Genetic Programming-based Clustering Using an Information Theoretic Fitness Measure

Neven Boric and Pablo A. Estévez

Abstract— **A clustering method based on multitree genetic programming and an information theoretic fitness is proposed. A probabilistic interpretation is given to the output of trees that does not require a conflict resolution phase. The method can cluster data with irregular shapes, estimate the underlying models of the data for each class and use those models to classify unseen patterns. The proposed scheme is tested on several real and artificial data sets, outperforming k-means algorithm in all of them.**

I. Introduction

Clustering consists in partitioning a data set into subgroups such that pairs of patterns belonging to the same group are more *similar* than those belonging to different groups. This definition, as noted by several authors, is circular because the similarity measure defines the optimal clustering [1].

Although some researchers have suggested that "the objective of cluster analysis is simply to find a convenient and valid organization of the data, not to establish rules for separating future data into categories" [2], it is often desired to use the resulting clustering to predict the labels of unseen data, a process denoted in [1] as prediction based on groups.

Given the variety of research fields where clustering is applied, a multitude of approaches for clustering can be found in the literature. Clustering algorithms can be classified as hierarchical and partitional [3], depending on the form of the generated clustering. A hierarchical algorithm produces a nested sequence of partitions, whereas a partitional algorithm produces a single partition. Most partitional clustering algorithms try to minimize the distance between the data and the cluster prototypes, which is equivalent to minimizing the cluster variance. Such algorithms can detect clusters of hyperspherical shape only.

Information theory has been used by several researchers as a basis for clustering, as it provides means to gather higher order statistics in order to reveal the structure of the data [4]–[6]. Gokcay and Principe [4], [7] presented a clustering evaluation function based on Renyi's entropy. Li et. al [5] used k-nearest neighbors to estimate Havrda-Charvat cluster entropy. Both approaches use ad-hoc iterative algorithms for optimization. In [6] the recently developed Cauchy-Schwartz divergence between probability density functions [8] is optimized through gradient descent. None of these methods can be used for clustering unseen data.

Genetic Programming (GP) is an evolutionary technique related to genetic algorithms, where the structures evolved are computer programs, usually represented as trees [9]. GP has proven to be a powerful framework to deal with supervised classification problems. The free nature of GP trees can be exploited to generate programs that separate regions with any kind of shape. Another advantage, shared with other evolutionary techniques, is the capability of avoiding local minima, contrary to gradient descent methods such as backpropagation. GP has been used to solve binary supervised classification problems since its inception [9]. The extension to multiclass problems is not direct, but several approaches have been proposed [10]–[14]. In [10] the multiclass problem is modeled as a set of binary class problems. In [11] the output space of the trees is segmented and each segment is mapped to a class. The best approach so far is the one proposed by Muni et al. [12], [13], where multitree individuals consisting of c trees are evolved for a c class problem. Each tree encodes a rule that is activated when a pattern is said to belong to that class. If multiple rules are activated for the same pattern, a conflict resolution phase is required.

Given the success of GP in supervised classification problems, it seems appropriate to apply GP for clustering. On this matter, the authors are aware of only the work of De Falco et al. [15], which shows the feasibility of using GP for clustering. Their approach is similar to [12] in that it also uses multitree individuals encoding binary rules. Therefore, it requires a conflict resolution phase too. The method allegedly finds the optimal number of clusters by using a variable number of trees per individual. However, instead of specifying c, the user needs to specify another parameter that directly influences the number of clusters found. Cluster variance is used as fitness, constraining clusters to be hyperspherical, and thus, diminishing the main advantage of using GP. Other evolutionary computational techniques applied to clustering include genetic algorithms [16]–[18] and particle swarm optimization [19]. All of them use fitness measures based on cluster variance.

In this paper, a clustering algorithm based on multitree GP is presented. By using an information theoretic fitness measure, the evolved trees are capable of separating clusters of any shape, not only hyperspherical. Unlike other multitree GP approaches that generate a set of binary rules, in our approach the interpretation given to trees' output is probabilistic and thus no conflict resolution phase is required. Compared to purely information theoretic approaches and other evolutionary techniques, the main advantage of the proposed method is that the output of the algorithm is not only the optimal clustering, but also the program (set of

The authors are with Computational Intelligence Laboratory, Department of Electrical Engineering, Universidad de Chile (e-mail: nboric@ing.uchile.cl; pestevez@ing.uchile.cl).

functions) that generates such clustering. As a consequence the evolved programs can easily be applied to cluster unseen data.

