Solving Cluster Ensemble Problems by Correlation’s matrix & GA

Dr. Morteza analoui analoui@iust.ir

Niloufar sadighian todaybox@yahoo.ca , n_sadighian@iust.ir

Abstract

Clustering ensembles have emerged as a powerful method for improving both the robustness and the stability of unsupervised classification solutions. However, finding a consensus clustering from multiple partitions is a difficult problem that can be approached from graph-based, combinatorial or statistical perspectives. We offer a probabilistic model of consensus using a finite mixture of multinomial distributions in a space of clustering. A combined partition is found as a solution to the corresponding maximum likelihood problem using the GA algorithm. The excellent scalability of this algorithm and comprehensible underlying model are particularly important for clustering of large datasets. This study includes two sections, at the first, calculate correlation matrix .this matrix show correlation between samples and we found the best samples that can be in the center of clusters. In the other section a genetic algorithm is employed to produce the most stable partitions from an evolving ensemble (population) of clustering algorithms along with a special objective function. The objective function evaluates multiple partitions according to changes caused by data perturbations and prefers those clustering that are least susceptible to those perturbations.

Introduction

Clustering for unsupervised data exploration and analysis has been investigated for decades in the statistics, data mining, and machine learning communities. A recent advance of clustering techniques is the development of cluster ensemble or consensus clustering techniques (Strehl & Ghosh, ۲۰۰۲; Fern & Brodley, ۲۰۰۳;Monti et al., ۲۰۰۳; Topchy et al., ۲۰۰۳), which seek to improve clustering performance by first generating multiple partitions of a given data set and then combining them to form a final (presumably superior) clustering solution. Such techniques have been shown to provide a generic tool for improving the performance of basic clustering algorithms. At the most clustering techniques use similarity attributes between samples for separate samples that can be such as Euclidean distance or ….
There are too much manner for clustering information, for example they are Ants,Isodata, K_means ,Forgy,… A critical problem in this clustering manner is how to adjust initial parameters, for example in Forgy’s algorithm there are two parameters that should adjust:
۱- cluster’s number.
۲- seed point samples .

In this paper we approach this problem by create correlation’s matrix [section ۱] and GA’s algorithm [section ۲].

۱ .correlation matrix

Algorithm :
this algorithm is execute in ۳steps :

۱- *repeat one of the clustering methods like (Ants , ISODATA, K_mean ,Forgy clustering) and create correlation matrix*
۲- *divided correlation matrix into some cluster*
۳- *apply changed Forgy loop*

۱- ۱ repeat one of the clustering methods and create correlation matrix

In this step a clustering algorithm such as (Ants , ISODATA, K_mean , Forgy clustering) execute for some iterations . Suppose we know cluster’s number, then we can use Fforgy’s algorithm .
Forgy’s algorithm :

۱. Initialize the cluster centroids to the seed points.

۲. For each sample, find the cluster centroid nearest it. Put the sample in the cluster identified with this nearest cluster centroid.

۳. If no samples changed clusters in step ۲, stop.

۴. Compute the centroids of the resulting clusters and go to step ۲.

There are ۲ way for iteration’s time*:*

(۱) Iteration=samples/clusters

$$(۲)\quad Iteration = \binom{Samples}{Clusters}$$

Correlation's matrix is n*n matrix .
n is sample's number.
The value of C[I,j] shows how many time sample I and sample J are at one cluster. This matrix is symmetric and the manner of update this matrix shows flow:

```
For I=۰ to samples-۱
  For J=۰ to samples-۱
    If samplecluster[I]=samplecluster[J] then
          Correlation[I,J]=correlation[I,J]+۱
    End if
  Next J
Next I
```

۱-۲ Dividing correlation's matrix into some clusters

There is one correlation's matrix that shows relevant between samples . We should divide these samples into k clusters .First of all we choose k samples as seed points . We could choose these k samples from the correlation's matrix that have minimum relevant and correlation to each others. May be according to this we got some noisy seed points values . I suggest new solution for this problem : choose a sample as seed point , that p samples in all of points have correlation with it at least ۷۰٪ of iterations .

$$p = \left(\frac{samples}{۲ * clusters}\right)$$

There are (*samples/clusters*) Sample in each cluster. Algorithm for choose seed points is written follow:

```
۱-Limit = ۱
۲-find < samples I,J > where Correlation
                                  [I][J] < Limit

۳-for all samples if number of samples that
Correlation[I][samples] >(۷۰٪ Iteration )
                     is greater than p
                        seedpoints ← I
  If (seed<clusters)    then
        For all samples if number of samples
          that Correlation[J][samples] > ۷۰٪
                                   Iteration
                           is greater than p
                              seedpoints ← J
    If (seed<clusters)    then
                    Limit++ and go to step ۲
    Else
      End the Algorithm
```

After we found seed points, we choose $\left(\frac{samples}{clusters}\right)$ points for each cluster that have most Correlation with them.

