Unsupervised Classifier Based on Heuristic Optimization and Maximum Entropy Principle

Edwin Aldana-Bobadilla and Angel Kuri-Morales

Universidad Nacional Autónoma de México, Mexico City, Mexico
Instituto Técnologico Autónomo de México, México City, Mexico
ealdana@uxmcc2.iimas.unam.mx,
akuri@itam.mx

Abstract. One of the basic endeavors in Pattern Recognition and particularly in Data Mining is the process of determining which unlabeled objects in a set do share interesting properties. This implies a singular process of classification usually denoted as "clustering", where the objects are grouped into k subsets (clusters) in accordance with an appropriate measure of likelihood. Clustering can be considered the most important unsupervised learning problem. The more traditional clustering methods are based on the minimization of a similarity criteria based on a metric or distance. This fact imposes important constraints on the geometry of the clusters found. Since each element in a cluster lies within a radial distance relative to a given center, the shape of the covering or hull of a cluster is hyper-spherical (convex) which sometimes does not encompass adequately the elements that belong to it. For this reason we propose to solve the clustering problem through the optimization of Shannon's Entropy. The optimization of this criterion represents a hard combinatorial problem which disallows the use of traditional optimization techniques, and thus, the use of a very efficient optimization technique is necessary. We consider that Genetic Algorithms are a good alternative. We show that our method allows to obtain successfull results for problems where the clusters have complex spatial arrangements. Such method obtains clusters with non-convex hulls that adequately encompass its elements. We statistically show that our method displays the best performance that can be achieved under the assumption of normal distribution of the elements of the clusters. We also show that this is a good alternative when this assumption is not met.

Keywords: Clustering, Genetic Algorithms, Shannon's Entropy, Bayesian Classifier.

1 Introduction

Pattern recognition is a scientific discipline whose purpose is to describe and classify objects. The descriptive process involves a symbolic representation of

M. Emmerich et al. (eds.), *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation IV*, Advances in Intelligent Systems and Computing 227,
DOI: 10.1007/978-3-319-01128-8_2,

these objects called *patterns.* In this sense, the most common representation is through a numerical vector $\boldsymbol{x}$:

$$\boldsymbol{x} = [x_1, x_2, \ldots x_n] \in \Re^n \tag{1}$$

where the n components represent the value of the properties or attributes of an object. Given a pattern set X, there are two ways to attempt the classification: a) *Supervised Approach* and b) *Unsupervised Approach.*

In the supervised approach, $\forall \boldsymbol{x} \in X$ there is a class label $y \in \{1, 2, 3, ..., k\}$. Given a set of class labels Y corresponding to some observed patterns $\boldsymbol{x}$ ("training" patterns), we may postulate a hypothesis about the structure of X that is usually called the *model.* The model is a mathematical generalization that allows us to divide the space of X into k decision regions called *classes.* Given a model M, the class label y of an unobserved (unclassified) pattern $\boldsymbol{x}'$ is given by:

$$y = M(\boldsymbol{x}') \tag{2}$$

On the other hand, the unsupervised approach consists in finding a hypothesis about the structure of X based only on the similarity relationships among its elements. The unsupervised approach does not use prior class information. The similarity relationships allow to divide the space of X into k subsets called *clusters.* A cluster is a collection of elements of X which are "similar" between them and "dissimilar" to the elements belonging to other clusters. Usually the similarity is defined by a *metric* or distance function $d : X \times X \to \Re$.

In this work we discuss a clustering method which does not depend explicitly on minimizing a distance metric and thus, the shape of the clusters is not constrained by hyper-spherical hulls. Clustering is a search process on the space of X that allows us to find the k clusters that satisfy an optimization criteria. Mathematically, any criterion involves an objective function f which must be optimized. Depending on the type of f, there are several methods to find it. Since our clustering method involves an objective function f where its feasible space is, in general, non-convex and very large, a good optimization algorithm is compulsory. With this in mind, we made a comprehensive study [13] which dealt with the relative performance of a set of structurally different GAs and a non-evolutionary algorithm over a wide set of problems. These results allowed us to select the statistically "best" algorithm: the EGA [20]. By using EGA we may be sure that our method will displays high effectiveness for complex arrangements of X.