II. Genetic Programming

GP is an evolutionary technique to automatically generate computer programs, based on the Darwinian principle of survival of the fittest [9]. The individuals undergoing evolution are computer programs, encoded as hierarchical trees composed of functions and terminals. Each individual is a possible solution to an optimization problem. A GP simulation starts with a population of program trees generated at random. All trees are evaluated and a fitness value is assigned to each one. Trees with better fitness have a higher probability of being selected as parents for the next generation. Genetic operators such as crossover and mutation are applied to the selected trees. The offspring form the next generation of individuals. The process is repeated until satisfying some stopping criterion.

Koza [9] specified five preparatory steps for applying GP, all of which are problem dependent:

- determining the set of terminals,
- determining the set of functions,
- determining the fitness measure,
- determining the parameters and variables for controlling the run, and
- determining the method of designating a result and the criterion for terminating a run.

For applying GP to multicategory classifier design, whether supervised or unsupervised, some further steps are necessary. As the output of a tree is a scalar, it is necessary to adapt the system so that a label is obtained when evaluating an individual. Several possibilities for solving this issue have been explored in the literature, as mentioned in the introduction. In this paper we focus on the multitree approach, as it has shown the best results so far for supervised classification, and extend this approach to clustering. The following sections present in detail our choice for a fitness measure based on information theory and the probabilistic interpretation of the trees' output.

III. Information Theoretic Fitness Measure

A. Background

In pattern recognition, it is often assumed that the patterns corresponding to each of the classes come from a probability density function (pdf) representing the generating model of the data [1], i.e. there exist class conditional pdfs $p\left(x|C_k\right)$, $k \in \{1,\dots,c\}$ such that the samples for each C_k are drawn from these pdfs. Of course, if the pdfs were known, the classification problem would be automatically solved by using Bayes decision rule [1]. Thus many supervised classification methods try to estimate the pdfs from data samples in a way that the class labels are maintained.

In unsupervised classification, the C_k labels are unknown, so usually the problem becomes to determine the optimal labels by minimizing a certain cost function. Most clustering algorithms use a cost function based on second order statistics of the data such as cluster variance.

Information theory provides the foundations to build more complex cost functions that include higher order statistics, and are able to better capture the real structure of the data. The basis of information theory is given by Shannon's entropy. Given a random variable x, its entropy is defined as

$$H\left(x\right) = -\int p\left(x\right) \ln p\left(x\right) dx$$

and it measures the amount of uncertainty present in x or, equivalently, its *randomness*. If the entropy is low, the variable is *less random*. Based on this property, one could minimize the entropy of the patterns belonging to the same cluster, as a lower entropy means that the patterns are more *alike*. The quantity that measures the entropy of all clusters together is the total conditional entropy of x given the clustering C:

$$H\left(x|C\right) = -\sum_{k=1}^{c} P\left(C_k\right) \int p\left(x|C_k\right) \ln p\left(x|C_k\right) dx \quad (1)$$

where $P\left(C_k\right)$ is the *a priori* probability that any given pattern belongs to C_k.

Eq. (1) is minimized when all patterns within each cluster are very similar. Following a similar reasoning, another option is to maximize some kind of inter cluster cross entropy to ensure that patterns belonging to different clusters are dissimilar. One such measure is the Kullback-Leibler divergence between pdfs $p\left(x\right)$ and $q\left(x\right)$:

$$D_{KL}\left(p,q\right) = p\left(x\right) \int \ln \frac{p\left(x\right)}{q\left(x\right)} dx \quad (2)$$

Intuitively, both (1) and (2) seem good candidates for creating a fitness function for GP, as they would drive individuals to cluster together similar patterns and separate dissimilar ones. However, these measures are very difficult to estimate directly from data without imposing assumptions about the pdfs [8], and thus some alternative measures of entropy are needed.

B. Cauchy-Schwartz Divergence

Principe et al. [8] devised a way to estimate entropy directly from data. In this section we briefly review it to justify the choice of our fitness function. Instead of Shannon's entropy, they use Renyi's entropy, given by:

$$H_\alpha\left(x\right) = \frac{1}{1-\alpha} \ln \int p^\alpha\left(x\right) dx \quad (3)$$

Given a data set $X = \{x_1,\dots,x_n\}$, $x_i \in \Re^d$, a Parzen window with a Gaussian kernel can be used to estimate $p\left(x\right)$:

$$\tilde{p}\left(x\right) = \frac{1}{n} \sum_{i=1}^{n} G\left(x - x_i, \sigma^2\right) \quad (4)$$

where

$$G\left(x-x_i,\sigma^2\right)=\frac{1}{\left(2\pi\sigma^2\right)^{\frac{d}{2}}}\exp\left\{-\frac{\|x-x_i\|^2}{2\sigma^2}\right\}$$