۲. Clustering by genetic algorithm

In this section we present a scheme driven by evolutionary computation to overcome the problem of comparing clustering results. The clustering results are achieved by qualitatively different clustering algorithms, which produce different partitioning. Our scheme helps us to overcome these problems of algorithms by generating clustering ones and selecting the best evaluated to evolve in another generation until the whole procedure reaches a robust result.
The proposed scheme generates clustering-bitstrings by clustering the *l* modifications with all clustering algorithm in use. Then these clustering-bitstrings are evaluated, compared through a fitness function. The best ones continue their evolution to the next generation.
The final clustering bitstrings have evolved towards the stable schemata, which provide us a robust partition of the data points in the set.

۲-۱ Search of Schemata

Evolutionary algorithms are a family of computer models based on the mechanics of natural selection and natural genetics. Among them are genetic algorithms (GA) [۲۳] and genetic programming (GP) [۲۴]. Genetic algorithms were introduced and investigated by John Holland [۲۳]. Later, they became popular by the book of David Goldberg [۲۵].

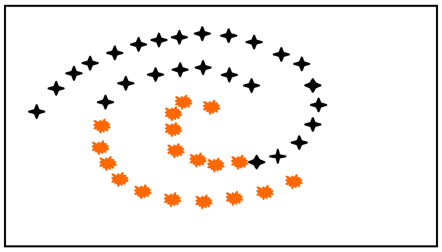

Figure ۱: Intertwined spirals clustered by the average linkage algorithm.

Also, consider the GA tutorial of David Whitley [۲۶] as a very good introduction to the field. GAs and GPs are typically used for optimization problems. An optimization problem is given by a mapping $F : X \rightarrow Y$.

The task is to find an element $x\in X$ for which $y = f(x)$; $y\in Y$ is optimal in some sense. Genetic algorithms encodes a potential solution on a simple chromosome-like data structure, and apply genetic operators such as crossover or mutation to these structures. Then, the potential solution is decoded to the value x in the search space X, and $y = f(x)$ is computed.
The obtained value y is considered as a quality measure, i.e. the fitness for this data structure. Some genetic operators, such as the mating selection, are under control of these fitness values, some other, like the mutation, are not related to fitness at all. An implementation of a GA begins with a population of "chromosomes" (generation ۱). For standard GA, each chromosome (also referred to as individual) is represented as a bitstring of a fixed length (e.g. `0101101` as a bitstring of length ۷). Then, the genetic operators are applied onto all bitstrings iteratively in a fixed order, going from one generation to the next until a given goal (e.g. fitness value exceeds a given threshold or a predefined number of generations was completed) is met. Finally, the individual (or chromosome) with the best fitness value in the final generation is taken as the evolved solution of the optimization problem .At first, ۲m bitstrings are selected out of the k individuals of generation n for mating. Usually, this is done by fitness-proportionate selection , i.e., the relative probability for an individual to be selected is proportional to its fitness value. The better the fitness, the better is the chance to spread out its "genetic material" (i.e., some of its bits) over the next generation .Once the ۲m individuals are chosen, they are paired. In the two bitstrings of each pair, a common splitting point is randomly selected, and a new bitstring is constructed by combining a half of the first bitstring with the other half of the other bitstring. Then, the new individuals are mutated, i.e. some of its bits are reversed with a given (usually small) probability. This gives the so-called m children of parent generation n. Now, the fitness values of the children are evaluated by decoding them into x values and computing the $f(x)$.
Some of the children might have a better fitness than its parents. From the k individuals of generation n and the m children, the best k individuals constitute the next generation (n+۱). While randomized, GAs are no simple random walks. For the standard GA, John Holland has derived the well-known Schemata Theorem, which models a GA by means of the so-called schematas (or similarity templates). A schema is an incomplete bitstring in the sense that it contains unspecified bits. An example for a schema is `10*110`, which leaves position ۳ unspecified. `101110` is a realization of this schema. Generation n contains each possible schema to some extent. It can be said, that such a schema is tested by the GA, or that trials are allocated to it by the GA. Now, one measure for a schema is the average fitness of all of its realizations. A second measure is the ratio of this average to the "average average" of all schemata present in the generation n, i.e. its above-averageness . The Schemata Theorem relates the rate of a schema within a population with this measure. It says, that the rate of a schema within a population grows exponentially with its above averageness. The most important point here is that all schematas are tested in parallel.
Strongly related to the application of a GA is the encoding problem. In general, GAs are applied to highly non-linear, complex problems, where it is hard to find a model which provides an approach to the solution. In these applications, they are the most simple approach. However, a GA is not guaranteed to find the global optimum of a problem. It only ensures, by the Schemata Theorem, to find better solutions than the random initialized ones. GAs find evolved solutions.

