary. (b) Data Set 1 for $n = 1000$ and the boundary provided by SA classifier (marked with an arrow) along with Bayes decision boundary. (c) Data Set 1 for $n = 4000$ and the boundary provided by SA classifier (marked with an arrow) along with Bayes decision boundary (class 1 is represented by '+' and class 2 by 'o').

It is obvious from the figures that as n increases from 100 to 4000, the SA line also approaches the Bayes line, so much so, that for $n = 4000$, they lie very close to each other.

6. Discussion and conclusions

A pattern classification methodology in $\mathbb{R}^N$, using simulated annealing for search and placement of a number of hyperplanes in order to approximate the class boundaries of a given training data set, has been described. An extensive comparison of the methodology with other classifiers, namely the Bayes classifier (which is well known for discriminating overlapping classes), k-NN classifier and MLP (which are well known for discriminating non-overlapping, non-linear regions by generating piecewise linear boundaries) is also presented. The results of the proposed algorithm are seen to be comparable to, sometimes better than, them in discriminating both overlapping and non-overlapping, non-convex regions.

A distinguishing feature of this approach is that the boundaries (approximated by piecewise linear segments) need to be generated explicitly for making decisions. This is unlike the conventional methods or the MLP based approaches, where the generation of boundaries is a consequence of the respective decision making processes.

A theoretical analysis of the aforesaid classifier establishes that under limiting conditions of infinitely large training data sets, the error rate of the SA classifier during training is less than or equal to that of the Bayes classifier. This, in turn, indicates that when the size of the training data set goes to infinity, the boundary provided by the SA classifier approaches the Bayes boundary. This finding is also experimentally verified for a data set, generated using triangular distribution, where the Bayes boundary is known exactly.

A comparison of SA with GA for this classification problem shows that both perform comparably in terms of the test recognition scores. This is expected, since both are stochastic optimization techniques, working on the same principle of approximating the class boundaries using a number of hyperplanes. In terms of string evaluations required to obtain the optimal performance, SA appears to score over GA. However, one must note that it is very difficult to obtain the actual number of distinct string evaluations in GA, since strings are often replicated. The actual number of distinct evaluations will, in fact, be a small fraction of the quantity (population size × number of iterations).

Although SA is found to perform comparably to GA, there appear to be several factors contributing to the predominance of GAs in the literature. In SA, two main control parameters are to be selected appropriately in order to obtain good performance. These are the values of r (which controls the sequence of T) and k (the number of iterations executed at each temperature). On the other hand, in GA, only the maximum number of iterations (or stopping time) must be appropriately selected. Other than this, both the methods need proper tuning of several other parameters, e.g., $T_{\max}$, $T_{\min}$ in SA, probabilities for crossover, mutation in GA, etc. Additionally, in the advanced stages of the SA algorithm, the temperature values should be smaller than the smallest difference of the energy values in order to provide good performance. Since for the pattern classification problem, this value is 1 (minimum non-zero difference of number of misclassified points) and $T_{\min} = 0.01$, this requirement is met. GA, with roulette wheel selection, is, on the other hand, immune to this difference. Finally, since SA is inherently sequential in nature, not much improvement can be derived in parallel computing platforms, while there is scope for such improvement in GA. One must note that very basic versions of both SA and GA are used here. Use of enhanced models and improved operators for both SA and GA may provide better performance. For example, in case of SA, other cooling schedules [16,17] may be used. Similarly, modified versions of GA, like genetic algorithm with chromosome differentiation, (GACD), may be applied which has been found to improve the classification performance [18].

Acknowledgements

This work was carried out when Ms. Sanghamitra Bandyopadhyay held a fellowship awarded by the Department of Atomic Energy, Govt. of India.

References

[1] P.J.M. van Laarhoven, E.H.L. Aarts, Simulated Annealing: Theory and Applications, Reidel, Dordrecht, 1987.

[2] S. Kirkpatrick, C.D. Gelatt, Jr., M.P. Vechhi, Optimization by simulated annealing, Science 220 (1983) 671–680.

[3] L. Davis, ed., Genetic Algorithms and Simulated Annealing, Pitman Publishing, London, 1987.

[4] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer, Berlin, 1992.

[5] S. Bandyopadhyay, C.A. Murthy, S.K. Pal, Pattern classification using genetic algorithms, Patt. Recog. Lett. 16 (8) (1995) 801–808.

[6] D.E. Goldberg, Genetic Algorithms: Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.

[7] J.T. Tou, R.C. Gonzalez, Pattern Recognition Principles, Addison-Wesley, Reading, MA, 1974.

[8] J. Hertz, A. Krogh, R.G. Palmer, Introduction to the theory of neural computation, Addison-Wesley, Reading, MA, 1991.

[9] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, E. Teller, Equations of state calculations by fast computing machines, J. Chem. Phys. 21 (1953) 1087–1091.

[10] S. Geman, D. Geman, Stochastic relaxation, gibbs distribution, and the bayesian restoration of images, IEEE Trans. Patt. Anal. Machine Intell. PAMI- 6 (1984) 721–741.

[11] R.O. Duda, P.E. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973.

[12] T.M. Apostol, Mathematical Analysis, Narosa, New Delhi, 1985.

[13] R.B. Ash, Real Analysis and Probability, Academic Press, New York, 1972.

[14] S.K. Pal, S. Mitra, Multilayer perceptron, fuzzy sets and classification, IEEE Trans. Neural Networks 3 (5) (1992) 683–697.

[15] R.A. Fisher, The use of multiple measurements in taxonomic problems, Ann. Eugenics 3 (1936) 179–188.

[16] S. Shinomoto, Y. Kabashima, Finite time scaling of energy in simulated annealing, J. Phys. A 24 (1991) L141–L144.

[17] E.H.L. Aarts, P.J.M. van Laarhoven, A pedestrian review on the theory and applications of simulated annealing algorithm, in: J. van Hemmen, I. Morgenstern (Eds.), Heidelberg Colloquim on Glassy Dynamics, Springer, Berlin, 1987, pp. 288–307.

[18] S. Bandyopadhyay, S.K. Pal, Pattern classification with genetic algorithms: Incorporation of chromosome differentiation, Pattern Recog. Lett. 18 (1997) 119–131.