The σ^2 parameter is called the kernel size and must be specified *a priori*. Replacing (4) into (3), and using $\alpha=2$, it yields:

$$H_2(x)\approx\\ -\ln\int\frac{1}{n}\sum_{i=1}^{n}G\left(x-x_i,\sigma^2\right)\frac{1}{n}\sum_{j=1}^{n}G\left(x-x_j,\sigma^2\right)dx$$

which can be calculated in exact form by exchanging the integral with the sum operators and applying the convolution theorem for gaussians [8]. The final result is

$$H_2(x)\approx-\ln\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}G_{ij,2\sigma^2}$$

where

$$G_{ij,\sigma^2}\equiv G\left(x_i-x_j,\sigma^2\right)$$

The quantity

$$V(x)=\sum_{i=1}^{n}\sum_{j=1}^{n}G_{ij,2\sigma^2} \tag{5}$$

is called the Information Potential (IP) [8]. Maximizing $V(x)$ is equivalent to minimizing the entropy of x, therefore it could be used to approximate (1) as a cost function for clustering problems.

Principe et al. [8] also proposed divergence measures for comparing two pdfs that can be estimated in the same way as (5). One of these measures is the Cauchy-Schwartz (CS) divergence, defined as:

$$D_{CS}(p,q)=-\ln\frac{\int p(x)\,q(x)\,dx}{\sqrt{\int p^2(x)\,dx\int q^2(x)\,dx}}$$

Jenssen et al. [6] used the CS divergence as a cost function for clustering in the following way. Assume the problem consists of separating the data sets in two clusters, then it makes sense to maximize $D_{CS}(p_1,p_2)$, where $p_1=p(x|C_1)$, $p_2=p(x|C_2)$. Using again the Parzen window to estimate both pdfs, D_{CS} can be rewritten as

$$D_{CS}(p,q)\approx\\ -\ln\frac{\sum_{x_i\in C_1}\sum_{x_j\in C_2}G_{ij,2\sigma^2}}{\sqrt{\sum_{x_i,x_{i'}\in C_1}G_{ii',2\sigma^2}\sum_{x_j,x_{j'}\in C_2}G_{jj',2\sigma^2}}} \tag{6}$$

Notice that the numerator of (6) evaluates the gaussian kernel for pairs of patterns belonging to different clusters. Instead, each summation term in the denominator includes only pairs of patterns that are on the same cluster.

To extend (6) for multiple clusters, a crisp membership function of the form $m(x):\Re^d\to\{0,1\}^c$ is introduced, where every pattern is assigned a vector $m(x)=\{m_k\}^T$, $k\in\{1,\ldots,c\}$. Each component of m indicates if the pattern belongs to a certain cluster:

$$m_k(x)=\begin{cases}1 & x\in C_k\\ 0 & x\notin C_k\end{cases}$$

Using m and omitting the logarithm, the Cauchy-Schwartz cost function is defined as

$$J_{CS}(p_1,\ldots,p_c)=\\ \frac{\frac{1}{2}\sum_{i,j=1}^{n}\left(1-m^T(x_i)\,m(x_j)\right)G_{ij,2\sigma^2}}{\sqrt{\prod_{k=1}^{c}\sum_{i,j=1}^{n}m_k(x_i)\,m_k(x_j)\,G_{ij,2\sigma^2}}} \tag{7}$$

The denominator of (7) consists of the multiplication of the information potentials of the patterns belonging to each cluster. Recalling that $D_{CS}=-\ln(J_{CS})$, minimizing J_{CS} implies minimizing the sum of "within cluster" entropies, exactly as in (1). The numerator, on the other hand forms a "between cluster" cross information potential, and thus minimizing J_{CS} implies maximizing the inter cluster cross entropy, similar to (2).

Expressed in terms of minimizing (7), the clustering problem becomes an optimization problem over the set of $m_k(x)$ functions.

C. Determination of σ

Notice that in the derivation of (6), the kernel size σ for each Parzen estimator was chosen the same. This implies that the pdf estimation will not truthfully represent every underlying pdf, no matter the choice of σ. However, Jenssen et al. suggested to estimate the optimal kernel size for the whole data set by using Silverman's rule of thumb:

$$\sigma_{\text{opt}}=\sigma_X\left(\frac{4}{n(2d+1)}\right)^{\frac{1}{d+4}} \tag{8}$$

where $\sigma_X^2=\frac{1}{d}\sum_{i=1}^{d}\sigma_{ii}^2$, and σ_{ii}^2 are the variances for each dimension of the data. This choice of σ is sufficient for most cases shown bellow.

IV. GP-Based Clustering Algorithm

The proposed approach uses GP to obtain the set of membership functions that minimize (7), which is used as a fitness measure. The aim is to produce clustering solutions that assign highly similar patterns to the same cluster and dissimilar patterns to different clusters without imposing any shape constraints to the resulting partitions. The main drawback of J_{CS} as a fitness measure is that its calculation is $O\left(N^2\right)$.