۲-۲ Fitness Function

The fitness function $y = f(x)$; $x\in X$ in search, has to keep track of the difference of the tested indiviual compared to all other (original) individuals. Because all individuals are given as bitstrings, the Hamming distance, which keeps track of inverted bitstrings will be the right measure.

So we define the fitness function $y(b)$ as follows:

$$y(b) = \frac{1}{m}\sum_{i=1}^{m}\min\left[\sum_{j=1}^{n}|x_{i,j}-b_j|, \sum_{j=1}^{n}|(1-x_{i,j})-b_j|\right]$$

where:
n= length of the Bitstring
m= number of orginals (clusterstrings)
$x_g \in X$
so that:
x_{gv} is the vth bit of gth original clustering-string
The alternating measure of the fitness function reflects the issue that different clustering approaches may decide differently for assigning class ۱ or ۲. Hence, the more suitable schema should be more similar to either the cluster string or its inverted form.

۳. Experimental Results

We started our experiment so that ۲۸ original clustering strings with a length of ۱۰۰ bit were computed. As parameterization of the Genetic Algorithm we decided to chose :

- as the stop criteria ۳۰۰ generations
- ۳۰ parents in each generation
- ۷۰ children in each generation
- uniform crossover with p.=۰,۵
- mutation:
    - probability = ۰.۷۵
    - mutation methods: swap mutation
- selection metode : Tournament selection with Tournament size=۲

The highest fitness ۱/$y(b)$ we achieved was ۱/$y(b_{best})$ = ۰:۰۲۱۵۵۶. This bit string b_{best} was not equal to any original clustering-string, but had a very low distance (Hamming distance) to the first of the originals.
That's why we decided to name the clustering bit string with The minimal Hamming distance compared

with the individual reaching the highest fitness (b_{best}) where $y(b_{best})$ = min(y(b))^b ۲ X) the result of the procedure. The first original clustering-string, which represents, according to our results, the most appropriate clustering, is the one shown in figure ۱, which shows obviously the correct partitioning of the intertwined spirals problem. After some more trials we got even more results leading again to the first original clustering bit string with even higher fitnesses , i.e. ۱/y(b_{best}) =۰.۰۴۰۹۶ with only four different bits inside the compared strings.

۴. Conclusions

In this paper we have presented a scheme driven by evolutionary computation to overcome the problem of comparing results achieved by qualitatively different clustering algorithms. We have produced a couple of noisy copies of a given two-class clustering problem. Because it was a two-class problems, which means that we were clustering all data into two clusters, the clustering results could be represented as binary bit strings , so they were compatible to the format genetic algorithms work on. Taking the whole lot of clustering results as input to the genetic algorithm we assumed to let it find the scheme of the right clustering for the presented problem.
At the end we achieved reproducible result, for many runs of the genetic algorithm were leading to the same original clustering-bit string which of course points to the most appropriate clustering algorithm. It is, according to our results, possible to find the appropriate clustering for a given problem, but it is also possible to identify the most suitable clustering algorithm for an unknown dataset. Last but not least it seems to be feasible, to classify clustering problems in comparison to their appropriate clustering algorithm.

References

[۱] Peng-Yeng Yin and Ling-Hwei Chen. A new non iterative approach for clustering. *Pattern Recognition Letters*, ۱۵:۱۲۵–۱۳۳, ۱۹۹۴.

[۲] D. Chaudhuri, B. B. Chaudhuri, and C. A.Murthy. A new split-and-merge clustering technique. *Pattern Recognition Letters*, ۱۳:۳۹۹–۴۰۹, ۱۹۹۲.

[۳] Torbjorn Eltoft and Rui J. P. DeFigueiredo. A new neural network for cluster-detection-and labeling. *IEEE Transactions on Neural Networks*, ۹(۵):۱۰۲۱–۱۰۳۴, September ۱۹۹۸.

[۴] C. L. Begovich and V. E. Kane. Estimating the number of groups and group membership using simulation cluster analysis. *Pattern Recognition*, ۱۵(۴):۳۳۵–۳۴۲, ۱۹۸۲.

[۵] Donald E. Brown, Christopher L. Huntley, and Paul J. Garvey. Clustering of homogeneous subsets. *Pattern Recognition Letters*, ۱۲:۴۰۱–۴۰۸, ۱۹۹۱.

[۶] Lei Xu, Adam Krzyzak, and Erkki Oja. Rival penalized competitive learning for clustering analysis, rbf net, and curve detection. *IEEE Transactions on Neural Networks*, ۴(۴):۶۳۶–۶۴۹, ۱۹۹۳.

