Evolutionary Entropic Clustering: a new methodology for data mining

Angel Kuri-Morales[1], Edwin Aldana-Bobadilla[2]
[1] Department of Computation,
Autonomous Institute Technology of Mexico,
Rio Hondo No. 1,
Mexico City, Mexico

[2] Applied Mathematics and Systems Research Institute,
Autonomous University of Mexico,
University City,
Mexico City, Mexico
akuri@itam.mx, ealdana@uxmcc2.iimas.unam.mx

Abstract. The analysis of arbitrary sets of data usually requires the assumption of selected mathematical characteristics of the said data. This fact imposes severe restrictions on the effectiveness of the clustering algorithms. Usually the elements of a cluster are determined by applying some sort of predefined metric. For instance, one of the Minkowsky metrics, the Mahalanobis distance, etc. In every case, however, this fact determines that (in an axis scaled by the units of the metric) the clusters are n-spherical (for some n space). In practice, the forms of the "best" experimentally determined clusters may significantly differ from an n-spherical configuration. In this paper we advance a novel way of determining the inclusion of an arbitrary object into a purported cluster in the data. Basically, our method is based on the assignment of the membership via the experimental measurement of the information contained in every object of the set. The information is seen from the statistical point of view (i.e. Shannon's theory of communication). Since neither the symbols nor the probabilities of such symbols are known, we have to estimate these experimentally. Ultimately this leads to a problem of optimization. The problem is highly non-linear and possibly non-convex. This disallows the use of traditional optimization techniques. Hence, we apply a rugged genetic algorithm (the so-called Vasconcelos GA). We artificially created a set of data with known properties and handed it to our program. With no information on the characteristics of the set we have obtained high efficiency (97% accuracy) on disjoint data for n=3. We conclude by establishing a program which will allow us to extend these results to arbitrary n.

Keywords: clustering, data mining, information theory, genetic algorithms

1 Introduction

Clustering is an unsupervised process that allows the partition of a data set D in k groups or *clusters* in accordance with a similarity criterion. This process is unsupervised because it does not require a priori knowledge about the clusters.

Generally the similarity criterion is a distance metrics based in *Minkowsky Family* [1] which is given by:

$$d_{mk}(P,Q) = \sqrt[p]{\sum_{i=1}^{n} |P_i - Q_i|^p} \qquad \textbf{(1)}$$

where *P* and *Q* are two vectors in an n-dimensional space.
From the geometric point of view, these metrics represent the distance between two points. However, this distance is sometimes not an appropriate measure. For this reason sometimes the clustering methods use statistical metrics such as *Mahalanobis* [2], *Bhattacharyya* [3], or *Hellinger* [4], [5]. These metrics statistically determine the similarity of the probability distribution between random variables *P* and *Q*.

In addition to a similarity criterion, the clustering process requires the specification of the number of clusters. This number frequently depends on the application domain. Hence, it is usually calculated empirically even though there are methodologies which may be applied to this effect [6].

In general a good clustering method must exhibit:

- Possible handling of multidimensional data sets.
- Independence of the application domain.
- Small number of settings.
- Efficient computability.
- Clusters of arbitrary shape.

With respect to last point, the great majority of the clustering methods restrict the shape of the clusters to hyperspherical shapes owing to the use of distance metrics as a similarity criterion. Thus, implicitly the distance between each point inside a cluster and its center is smaller than the radius of an n-dimensional sphere. This is illustrated in Figure 1 for the simplest case where n=2 (and, thus, yields a circle).

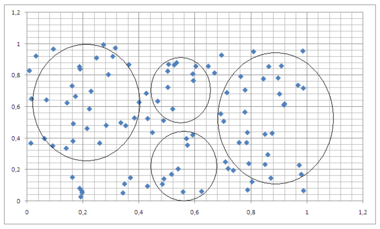


Fig. 1. Clusters with hyperspherical shapes

An ideal case would allow us to obtain arbitrary shapes for the clusters that adequately encompass the data. Figure 2 illustrates this fact.