A. Multitree Representation

Similar to [12], in our system an individual is formed by c trees, each of them associated with one of the clusters that are to be formed. The ith individual of the population is denoted as I_i and each of its trees as T_k^i. The output of I_i is a vector combining the outputs of all its trees, $I_i(x)=\left[T_1^i(x),\ldots,T_c^i(x)\right]^T$. As in most clustering algorithms, the numbers of clusters c is specified beforehand by the user. Fig. 1 shows an example of an individual in our system.

B. Probabilistic Interpretation

The optimization problem described in the previous section has degenerated solutions that must be avoided in order to achieve a proper clustering. For instance, if all the patterns were assigned to a single cluster, the numerator of (7) would be exactly zero, yielding an unwanted optimum. To overcome this problem the membership functions are extended to the continuous range $[0,1]$. If the membership functions are further restricted to sum to one such that

$$\sum_{k=1}^{c} m_k(x_i) = 1, \forall i = 1,\ldots,n$$

they can be regarded as a set of *a posteriori* probabilities $m_k(x) = P(C_k|x)$. Our GP based approach tries to obtain these normalized membership functions. Notice that by using GP, the functions themselves are evolved, instead of just their values for the given data set. Using Bayes theorem the class conditional pdfs $p(x|C_k)$ can be obtained:

$$p(x|C_k) = \frac{P(C_k|x)\,p(x)}{P(C_k)} \qquad (9)$$

These pdfs approximate the real class conditionals only to the extent of minimizing J_{CS}, given the (often limited) availability of data samples.

The output of a GP tree is unrestricted in $\Re$. In order to interpret the trees' output as posterior probabilities, some transformations must be made. First, the logistic sigmoid function is applied

$$\tilde{T}_k(x) = \frac{1}{1+e^{-T_k(x)}}$$

to insure that $\tilde{T}_k(x) \in [0,1]$. Then, $\tilde{T}_k(x)$ is normalized with respect to all the outputs of the trees in the same individual, i.e.

$$P(C_k|x) = \frac{\tilde{T}_k(x)}{\sum_{k'=1}^{c} \tilde{T}_{k'}(x)}$$

In summary, the output of an individual in our approach is a vector

$$I_i(x) = \left[P_i(C_1|x), \ldots, P_i(C_c|x)\right]^T$$

where the components correspond to the trees' output transformed such that they are all in $[0,1]$ and sum to 1.

To decide to which cluster a pattern belongs to, the membership functions must be converted back to crisp values. The Bayes rule is to assign pattern x to class $C_{\hat{k}}$ so that

$$\hat{k} = \underset{k}{\operatorname{argmax}}\, P(C_k|x) \qquad (10)$$

In this way, a label can be assigned to each pattern avoiding a conflict resolution phase.

C. Terminal and Function Sets

The function set used in this work is $F = \left\{+,-,\times,/,(\cdot)^2,\sqrt{\cdot},\sin,\cos\right\}$. The terminal set corresponds to $T = \{\text{feature variables}, R\}$, where R are random constants in $[0,100]$.

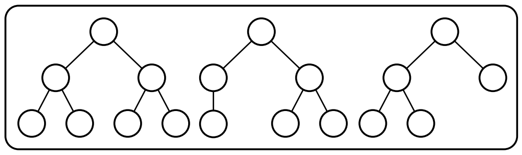


Fig. 1. An example of a multitree individual with $c = 3$

D. Genetic Operators

When using multitree GP, it is necessary to specify new genetic operators that take into account the structure of individuals. Muni et al. [12] introduced an operator that performs crossover between trees that are in the same position in their respective multitree individuals (i.e. for individuals I_{i_1} and I_{i_2}, perform crossover between trees $T_k^{i_1}$ and $T_k^{i_2}$). In this way only trees representing the same class are mixed together. However, this operator loses its appeal for clustering tasks because clusters are indistinguishable between each other during evolution. As the class labels are not available during the GP run, it would be misleading to use a crossover operation that always mixes trees that are in the same position in their respective individuals. In our approach we randomly select one tree from two parents and then perform the usual crossover operation between them (i.e. for individuals I_{i_1} and I_{i_2}, select k_1, k_2 at random and perform crossover between trees $T_{k_1}^{i_1}$ and $T_{k_2}^{i_2}$). This has the disadvantage that the evolution is less directed and more random than in [12]. On the other hand, this approach follows the rationale of the original crossover operation, that randomly selects subtrees to be exchanged without further care for *directing* the evolution.