[۷] Lei Xu. Bayesian ying-yang machine, clustering and number of clusters. *Pattern Recognition Letters*, ۱۸:۱۱۶۷–۱۱۷۸, ۱۹۹۷.

[۸] G. Celeux and G. Soromenho. An entropy criterion for assessing the number of clusters in amixture model. *Journal of Classification*, ۱۳:۱۹۵–۲۱۲, ۱۹۹۶.

[۹] Christophe Biernacki, Gilles Celeux, and Gerard Govaert. Assessing a mixture model for clustering with the integrated classification likelihood. Rapport de recherche ۳۵۲۱, Theme ۴, Unite de recherche INRIA Lorraine, http://www.inria.fr, ۱۹۹۷.

[۱۰] Richard S. Wallace and Takeo Kanade. Finding natural clusters having minimum description length. In *Proceedings of the ۱۰th International Conference on Pattern Recognition*, volume ۱, pages ۴۳۸–۴۴۲, Los Alamitos, CA, USA, ۱۹۹۰. IEEE Comput. S. Press.

[۱۱] A. Marazzi, P. Gamba, A. Mecocci, and A. Semboloni. Automatic selection of the number of clusters in multidimensional data problems. In *Proceedings of the International Conference on Image Processing*, volume ۳, pages ۶۳۱–۶۳۴, NY, USA, ۱۹۹۶. IEEE.

[۱۲] Shri Kant, T. L. Rao, and P. N. Sundaram. An automatic and stable clustering algorithm. *Pattern Recognotion Letters*, ۱۵:۵۴۳–۵۴۹, ۱۹۹۴.
[۱۳] G. H. Ball and D. J. Hall. Isodata, a novel method of data analysis and pattern classification. Technical report, Stanford Research Institute, Menlo Park, ۱۹۶۵.

[۱۴] G. Carpenter and S. Grossberg. Adaptive resonance theory: Stable selforganization of neural recognition codes in response to arbitrary lists of input patterns. In *Proc. ۸th Annu. Conf. Cognitive Sci. Soc.*, pages ۴۵–۶۲, ۱۹۸۶.

[۱۵] R. J. P. DeFigueiredo. The oi, os, omni and osman networks as best approximations of nonlinear systems under training data constraints. In *Proc. IEEE Int. Symp. Circuits Syst.*, Seattle, WA, ۱۹۹۶.

[۱۶] Yoseph Linde, Andrs Buzo, and Robert M. Gray. An algorithm for vector quantizer design. COM- ۲۸(۱):۸۴–۹۵, January ۱۹۸۰.

[۱۷] Thomas M. Martinetz, Stanislav G.Berkovich, and Klaus J. Schulten. “Neural-Gas” network for vector quantization and its application to timeseries prediction. ۴(۴):۵۵۸–۵۶۹, July ۱۹۹۳.

[۱۸] A. Weingessel E. Dimitriadou and K. Hornik. A voting-merging clustering algorithm. In Fuzzy-Neuro Systems ’۹۹, editor, *SFB ”Adaptive Information Systems and Modeling in Economics and Management Science*, number Working Paper ۳۱،۱۹۹۹.

[۱۹] A. Weingessel E. Dimitriadou and K. Hornik. Fuzzy voting in clustering. In Fuzzy-Neuro Systems ’۹۹, editor, *G. Brewka, R. Der S. Gottwald and A. Schierwagen*, pages ۶۳–۷۴, ۱۹۹۹.

[۲۰] Wolfgang von der Gablentz and Mario K¨oppen. agglomerative single-linkage clustering and its capability for solving the interwtined spirals problem, a technical report. experimental results, Fraunhofer IPK, http://www.ipk.fhg.de, ۲۰۰۰.

[۲۱] Marvin L. Minsky and Seymour A. Papert. *Perceptrons – Expanded Edition*. MIT Press, ۱۹۸۸.

[۲۲] Cran. http://cran.at.r-projects.org.

[۲۳] J. A. Holland. *Adaptation in natural and artificial systems*. MIT Press, Cambridge,MA, ۱۹۷۵.

[۲۴] J. Koza. *Genetic programming — On the programming of computers by means of natural selection*. MIT Press, Cambridge, MA, ۱۹۹۲.

[۲۵] D. E. Goldberg. *Genetic algorithms in search, optimization & machine learning*. Addison- Wesley,Reading, MA, ۱۹۸۹.

[۲۶] D.Whitley. A genetic algorithmtutorial. In *Statistics and Computing*, ۴, pages ۶۵–۸۵, ۱۹۹۴.

[۲۷] R. G. Reynolds. An introduction tu cultural algorithm. In *In Proceedings of Evolutionary Programming*, EP-۹۴. San Diego. CA, ۱۹۹۴.