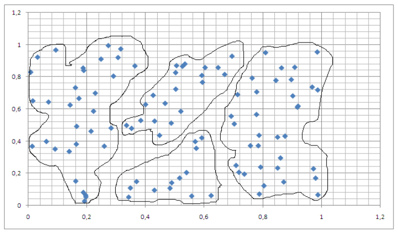


Fig. 2. **Clusters with arbitrary shapes**

Given the above, one of the purposes of this paper is to propose a clustering algorithm that does not depend on distance metrics as a similarity criterion and that allows to find clusters with arbitrary shapes in n-space. This work was preceded by a previous one with similar goals. In that work [7] we successfully tackled irregular clustering in a tridimensional space but the method reported there is not satisfactory for an n-dimensional space owing to its computational complexity. Here we present a proposal based in concepts of information theory and the use of genetic algorithms. We have denominated it "Evolutionary Entropic Clustering".

2 Generalities

2.1 Clustering Methods

The majority of the clustering methods are classified as partitional, hierarchical, density-based and grid-based clustering. In [6] the main algorithms are discussed. Several approaches are possible: for instance: Fuzzy C-Means [8] and Kohonen Maps [9]. The performance of each method depends on the application domain. However, Halkidi presents several approaches that allow to measure the quality of clustering methods via the so-called "quality indices" [6].

2.2 Information Theory

Information theory addresses the problem of collecting and handling of information from a mathematical point of view. There are two approaches: the statistical theory of communication proposed by Claude Shannon [10] and the algorithmic complexity proposed by Andrei Kolmogorov [11] , which may also be found in [12]. In this paper we rely on the statistical approach in which information is a series of symbols that compose a *message,* which is produced by an *information source* and is received by a *receiver* through a *channel*.

Where:

Message: Is a finite succession or sequence of symbols.

Information Source: Is a mathematical model denoted by S which represents an entity which produces a sequence of symbols (message) randomly. The space of all possible symbols is called source alphabet and is denoted as Σ [13].
Receiver: Is the end of the communication's channel which receives the message.
Channel: Is the medium used to convey a *message* from an *information source* to a *receiver*.

In this document we apply two key concepts which are very important for our proposal.
Self Information: It is the information contained in a symbol s_i, which is defined[1] as:

$$I(s_i) = -\log_2 p(s_i) \quad \textbf{(2)}$$

Where $p(s_i)$ is the probability that the symbol s_i is generated by the source *S*. We can see that the information of a symbol is greater when its probability is smaller. Thus, the self information of a sequence of statistically independent symbols is:

$$I(s_1 s_2 \dots s_n) = I(s_1) + I(s_2) + \cdots + I(s_n) \quad \textbf{(3)}$$

Entropy: The entropy is the expected value of the information of the symbols generated by the source *S*. This value may be expressed as:

$$H(S) = \sum_{i=1}^{n} p(s_i) I(s_i) = -\sum_{i=1}^{n} p(s_i) \log_2 p(s_i) \quad \textbf{(4)}$$

Where *n* is the size of the alphabet Σ. Therefore, we see that entropy is greater the more uniform the probability distribution of symbols is.

2.3 Genetic Algorithms

Genetic Algorithms (GA) (a very interesting introduction to genetic algorithms and other evolutionary algorithms may be found in [14]) are optimization algorithms which are frequently cited as “partially simulating the process of natural evolution”. Although this a suggestive analogy behind which, indeed, lies the original motivation for their inception, it is better to understand them as a kind of algorithms which take advantage of the implicit (indeed, unavoidable) granularity of the search space which is induced by the use of the finite binary representation in a digital computer.