V. Experimental Results

A. Cluster Evaluation

There exist several ways to measure the performance of a clustering algorithm. If previous knowledge about a data set in the form of class labels is available, the external criterion is applicable [1]. This criterion consists in comparing the partition obtained using the clustering algorithm with the actual partition (existing labels). Several indices have been developed for this purpose, such as the (adjusted) Rand index [20], [21]. If the number of clusters c is equal to the number of predefined classes of the data, it is possible to translate the cluster labels into class labels and calculate a rate of correct classification. A translation process is necessary because it is not possible to know beforehand how clusters and class labels are related to each other. Intuitively, this is solved by associating a cluster with the class of the majority of patterns in that cluster. Although it is usually left unnoticed, most publications implicitly use this interpretation when stating how many patterns were correctly classified or mislabeled by an algorithm. See for instance [16].

B. Experiment Design

To evaluate the performance of the proposed method, three kinds of experiments were carried out. In all of them, the number of clusters c was set equal to the number of actual classes.

1) Clustering: This is a simple clustering experiment. For each data set, the GP is run using all available samples. The resulting membership functions are evaluated on each pattern and a label is assigned using (10). The obtained labels are compared with the actual class labels and a classification score is obtained. The process is repeated 10 times to deal with the random nature of GP. The GP results are compared with those of the popular k-means algorithm [22].

2) Generalization: Here we try to test whether our approach is capable of learning the distribution of each cluster. For this aim, the GP is trained with only a subset of the data and tested on the remaining patterns. A tenfold cross validation procedure was used as follows: first, randomly split the data set into ten subsets. For each GP run, one of the subsets is left out as the validation set. The remaining nine subsets form the training set. The evolution is run using only the training set. The resulting membership functions are evaluated on the validation set and labels are assigned and compared with the actual labels. The process is repeated 10 times (until every subset has been used as a validation set) and the classification scores are averaged.

3) Learning Conditional pdf: On this experiment we intend to show if our approach is capable of estimating the pdf for the data of each class. With this aim, the GP is trained on a two dimensional data set specially designed for graphical evaluation. It consists of four easily separable gaussian clusters, with 100 patterns each. All patterns are used during the training phase. The resulting trees are evaluated on a grid of equidistant points within the range of the data set. This procedure allows us to obtain a graphical representation of the set of *a posteriori* pdfs, $P\left(C_i|x\right)$, obtained by our GP approach. Finally, applying Bayes theorem for every point in the grid, the set of class conditional pdfs $p\left(x|C_i\right)$ for each cluster is obtained. The graphical representation of the class conditionals should show a single gaussian for each cluster.

C. Data Sets

The clustering and generalization tests were performed on two databases commonly used for benchmarking purposes and on an artificial problem suite specially designed for clustering applications. This gives a total of 10 data sets ranging from 2 to 9 dimensions and from 150 to 4096 patterns.

1) Iris: This data set corresponds to the well known Iris data set by Fisher [23]. It contains 150 patterns in four dimensions measured on Iris flowers of three different species. Each class is represented by 50 patterns.

2) Wisconsin Breast Cancer (WBC): It consists of 699 instances of 9 attributes measured on cancer patients. Each instance is labeled either as benign (65.5%) or malignant (34.5%). Sixteen instances were removed as they contain missing values. WBC was obtained from UCI online repository [24].

TABLE I
SUMMARY OF PROPERTIES OF FCPS DATA SETS

Data set	Size	Dimensions	Classes
Hepta	212	3	7
Lsun	400	2	3
Tetra	400	3	4
Chainlink	1000	3	2
Atom	800	3	2
EngyTime	4096	2	2
TwoDiamonds	800	2	2
WingNut	1070	2	2

TABLE II
VALUES OF GP PARAMETERS COMMON TO ALL EXPERIMENTS

Parameter	Value
Max. Generations	200
Population Size	1000
Max. Tree Depth	8
Tournament Size	5
Probability of Crossover	0.85
Probability of Mutation	0.05
Probability of Reproduction	0.1

3) Fundamental Clustering Problem Suite (FCPS): This is a suite of artificial data sets designed by Ultsch [25] that tries to represent all the problems that a clustering algorithm may face. Each data set was designed to test specific scenarios where some clustering algorithms fail. All the data sets are of dimension less than or equal to 3, allowing one to visually assess the performance of a clustering algorithm. Table I shows the main characteristics of 8 data sets taken from this problem suite. Fig. 2 shows a graphical representation of the data sets. FCPS contains two more data sets that were not used in this work, namely "GolfBall" and "Target". The former was not used since it contains no clusters, and the latter because some clusters are formed by only three patterns. See [25] for more details on FCPS.

D. Results

The experiments were performed using GPalta toolbox [26]. Table II shows the value of the parameters used for all GP runs.