The paper is organized as follows: In Section 2, we briefly show the results that led us to select the EGA. Then we present the different sets of patterns X that will serve as a the core for our experiments. We use a Bayesian Classifier[2,4,8] as a method of reference because there is theoretical proof that its is optimal given data stemming from normal distributions. In this section we discuss the issues which support our choice. In Section 3 we discuss the main characteristics of our method and the experiments which show that it is the best alternative. In Section 4 we present our general conclusions.

2 Preliminaries

As pointed out above, a "good" optimization algorithm must be selected. We rest on the conclusions of our previous analysis regarding the performance of a set of GAs[13] . Having selected the best GA, we prove the effectiveness of our clustering method by classifying different pattern sets. To this effect, we generated pattern sets where, for each pattern, the class of the objects is known. Hence, the class found by our clustering method may be compared to the true ones. To make the problems non-trivial we selected a non-linearly separable problems. We discuss the process followed to generate these sets. Finally, we resort to a *Bayesian Classifier* [4] in order to show that the results obtained by our method are similar to those obtained with it.

2.1 Choosing the Best Optimization Algorithm

This section is a very brief summary of the most important results found in [13]. A set A of 4 structurally different GAs and a non-evolutionary algorithm (NEA) was selected in order to solve, in principle, an unlimited supply of systematically generated functions in $\Re \times \Re$(called unbiased functions). An extended set of such functions in $\Re \times \Re^2$ and $\Re \times \Re^3$ was generated and solved. Similar behavior of all the GAs in$A$ (within statistical limits) was found. This fact allowed us to hypothesize that the expected behavior of A for functions in $\Re \times \Re^n$ will be similar. As supplement, we tackled a suite of problems (approximately 50) which includes hard unconstrained problems (which traditionally have been used for benchmarking purposes) [19,3] and constrained problems [11]. Lastly, atypical GA-hard functions were analyzed [18,16].

Set of Algorithms. The set A included the following GAs: a)An elitist canonical GA (in what follows referred to as TGA [eliTist GA]) [21], b) A Cross generational elitist selection, Heterogeneous recombination, and Cataclysmic mutation algorithm (CHC algorithm) [5], c) An Eclectic Genetic Algorithm (EGA) [20], d) A Statistical GA (SGA) [23,12] and e) A non-evolutionary algorithm called RMH [17].

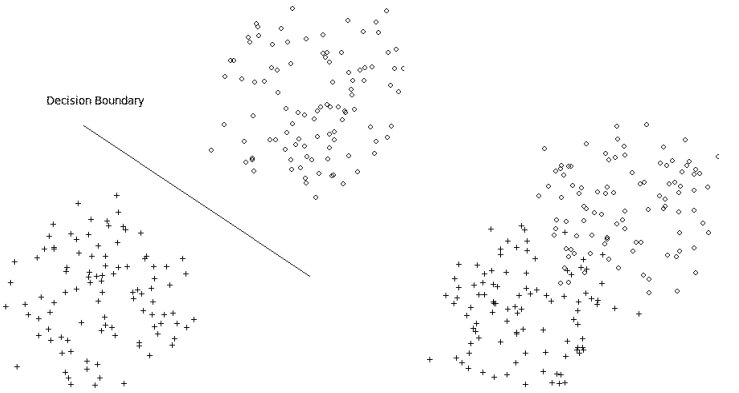
Table 1 shows the relative global performance of all algorithms for the functions mentioned. The best algorithm in the table is EGA.

Table 1. Global Performance

A_i	Unbiased	Suite	Atypical	Global Performance	Relative
EGA	9.64	8.00	4.48	7.37	100.00%
RMH	6.24	0.012	2.04	2.76	37.49%
TGA	1.35	1.16	4.77	2.43	32.91%
SGA	1.33	0.036	3.33	1.57	21.23%
CHC	2.12	0.08	2.10	1.43	19.44%

2.2 The Pattern Set

Given a set of patterns to be classified, the goal of any classification technique is to determine the decision boundary between classes. When these classes are unequivocally separated from each other the problem is separable; otherwise, the problem is non-separable. If the problem is linarly separable, the decision consists of a hyperplane. In Figure 1 we illustrate ths situation.