In such finite space numbers originally thought of as existing in $\Re^n$ actually map into ***B*** space. Thereafter it is simple to establish that a genetic algorithmic process is a Markov chain (MC) whose states are the populations arising from the so-called genetic *operators*: (typically) selection, crossover and mutation. As such they display all of the properties of a MC. Therefore one may conclude the following mathematical properties of a GA: 1) The results of the evolutionary process are independent of the initial population; 2) A GA preserving the best individual arising during the process will converge to the global optimum (albeit the convergence

[1] The base for the logarithms is arbitrary. When (as above) we choose base 2 the information is measured in "bits".

process is not bounded in time). For a proof of these facts the interested reader may see [15]. Their most outstanding feature is that, as opposed to other more traditional optimization techniques, the GA iterates simultaneously over *several* possible solutions on each iteration of the algorithm. Thereafter, other plausible solutions are obtained by combining (*crossing over*) the *codes* of these solutions to obtain hopefully better ones. The solution space (SS) is, therefore, traversed stochastically searching for increasingly better plausible solutions. In order to guarantee that the SS will be globally explored some bits of the encoded solution are randomly selected and changed (a process called *mutation*). The main concern of GA-practitioners (given the fact that the GAs, in general, will find the best solution) is to make the convergence as efficient as possible. The work of Forrest et al. (see [16]) has determined the characteristics of the so-called *Idealized GA* (IGA) which is impervious to GA-hard problems (see [17]).

2.4 Vasconcelos' Genetic Algorithms

Clearly the implementation of the IGA is unattainable. However, a practical approximation called the Vasconcelos' GA (VGA) has been repeatedly tested and proven to be highly efficient [18]. The VGA, therefore, turns out to be an optimization algorithm of broad scope of application and demonstrably high efficiency.

3 Related Work

In [7] a clustering algorithm in which a data set that belongs to n-dimensional space, is divided in *k* clusters. Each cluster is represented as set of points contained in bodies of arbitrary shape. The shape of these bodies is determined by a formula originally developed by Johan Gielis which has been called the "superformula" and whose mathematical expression in polar coordinates is:

$$r(\phi) = \left[\left| \frac{\cos\left(\frac{m\phi}{4}\right)}{a} \right|^{n_2} + \left| \frac{\sin\left(\frac{m\phi}{4}\right)}{b} \right|^{n_3} \right]^{-\frac{1}{n_1}} \quad \textbf{(5)}$$

It allows the generation of 2-dimensional bodies of arbitrary shape by modifying the parameters *a, b, m, n1, n2, n3*. Figure 3 shows some of the different forms in bi-dimensional space that have been obtained by modifying these parameters.

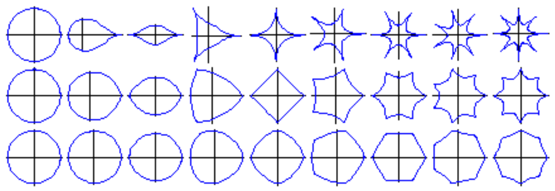


Fig. 3. Some forms generated with the superformula

It may be possible to extend this formula to n-dimensional space, but this problem is still open. We used the Vasconcelos Genetic Algorithm (VGA) in order to obtain the optimal forms of the bodies that encompass all points of the data set. Although the results were satisfactory for a test data set in 3-space, it is computationally intensive and, therefore, not optimal for n-space.

4 Evolutionary Entropic Clustering

4.1 Description

The Evolutionary Entropic Clustering (EEC) is a clustering algorithm in which a data set is enclosed in a *Hypercubic Wrapper*. This wrapper is divided in elements we have called *hypervoxels (*or, for convenience, simply "voxels"*)*. Figure 4 shows a tri-dimensional case.

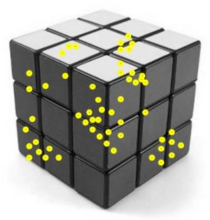


Fig. 4. Hypercubic Wrapper in a tri-dimensional space

Via this wrapper, we use a clustering method based in the concept of *Self Information*, *Entropy*, and *Vasconcelos Genetic Algorithm* (VGA). We start by describing the process to generate the Hypercubic Wrapper. Then, our algorithm based in information theory and genetic algorithms. Finally we annotate en experiment designed to test our initial proposal.