The results for the clustering experiment are shown in Table III. It can be seen that the proposed method made a correct clustering of all the data sets except for the "ChainLink" data set, where there is a larger discrepancy between the best and average results. To explain the later result the fitness values for every GP run were analyzed. For all the data sets, the best fitness value matched the best classification score. This confirms that the cost function combined with σ estimation form a good fitness measure, as there are no global optima that lead to undesirable results. Consequently, we believe that the poor convergence on the "ChainLink" data set was caused by the complicated structure of the data, which makes it difficult to generate a set of functions that appropriately partitions it. Nevertheless, in all the data sets, our approach outperformed the popular k-means algorithm.

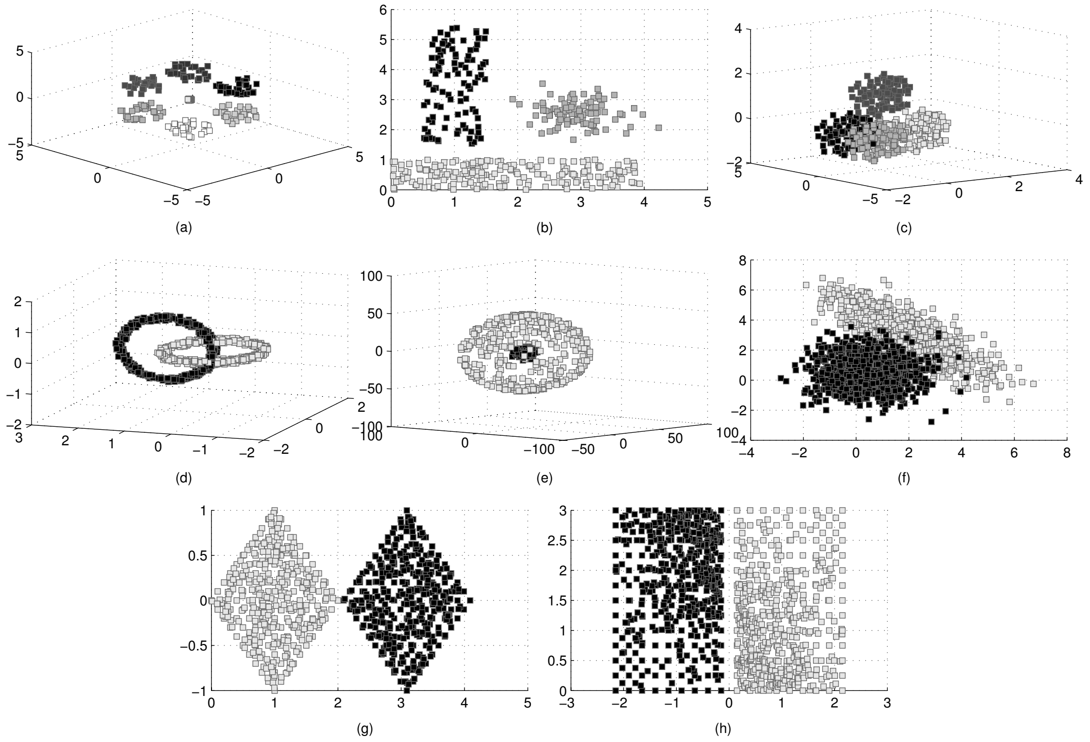


Fig. 2. Fundamental Clustering Problem Suite data sets [25]: (a) Hepta; (b) Lsun; (c) Tetra; (d) Chainlink; (e) Atom; (f) EngyTime; (g) TwoDiamonds; (h) WingNut

TABLE III
CLUSTERING RESULTS USING THE GP BASED APPROACH AND K-MEANS

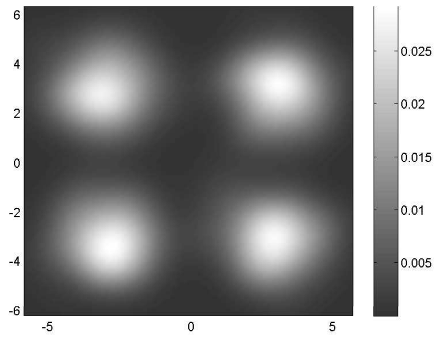
Data sets	GP		k-means	
	Average	Best	Average	Best
Iris	93.00	97.33	89.33	89.33
WBC	96.86	97.21	96.04	96.04
Hepta	100	100	76.09	100
Lsun	99.90	100	76.20	76.50
Tetra	99.40	100	96.40	100
Chainlink	84.68	100	65.42	65.90
Atom	99.58	100	71.53	72.13
EngyTime	95.08	96.46	95.14	95.14
TwoDiamonds	99.89	100	100	100
WingNut	99.21	100	96.36	96.36