(a) Linearly Separable patterns (b) Non-linearly separable patterns

Fig. 1. Decision boundary

When there is overlap between classes some classification techniques (e.g. Linear classifiers, Single-Layer Perceptrons [8]) may display unwanted behavior because decision boundaries may be highly irregular . To avoid this problem many techniques has been tried (e.g. Support Vector Machine [9], Multilayer Perceptrons [22]). However, there is no guarantee that any of this methods will perform adequately. Nevertheless, there is a case which allows us to test the appropriateness of our method. Since it has been proven that if the classes are normally distributed, a Bayesian Classifier yields the best possible result (in Section 2.3 we discuss this fact) and the error ratio will be minimized. Thus, the Bayesian Classifier becomes a good method with which to compare any alternative clustering algorithm.

Hence, we generated Gaussian pattern sets considering singular arrangements in which determining the decision boundaries imply non-zero error ratios. Without loss generality we focus on patterns defined in $\Re^2$. We wish to show that the results obtained with our method are close to those obtained with a Bayesian Classifier; in Section 3, the reader will find the generalization of our method for $\Re^n$.

Gaussian Patterns in $\Re^2$. Let X_j be a pattern set defined in $\Re^2$ and $C_i \subset X_j$ a pattern class. A pattern $\boldsymbol{x} = [x_1, x_2] \in C_i$ is drawn from a Gaussian distribution if its joint probability density function (pdf) is given by:

$$f(x_1, x_2) = \frac{1}{2\pi\sigma_{x_1}\sigma_{x_2}\sqrt{1-\rho^2}} e^{\left(-\frac{1}{2(1-\rho^2)}\left[\left(\frac{x_1-\mu_{x_1}}{\sigma_{x_1}}\right)^2 - 2\rho\frac{(x_1-\mu_{x_1})(x_2-\mu_{x_2})}{\sigma_{x_1}\sigma_{x_2}} + \left(\frac{x_2-\mu_{x_2}}{\sigma_{x_2}}\right)^2\right]\right)} \tag{3}$$

where $-1 < \rho < 1$, $-\infty < \mu_{x_1} < \infty$, $-\infty < \mu_{x_2} < \infty$, $\sigma_{x_1} > 0$, $\sigma_{x_2} > 0$. The value ρ is called the correlation coefficient.

To generate a Gaussian pattern $\boldsymbol{x} = [x_1, x_2]$, we use the *acceptance-rejection* method [1,10] which allows us to generate random observations (x_1, x_2) that are drawn from $f(x_1, x_2)$. In this method, a uniformly distributed random point (x_1, x_2, y) is generated and accepted iff $y < f(x_1, x_2)$. In Figure 2.1 we show different pattern sets obtained by applying this method with distinct statistical arguments in (3)

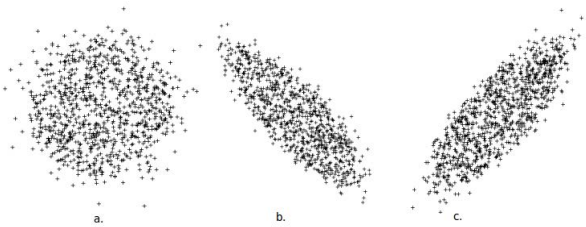


Fig. 2. Different Gaussian pattern sets with $\mu_{x_1} = \mu_{x_2} = 0.5$,$\sigma_{x_1} = \sigma_{x_2} = 0.09$. Each set was generated with different correlation coefficient: a. $\rho = 0.0$, b. $\rho = -0.8$, c. $\rho = 0.8$.

The degrees of freedom in (3) allow us to generate Gaussian pattern sets with varied spatial arrangements. In principle, we analyze pattern sets with the following configurations:

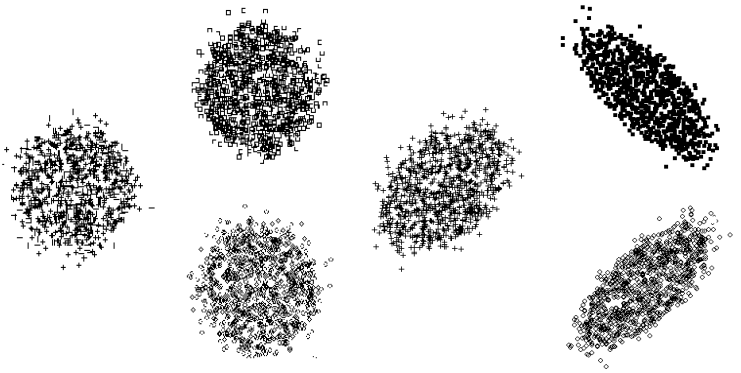
- Sets with disjoint pattern classes.
- Sets with pattern classes that share some elements (partial overlap between classes)
- Sets with pattern classes whose members may share most elements (total overlap).