4.2 Hypercubic Wrapper

In order to set experimental probabilities to the elements of the data set, we assign them a position in n-space. We enclose such elements in a finite, bounded n-dimensional cube which we call a Hypercubic Wrapper (HW). The HW is comprised by a set of voxels whose dimensions depend on high finely we decide to discriminate the elements of the data set. Therefore, the number of voxels depends on the Dimensional Resolution Factor (DRF) which represents the number of voxels per dimension. Thereafter, every voxel will represent a symbol in a finite alphabet. This will allow us to calculate the entropy of the data set as will be shown in what follows. In Figure 5 we show a cube with a DRF equal to 3 in all of x, y and z.

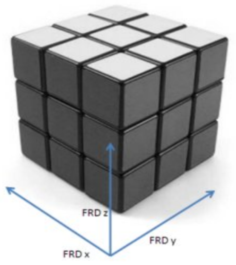


Fig. 5. Dimensional Resolution Factor

However the DRF per dimension value and the length of cube per dimension may be different. This fact allows obtain cubes as shown in Figure 6.

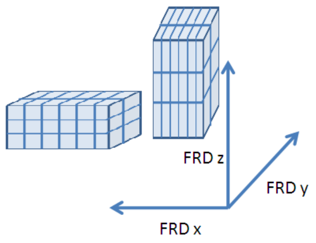


Fig. 6. Hypercubes with different lengths per dimension

Therefore, the dimension length for each hypervoxel is:

$$long_j = \frac{long_j\, Hypercube}{DRF_j} \quad \forall\, HyperVoxel_i \in Hypercube \tag{6}$$

Where $long_j$ is the length of *j-th* dimension for any hypervoxel, $long_j\, Hypercube$ is the length in the *j-th* dimension of hypercube and DRF_j is the dimensional resolution factor in the dimension *j*. We require that the wrapper adapts to the "spatial" distribution of the data.

4.3 Algorithm

We can see that:

- Each data point belongs to one and only one voxel.
- Each voxel is unique in n space.
- Each voxel may have zero or more points.

Then we proceed as follows:

1. Define the number of clusters *k*.
2. Generate the Hypercubic Wrapper.
3. Label each voxel with a natural number according to its position in the hypercube as shown in Figure 7.

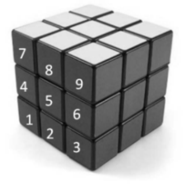


Fig. 7. Hypervoxels Labeled.

4. Count the number of elements of the data set which belong to every voxel as shown in table 1.

Table 1. Number of data per hypervoxel

Hypervoxel	Number of Data
1	**4**
2	**3**
…	**…**
27	**1**
Total ∑	**N**

As stated, we assume that each voxel is a symbol identified by the label assigned previously (see step 3). We may calculate the experimental probability of occurrence of each symbol as the ratio of number of data in the voxel to the total number of elements as illustrated in Table 2.

Table 2. Probability of occurrence per symbol (hypervoxel)

Hypervoxel	Occurrences	P
1	**4**	**4/N**
2	**3**	**3/N**
…	**…**	

27	**1**	**1/N**
Total ∑	**N**	**1.0**

Given this ratio (experimental probability) we may calculate the information and entropy in accordance with equations (3) and (4). Recall that the number *k* of clusters is known a priori. The idea is to construct *k* groups of symbols where the entropy per group is maximized. We must take into account the fact that the sum of the entropy of each group may not exceed the total entropy. Now we maximize the entropy per cluster by running the VGA whose individual has been encoded as follows:

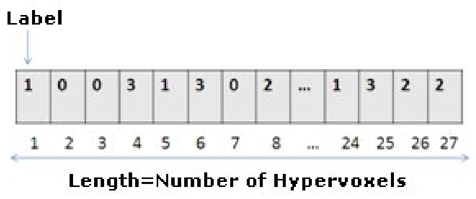


Fig. 8. Genome of the individual in VGA

The length of the individual's genome is equal to number of voxels in the HW. The *i-th* gene expresses the cluster to which the *i-th* voxel belongs. The clusters labeled as 0 correspond to what we call the *null cluster*. This cluster is necessary because not all elements in the HW convey information. The fitness function with which we evaluate each individual is

$$f(individual_j) = max \sum_{i=0}^{k} H_i \quad j \leq N \tag{7}$$

Subject to:

$$\sum_{i=0}^{k} H_i \leq H_{total} \tag{8}$$

$$|Vox_{null}|_{genome} = |Vox_{null}|_{a\ priori} \pm \Delta_a \tag{9}$$

$$|Vox_{not\ null}|_{genome} = |Vox_{not\ null}|_{a\ priori} \pm \Delta_b \tag{10}$$

Where:

H_i = Entropy of cluster i

H_{total} = Total Entropy

$|Vox_{null}|_{genome}$ = Number of empty hipervoxels by the genome

$|Vox_{null}|_{a\ priori}$ = Number of empty hipervoxels in the Hypercube

$|Vox_{nonulls}|_{genome}$ = Number of nonempty hipervoxels proposed by the genome

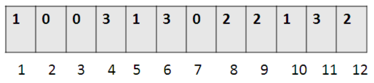
$|Vox_{not\ nulls}|_{a\ priori}$ = Number of nonempty hipervoxels in the Hypercube

Δ_a = Threshold error of number of hypervoxels nulls

$\Delta_b = \text{Threshold error of number of hypervoxels not nulls}$

These values are calculated as follows:

a. H_i: The entropy of *i-th* cluster is calculated summing the values of entropy of all hypervoxels of genome, whose cluster is *i*. This is shown in Figure 10 for a genome with length 12 in which there are 4 clusters (0, 1, 2, and 3).



1	0	0	3	1	3	0	2	2	1	3	2
1	2	3	4	5	6	7	8	9	10	11	12

Fig. 9. Example of an individual genome

If for example we calculate the entropy of cluster 3 would have:

$$H_3 = H_{hipervoxel\ 4} + H_{hipervoxel\ 6} + H_{hipervoxel\ 11}$$

Where $H_{hypervoxel\ j}$ is the entropy of *j-th* hypervoxel whose value we can be calculated with the Table 2.

b. H_{total}: Is the sum of the entropy of all hypervoxels.

c. $|Vox_{null}|_{genome}$: Number of genes whose values are 0 or the number of hypervoxels in the genome that belong to cluster 0. In accordance with the Figure 9 this value is 3.

d. $|Vox_{not\ null}|_{genome}$: Is the number of genes whose value is different from 0. In accordance with the Figure 9 this value is 9.

e. $|Vox_{nulls}|_{a\ priori}$: Is the number of empty hypervoxels from the Table 1.

f. $|Vox_{no\ nulos}|_{a\ priori}$: Is the number of nonempty hypervoxels from Table 1.

g. $\Delta_a$and Δ_b: Configuration parameters whose value is calculated empirically.

The end result of the algorithm is an individual whose genome suggests a grouping of voxels. This is shown in the Figure 10.

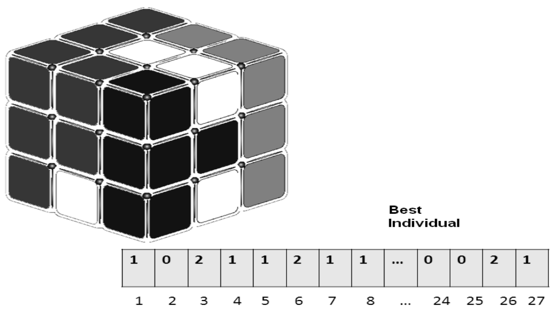


Fig. 10. Possible clustering delivered by the VGA. Different intensities in the cube represent a different clusters. White clusters are null clusters.