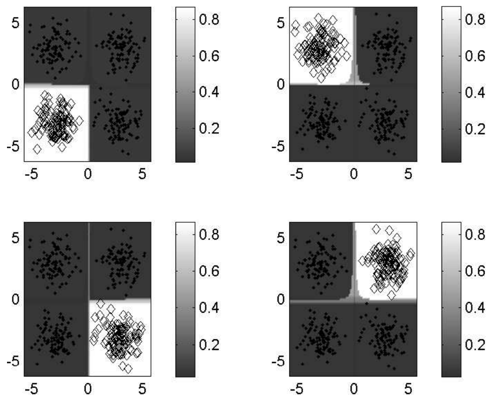
TABLE IV
GENERALIZATION RESULTS USING THE GP APPROACH

Data sets	Classification
Iris	92.67
WBC	90.50
Hepta	98.74
Lsun	92.75
Tetra	97.75
Chainlink	78.30
Atom	99.17
EngyTime	93.69
TwoDiamonds	99.88
WingNut	99.51

The results for the generalization experiment are shown on Table IV. On every case, our method was able to classify the patterns from the validation set with almost the same performance as in the clustering experiment. These results show that the proposed approach, although trained in an unsupervised way, can generalize well and the resulting trees can be easily applied to classify unseen data.

Fig. 3 shows a typical result for the learning conditional pdf experiment. Fig. 3(a) shows the Parzen window estimation of $p(x)$. The estimation yields four gaussians centered on the actual centers for each class, showing that the Parzen method is a good estimator for the pdf of the entire data set. Fig. 3(b) has four subfigures. Each of them shows the normalized output of one of the resulting trees. For a better interpretation, the data set is also represented on each subfigure, with the patterns associated to each cluster emphasized. It can be seen how the trees properly divide the input space. On Fig. 3(c) the output of the trees and the Parzen estimator are combined using (9). It can be observed that our method was capable of separating the four gaussians, providing a good estimation for the class conditional pdfs that clearly resemble the actual distribution of the data for each class.


(a)


(b)

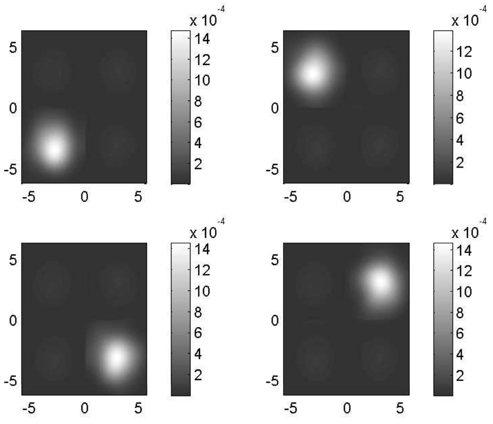

(c)

Fig. 3. Typical results for learning conditional pdf experiment: (a) Parzen window $p(x)$ estimation; (b) $P(C_i|x)$ as given by the normalized trees' output; (c) Estimation of $p(x|C_i)$ obtained using Bayes theorem.

E. Computational Costs

In GP all individuals must be evaluated to get their fitness value in every generation. This could be very costly. The fitness measure used in this paper is $O\left(N^2\right)$, which adds more complexity. However, the method usually finds the best individual within the first 100 generations. Average execution times for a single GP run range from 5 minutes to 5 hours, depending on the size of the data set. In our opinion the benefits of the proposed method outweigh the added complexity, at least for the studied data sets. Stochastic subsampling as described in [6] was explored, but it was found to hinder the performance of the algorithm.

VI. Conclusions

We have extended the recent work on GP applied to classifier design into the field of clustering. By combining GP with an information theoretic fitness measure, our method is able to cluster data without any shape constraints. The probabilistic interpretation given to trees' output eliminates the need for a conflict resolution phase present in other GP applications. Although the same as well as similar cost functions have been optimized by other authors through different means, the main advantage of using GP is the ability to *learn* the membership functions themselves instead of simply partitioning the data. In the generalization experiment, it was shown that these functions can be applied to cluster unseen data with high accuracy. Furthermore, the system was shown to indirectly evolve the generating model for the data, as expressed by the conditional pdfs encoded in GP trees.

Further research efforts should be directed towards improving the convergence of the method and reducing its computational costs. It would also be of interest to perform a comparison between the proposed method and other modern evolutionary approaches, specially when applied on high dimensional data sets.

Acknowledgments

This research was partially supported by FONDECYT 1050751. Neven Boric received a scholarship from CONICYT-CHILE for pursuing graduate studies.

References

[1] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. San Diego, CA, USA: Academic Press, 1999.

[2] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988.