We proposed these configurations in order to increase gradually the complexity of the clustering problem and analyze systematically the performance of our method. In the following subsections, we make a detailed discussion regarding the generation of sets with these configurations.

Gaussian Pattern Set with Disjoint Classes

Definition: Let X_1 be a pattern set with classes $C_i \subset X_1 \forall i = 1, 2...k$ which are drawn from a Gaussian distribution. X_1 is a set with disjoint classes if $\forall C_i,\ C_j \subset X_1,\ C_i \cap C_j = \phi$.

Based on the above definition, we generate two different sets where $\boldsymbol{x} \in [0,1]^2$ (in every set there are three classes and $|X| = 1000$). In Figure 3 we illustrate the spatial arrangement of these sets.



(a) Pattern set with $\rho = 0$ (b) Pattern set with $\rho = 0.4$,$\rho = -0.7$, $\rho = 0.7$

Fig. 3. Gaussian pattern sets with disjoint classes

Gaussian Pattern Set with Partial Overlap

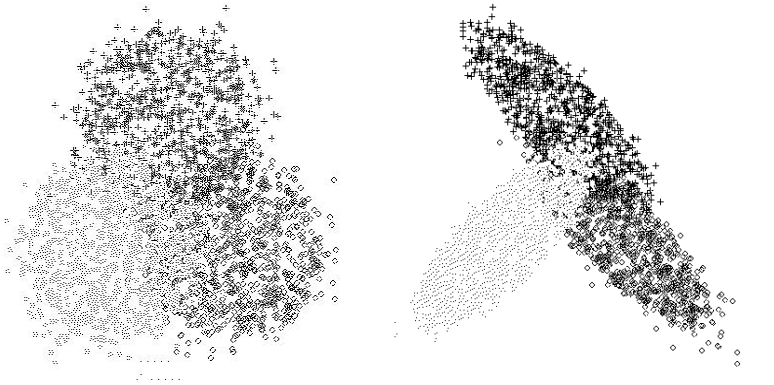
Definition: Let X_2 be a pattern set with classes $C_i \subset X_2 \forall i = 1, 2...k$ which are drawn from a Gaussian distribution. X_2 is a set with partial overlap if $\exists C_i,\ C_j \subset X_2$ such that $C_i \cap C_j \neq \phi$ and $C_i \nsubseteq C_j$.

Based on the above definition, we generate two different sets where $\boldsymbol{x} \in [0,1]^2$ (in every set there are three classes and $|X| = 1000$). In Figure 4 we illustrate the spatial arrangement of these sets.

Gaussian Pattern Set with Total Overlap

Definition: Let X_3 be a pattern set with classes $C_i \subset X_3 \forall i = 1, 2...k$ which are drawn from a Gaussian distribution. X_3 is a set with total overlap if $\exists C_i,\ C_j \subset X_3$ such that $C_i \subseteq C_j$.

Based on the above definition, we generate two different sets where $\boldsymbol{x} \in [0,1]^2$ (in every set there are three classes and $|X| = 1000$). In Figure 5 we illustrate the spatial arrangement of these sets.



(a) Pattern set with $\rho = 0$ (b) Pattern set $\rho = 0.8$, $\rho = -0.8$, $\rho = -0.8$

Fig. 4. Pattern sets with overlap classes

(a) Pattern set with$\rho = 0$ (b) Pattern set with $\rho = -0.9$

Fig. 5. Pattern sets with total overlap classes

2.3 Bayesian Classifier

If the objects in the classes to be clustered are drawn from normally distributed data, the best alternative to determine the decision boundary is using a Bayesian Classifier. The reader can find a extended discussion in [4,8].