5 Experimental Results

The algorithm was tested with a sample of 192 tri-dimensional vectors distributed intentionally in three disjoint spheres as shown in the Figure 11. These data was enclosed in a HW of 512 voxels.

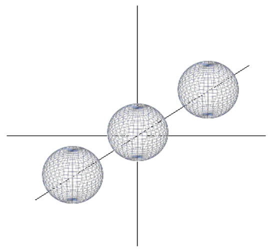


Fig. 11. Distribution of test data

The VGA was run with a population of 100 individuals for 5000 generations. Since the distribution data was known a priori, it was established that the solution of the algorithm is 97% effective. This percentage was obtained by comparing the distribution of data delivered by the algorithm with the original distribution. We found that often the VGA found the optimal individual in close to 1000 generations. This suggests that our proposal is highly efficient. Pending further tests, we believe that this fact owes to the efficiency of the VGA and our approach being based on information theory.

6 Conclusions and Future Work

Computationally, the analysis of the geometric and spatial membership relation between elements of a multidimensional dataset is hard. Our approach showed that in principle, membership relations in a dataset can be found through of its entropy without an excessive demand on computational resources. However the results obtained are initial since they correspond to a particular case and in tri-dimensional space. Therefore, future work requires the analysis of several alternative cases. For instance, one in which the clusters share elements, or that in which the elements belong to concentric spheres. It will also be necessary to generalize our proposal for a dataset in n-dimensional space (n> 3), to analyze its computational complexity and to propose the details of its mathematical formulation.

References

1. Cha, S. H. (2008). Taxonomy of Nominal Type Histogram Distance Measures. *American Conference on Applied Mathematics.* Massachusetts.
2. Mahalanobis, P. C. (1936). On the genaralized distance in statistics.
3. Bhattacharyya, A. (1943). On a measure of divergence between two statistical populations defined by probability distributions. Calcutta.
4. Yang, G. L., & Le Cam, L. M. (2000). Asymptotics in Statistics: Some Basic Concepts. Berlin: Springer.
5. Pollard, D. E. (2002). A user's guide to measure theoretic probability. Cambridge, UK: Cambridge University Press.
6. Halkidi, M., Batistakis, Y., & Vzirgiannis, M. (2001). On Clustering Validation Techniques. *Journal of Intelligent Information Systems* , 107-145.
7. Kuri, A., & Aldana, E. J. (2008). Clustering with an N-dimensional extension of Gielis superformula. *Proceedings of the 7th WSEAS International Conference on Artificial intelligence, knowledge engineering and data bases.* London.
8. Dunn, J. C. (1973). A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. *Journal of Cybernetics* , 32-57.
9. Kohonen, T. (1995). Self-Organizing Maps. Series in Information Sciences. *Series in Information Sciences Vol. 30. Springer* .
10. Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal, Vol 27* , 379-423.
11. Kolmogorov, A. N. (1965). Three approaches to the quantitative definition of information. *Problems on information transmission, Vol. 1* , 1-7.
12. Kuri, A. (2008). Teoria de la Información. En A. Gómez de Silva, *Introducción a la Computación.* Cengage Learning Editores.
13. Gray, R. M. (2008). *Entropy and Information Theory.* Springer Verlag.
14. Bäck, T., *Evolutionary Algorithms in Theory and Practice,* Oxford University Press, 1996.
15. Rudolph, G., *Convergence Analysis of Canonical Genetic Algorithms*, IEEE Transactions on Neural Networks, **5**(1):96-101, January, 1994.
16. Forrest, S. and Mitchell, M., *What makes a problem hard for a genetic algorithm?* Machine Learning, **13**:285-319, 1993.
17. Mitchell, M. *An introduction to genetic algorithms*, MIT Press, pp. 132-134, 1996.
18. Kuri, A., A Methodology for the Statistical Characterization of Genetic Algorithms, LNAI 2313, pp. 79-88, Springer-Verlag, 2002.