[3] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[4] E. Gokcay and J. Principe, "Information theoretic clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 2, pp. 158–171, Feb. 2002.

[5] H. Li, K. Zhang, and T. Jiang, "Minimum entropy clustering and applications to gene expression analysis," in *Computational Systems Bioinformatics Conference, 2004. CSB 2004. Proceedings. 2004 IEEE*, 16-19 Aug. 2004, pp. 142–151.

[6] R. Jenssen, D. Erdogmus, K. E. Hild II, J. C. Príncipe, and T. Eltoft, "Optimizing the cauchy-schwarz pdf distance for information theoretic, non-parametric clustering." in *EMMCVPR*, 2005, pp. 34–45.

[7] E. Gokcay and J. Principe, "A new clustering evaluation function using renyi's information potential," in *Acoustics, Speech, and Signal Processing, 2000. ICASSP '00. Proceedings. 2000 IEEE International Conference on*, vol. 6, 5-9 June 2000, pp. 3490–3493.

[8] J. Principe, D. Xu, and J. Fisher, “Information theoretic learning,” in *Unsupervised Adaptive Filtering*, S. Haykin, Ed. New York: John Wiley & Sons, 2000.

[9] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992.

[10] J. Kishore, L. Patnaik, V. Mani, and V. Agrawal, “Application of genetic programming for multicategory pattern classification,” *IEEE Trans. Evol. Comput.*, vol. 4, no. 3, pp. 242–258, Sept. 2000.

[11] T. Loveard and V. Ciesielski, “Representing classification problems in genetic programming,” in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 2, 27-30 May 2001, pp. 1070–1077vol.2.

[12] D. Muni, N. Pal, and J. Das, “A novel approach to design classifiers using genetic programming,” *IEEE Trans. Evol. Comput.*, vol. 8, no. 2, pp. 183–196, April 2004.

[13] ——, “Genetic programming for simultaneous feature selection and classifier design,” *IEEE Trans. Syst., Man, Cybern. B*, vol. 36, no. 1, pp. 106–117, Feb. 2006.

[14] L. Cordelia, C. De Stefano, F. Fontanella, and A. Marcelli, “Genetic programming for generating prototypes in classification problems,” in *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, vol. 2, 2-5 Sept. 2005, pp. 1149–1155.

[15] I. D. Falco, E. Tarantino, A. D. Cioppa, and F. Gagliardi, “A novel grammar-based genetic programming approach to clustering,” in *SAC ’05: Proceedings of the 2005 ACM symposium on Applied computing*. New York, NY, USA: ACM Press, 2005, pp. 928–932.

[16] L. Hall, I. Ozyurt, and J. Bezdek, “Clustering with a genetically optimized approach,” *Evolutionary Computation, IEEE Transactions on*, vol. 3, no. 2, pp. 103–112, July 1999.

[17] S. Bandyopadhyay and U. Maulik, “Nonparametric genetic clustering: comparison of validity indices,” *IEEE Trans. Syst., Man, Cybern. C*, vol. 31, no. 1, pp. 120–125, Feb. 2001.

[18] W. Sheng, S. Swift, L. Zhang, and X. Liu, “A weighted sum validity function for clustering with a hybrid niching genetic algorithm,” *IEEE Trans. Syst., Man, Cybern. B*, vol. 35, no. 6, pp. 1156–1167, Dec. 2005.

[19] D. van der Merwe and A. Engelbrecht, “Data clustering using particle swarm optimization,” in *Evolutionary Computation, 2003. CEC ’03. The 2003 Congress on*, vol. 1, 8-12 Dec. 2003, pp. 215–220.

[20] W. M. Rand, “Objective criteria for the evaluation of clustering methods,” *Journal of the American Statistical Association*, vol. 66, pp. 622–626, 1971.

[21] L. Hubert and P. Arabie, “Comparing partitions,” *Journal of Classification*, vol. 2, pp. 193–218, 1985.

[22] J. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Proc. 5th Berkeley Symp. Mathematical Statist. Probability*, 1967, pp. 281–297.

[23] R. A. Fisher, “The use of multiple measurements in taxonomic problems,” *Annals Eugen.*, vol. 7, no. 2, pp. 179–188, 1936.

[24] D. Newman, S. Hettich, C. Blake, and C. Merz, “UCI repository of machine learning databases,” 1998. [Online]. Available: http://www.ics.uci.edu/∼mlearn/MLRepository.html

[25] A. Ultsch, “Clustering with SOM: U*C,” in *Proc. Workshop on Self-Organizing Maps*, Paris, France, 2005, pp. 75–82.

[26] N. Boric, “GPalta genetic programming toolbox,” 2005. [Online]. Available: http://hayabusa.die.uchile.cl/∼nboric/gpalta