Given a sample of labeled patterns $\boldsymbol{x} \in X$, we can hypothesize a partitioning of the space of X into k classes C_i. The classification problem can be reduced to find the probability that given a pattern $\boldsymbol{x}$, it belongs to C_i. From Bayes's theorem this probability is given by:

$$p(C_i|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|C_i)p(C_i)}{p(\boldsymbol{x})} \tag{4}$$

where $p(C_i)$ is usually called the *prior* probability. The term $p(\boldsymbol{x}|C_i)$ represents the *likelihood* that observing the class C_i we can find the pattern $\boldsymbol{x}$. The probability to find a pattern $\boldsymbol{x}$ in X is denoted as $p(\boldsymbol{x})$ which is given by:

$$p(\boldsymbol{x}) = \sum_{i=0}^{k} p(\boldsymbol{x}|C_i)p(C_i) \tag{5}$$

The prior probability $p(C_i)$ is determined by the training pattern set (through the class labels of every pattern). From 4 we can note that the product $p(\boldsymbol{x}|C_i)p(C_i)$ is the most important term to determine $p(C_i|\boldsymbol{x})$ (the value of $p(\boldsymbol{x})$ is merely a scale factor). Therefore, given a pattern $\boldsymbol{x}$ to be classified into two classes C_i or C_j, our decision must focus on determining $max\left[p(\boldsymbol{x}|C_i)p(C_i), p(\boldsymbol{x}|C_j)p(C_j)\right]$. In this sense, if we have a pattern $\boldsymbol{x}$ for which $p(\boldsymbol{x}|C_i)p(C_i) \geq p(\boldsymbol{x}|C_j)p(C_j)$ we will decide that it belongs to C_i, otherwise, we will decide that it belongs to C_j. This rule is called *Bayes's Rule*.

Under gaussian assumption, the Bayesian classifier outperforms other classification techniques, such as those based on *linear predictor functions* [8]. We discuss our method assuming normality so as to measure its performance relative to that of a Bayesian Classifier. Our claim is that, if the method performs satisfactorily when faced with Gaussian patterns, it will also perform reasonably well when faced with other possible distributions.

3 Clustering Based on Shannon's Entropy (CBE)

We can visualize any clustering method as a search for k regions (in the space of the pattern set X), where the *dispersion* between the elements that belong to them is minimized. This dispersion can be optimized via a distance metric [15,7], a quality criterion or a membership function [14]. In this section we discuss an alternative based on *Shannon's Entropy* [24] which appeals to an evaulation of the *information content* of a random variable Y with possible values $\{y_1, y_2, ...y_n\}$. From a statistical viewpoint, the information of the event $(Y = y_i)$ is proportional to its likelihood. Usually this information is denoted by $I(y_i)$ which can be expressed as:

$$I(y_i) = log\left(\frac{1}{p(y_i)}\right) = -log\left(p(y_i)\right) \tag{6}$$

From information theory [24,6], the information content of Y is the expected value of I. This value is called Shannon's Entropy which is given by:

$$H(Y) = -\sum_{i=1}^{n} p(y_i)log\left(p(y_i)\right) \tag{7}$$

When $p(y_i)$ is uniformly distributed, the entropy value of Y is maximal. It means that all events in the probability space of Y have the same ocurrence probability and thus Y has the highest level of unpredictability. In other words, Y has the maximal information content.

In the context of the clustering problem, given an unlabeled pattern set X, we hypothesize that a cluster is a region of the space of X with a large information content. In this sense, the clustering problem is reduced to a search for k regions where the entropy is maximized. Tacitly, this implies finding k optimal probability distributions (those that maximize the information content for each regions). It is a hard combinatorial optimization problem that requires an efficient optimization method. In what follows we discuss the way to solve such problem through EGA. In principle, we show some evidences that allow us to think that our method based on entropy is succesful. We statistically show this method is the best.

3.1 Shannon's Entropy in Clustering Problem

Given an unlabeled pattern set X, we want to find a division of the space of the X into k regions denoted by C_i, where Shannon's Entropy is maximized. We consider that the entropy of C_i depends on the probability distribution of all possible patterns $\boldsymbol{x}$ that belong to it. In this sense, the entropy of C_i can be expressed as:

$$H(C_i) = \sum_{\boldsymbol{x} \in C_i} p(\boldsymbol{x}|C_i) log(p(\boldsymbol{x}|C_i) \tag{8}$$

Since we want to find k regions C_i that maximize such entropy, the problem is reduced to an optimization problem of the form:

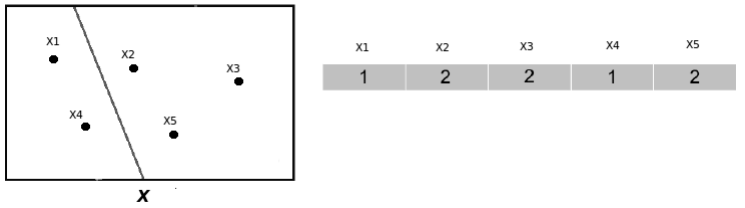
$$\begin{gathered} \text{Maximize: } \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in C_i} p(\boldsymbol{x}|C_i) log(p(\boldsymbol{x}|C_i)) \\ \text{subject to:} \\ p(\boldsymbol{x}|C_i) > 0 \end{gathered} \tag{9}$$

To find the C_i's that minimize (9) we resort to the EGA. We encoded an individual as a random sequence of symbols L from the alphabet $\sum = \{1, 2, 3...k\}$. Every element in L represents the class label of a pattern $\boldsymbol{x}$ in X such that the length of L is $|X|$. Tacitly, this encoding divides the space of X into k regions C_i as is illustrated in Figure 5.1.

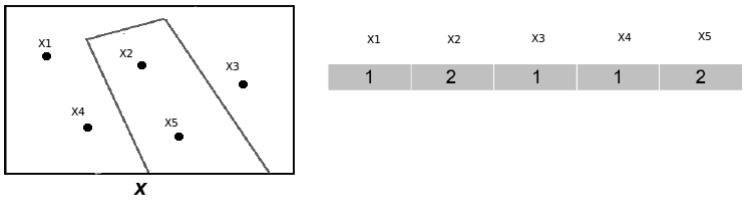
Given this partititon, we can determine some descriptive parameters θ_i of the probability distribution of C_i (e.g. the mean or the standard deviation). Having determined θ_i and under the assumption that the probability distribution of C_i is known, the entropy of C_i can be determined. Complementarily, in Subection 3.3.2 we discuss a generalization making this assumption unnecessary.

3.2 Effectiveness of the Entropic Clustering for Gaussian Patterns in $\Re^2$

Based on the above, given an encoding solution L of a clustering problem (where X is a pattern set in $\Re^2$) we can determine k regions C_i with parameter $\theta_i = [\mu(C_i), \sigma(C_i)]$. Assuming that C_i is drawn from a Gaussian distribution, the value of $p(\boldsymbol{x}|C_i)$ with $\boldsymbol{x} = [x_1, x_2]$ is given by:



(a)



(b)

Fig. 6. Possible divisions of the space of a pattern set X in $\Re^2$ based on the encoding of an individual

$$\int_{x_1}\int_{x_2} f(x_1, x_2) \tag{10}$$

where $f(x_1, x_2)$ is the bivariate Gaussian density function (see Equation 3). Given such probability the entropy for each C_i can be determined, the fitness of every L is given by $\sum_{i=1}^{k} H(C_i)$. The optimal solution will be the individual with the best fitness value after G iterations. To measure the effectiveness of our method, we ran 100 times the EGA (with 150 individuals and $G = 300$), selecting randomly different Gaussian pattern sets (disjoint, partial overlap and total overlap) in $\Re^2$. In Table 2 we show the performance of CBE for these problems. The "performance value" is defined as the success ratio based on the class labels of the patterns known a priori. We also illustrate the performance displayed by the Bayesian Classifier for the same set of problems.

Table 2. Performance of CBE and BC for different Gaussian pattern sets in $\Re^2$

Algorithm	Disjoint	Partial Overlap	Total Overlap	Global	Relative
CBE	99.0	70.8	57.6	75.8	99.7%
BC	99.9	71.7	56.5	76.0	100%

We see that there is not an important difference between results of our method and the results of a Bayesian Classifier. This result allows us to ascertain that our method is as good as the BC. Recall that the BC displays the best possible performance under Gaussian assumption. Furthermore, it is very important to stress that our method is unsupervised and, hence, the pattern set X is unlabeled. CBE allows us to find the optimal value of θ_i for all C_i that maximize the objective function (see Equation 9). These results are promising but we want to show that our method performs successfully, in general, as will be shown in the sequel.

3.3 Comprehensive Effectiveness Analysis

In order to evaluate the general effectivennes of CBE, we generated systematically a set of 500 clustering problems in $\Re^n$ assuming normality. The number of clusters for each problem and the dimensionality were randomly selected (the number of clusters $k \sim U(2, 20)$ and the dimensionality $n \sim U(2, 10)$). Thus, we obtained an unbiased set of problems to solve through CBE and BC. Similarly, we also propose a method to generate systematically a set of clustering problems in $\Re^n$ without *any assumption regarding the pdf* of the patterns to be classified.

Effectiveness for Gaussian Patterns in $\Re^n$. We wrote a computer program that generates Gaussian patterns $\boldsymbol{x} = [x_1, x_2, ...x_n,]$ through the *acceptance-rejection* method [1,10] given a value of n. Here, a uniformly distributed random point $(x_1, x_2, ...x_n, y)$ is generated and accepted iff $y < f(x_1, x_2, ..., x_n)$ where $f(x_1, x_2, ..., x_n)$ is the Gaussian density function with parameters μ and σ. Our program determines randomly the values of $\mu = [\mu_{x_1}, \mu_{x_2}..., \mu_{x_n}]$ and $\sigma = [\sigma_{x_1}, \sigma_{x_2}..., \sigma_{x_n}]$ such that $\mu_{x_i} \in [0, 1]$ and $\sigma_{x_i} \in [0, 1]$. In this way a cluster is a set of Gaussian patterns with the same values of μ and σ. and a clustering problem is a set of such clusters. The cardinality of a cluster is denoted by $|C_i|$ whose value was established as 200. It is important to note that the class label of every generated pattern was recorded in order to determine the performance or effectiveness of a classification process. We obtained a set of 500 different Gaussian clustering problems. To evaluate the performance of any method (CBE or BC) for such problems, we wrote a computer program that executes the following steps:

1. A set of $N = 36$ clustering problems are randomly selected.
2. A effectiveness value y_i is recorded for each problem.
3. For every N problems $\bar{y_i}$ is calculated.
4. Steps 1-3 are repeated until the values $\bar{y_i}$ are approximately normally distributed with parameter μ' and σ' (from *the central limit theorem*).

Dividing the span of means $\bar{y}_i$ in deciles, normality was considered to have been reached when:

1. $\chi^2 \leq 4$ and
2. The ratio of observations in the i-th decil $O_i \geq 0.5 \ \forall i$.

From Chebyshev's Inequality [25] the probability that the performance of an method denoted by τ lies in the interval $[\mu' - \lambda\sigma', \mu' + \lambda\sigma']$ is given by:

$$p(\mu' - \lambda\sigma' \leq \tau \leq \mu' + \lambda\sigma') \geq 1 - \frac{1}{\lambda^2} \tag{11}$$

where λ denotes the number of standard deviations. By setting $\lambda = 3.1623$ we ensure that the values of τ will lie in this interval with probability $p \approx 0.9$. Hence, the largest value of performance found (with $p \approx 0.95$ if we assume a symmetric distribution) by any method (CBE and BC) is $\mu + \lambda\sigma$. The results are shown in Table 3. These results allow us to prove statistically (with significance level of 0.05) that in $\Re^n$our method is as good as the BC under normality assumption.

Table 3. Comparative average success ratio for Gaussian Problems

Algorithm	μ	σ	$\mu + \lambda\sigma$	Relative
CBE	75.53	3.22	85.71	99%
BC	76.01	3.35	86.60	100%

Effectiveness for Non-Gaussian Patterns $\Re^n$. To generate a Non-Gaussian patterns in $\Re^n$we resort to polynomial functions of the form:

$$f(x_1, x_2, ..., x_n) = a_{m1}x_1^m + ... + a_{mn}x_n^m + ... + a_{11}x_1 + a_{1n}x_n \tag{12}$$

Given that such functions have larger degrees of freedom, we can generate many points uniformly distributed in $\Re^n$. As reported in the previous section, we wrote a computer program that allows us to obtain a set of 500 different problems. These problems were generated for random values of n and k with $|C_i| = 200$. As before, a set of tests with $N = 36$ was performed. As before, the number of samples of size N dependend on the distribution reaching normality. The results of this set of experiments is shown in Table 4. These results show that our method outperforms BC with a significant level of 0.05.

Table 4. Comparative average success ratio for non-Gaussian Problems

Algorithm	μ	σ	$\mu + k\sigma$	Relative
CBE	74.23	2.12	80.93	100%
BC	61.76	2.65	70.14	86%

4 Conclusions

The previous analysis, based on the solution of an exhaustive sample set, allows us to reach the following conclusions:

Entropic clustering (CBE) is reachable via an efficient optimization algorithm. In this case, based on previous work by the authors, one is able to take advantage of the proven efficiency of EGA. The particular optimization function (defined in (9)) yields the best average success ratio. We found that CBE is able to find highly irregular clusters in pattern sets with complex arrangements. When compared to BC's performance over Gaussian distributed data sets, CBE and BC have, practically, indistinguishable success ratios. Thus proving that CBE is comparable to the best theoretical option. Here we, again, stress that while BC corresponds to supervised learning whereas CBE does not. The advantage of this characteristic is evident. When compared to BC's performance over non-Gaussian sets CBE, as expected, displayed a much better success ratio. Based on comprehensive analysis (subsection 3.3), the conclusions above have been reached for statistical p values of $O(0.5)$. In other words, the probability of such results to persist on data sets outside our study is better than 0.95. Thus ensuring the reliability of CBE. Clearly above older alternatives.

References

1. Casella, G., Robert, C.P.: Monte carlo statistical methods (1999)
2. De Sa, J.M.: Pattern recognition: concepts, methods, and applications. Springer (2001)
3. Digalakis, J., Margaritis, K.: An experimental study of benchmarking functions for genetic algorithms (2002)
4. Duda, R., Hart, P., Stork, D.: Pattern classification, Section 10, P. 6. John Wiley, New York (2001)
5. Eshelman, L.: The chc adaptive search algorithm. how to have safe search when engaging in nontraditional genetic recombination (1991)
6. Gallager, R.G.: Information theory and reliable communication (1968)
7. Halkidi, M., Batistakis, Y., Vazirgiannis, M.: On clustering validation techniques. Journal of Intelligent Information Systems 17(2), 107–145 (2001)
8. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. (1998)
9. Hsu, C.W., Chang, C.C., Lin, C.J., et al.: A practical guide to support vector classification (2003)
10. Johnson, J.L.: Probability and statistics for computer science. Wiley Online Library (2003)
11. Kim, J.H., Myung, H.: Evolutionary programming techniques for constrained optimization problems (1997)
12. Kuri-Morales, A.: A statistical genetic algorithm (1999)
13. Kuri-Morales, A., Aldana-Bobadilla, E.: A comprehensive comparative study of structurally different genetic algorithms (sent for publication, 2013)
14. Kuri-Morales, A., Aldana-Bobadilla, E.: The Search for Irregularly Shaped Clusters in Data Mining. Kimito, Funatsu and Kiyoshi, Hasegawa (2011)

15. MacQueen, J., et al.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, California, USA, vol. 1, p. 14 (1967)
16. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press (1996)
17. Mitchell, M., Holland, J., Forrest, S.: When Will a Genetic Algorithm Outperform Hill Climbing? In: Advances of Neural Information Processing Systems 6, pp. 51–58. Morgan Kaufmann (1994)
18. Molga, M., Smutnicki, C.: Test functions for optimization needs, pp. 41–42 (2005), http://www.zsd.ict.pwr.wroc.pl/files/docs/functions.pdf (retrieved March 11, 2012)
19. Pohlheim, H.: Geatbx: Genetic and evolutionary algorithm toolbox for use with matlab documentation (2012)
20. Rezaee, J., Hashemi, A., Nilsaz, N., Dezfouli, H.: Analysis of the strategies in heuristic techniques for solving constrained optimisation problems (2012)
21. Rudolph, G.: Convergence Analysis of Canonical Genetic Algorithms. IEEE Transactions on Neural Networks 5(1), 96–101 (1994)
22. Rumelhart, D.E., Hintont, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature 323(6088), 533–536 (1986)
23. Sánchez-Ferrero, G., Arribas, J.: A statistical-genetic algorithm to select the most significant features in mammograms (2007)
24. Shannon, C.E.: A mathematical theory of communication. ACM SIGMOBILE Mobile Computing and Communications Review 5(1), 3–55 (2001)
25. Steliga, K., Szynal, D.: On markov-type inequalities (2010)