Unsupervised Gene Selection and Clustering using Simulated Annealing

Maurizio Filippone[1], Francesco Masulli[3], and Stefano Rovetta[1]

[1] Dipartimento di Informatica e Scienze dell'Informazione, Università di Genova
Via Dodecaneso 35, I-16146 Genova, Italy
[2] Dipartimento di Informatica, Università di Pisa, Largo B. Pontecorvo 3, I-56127 Pisa, Italy

Abstract. When applied to genomic data, many popular unsupervised explorative data analysis tools based on clustering algorithms often fail due to their small cardinality and high dimensionality. In this paper we propose a wrapper method for gene selection based on simulated annealing and unsupervised clustering. The proposed approach, even if computationally intensive, permits to select the most relevant features (genes), and to rank their relevance, allowing to improve the results of clustering algorithms.

1 Introduction

Unsupervised explorative data analysis using clustering algorithms provide an useful tool to explore data. In the case of genomic data, that are often characterized by small cardinality and high dimensionality (e.g., in the case of gene expression data obtained from DNA microarrays) this approach can fail, as many clustering algorithms suffer from being applied in high-dimensional spaces (each dimension or feature corresponding in our case to a gene expression data), as clustering algorithms often seek for areas where data is especially dense. Moreover, some (or most) genes are not relevant for the clustering learning task and a gene (feature) selection procedure could highlight the relevant genes and improve the clustering results at the same time [11, 14, 2].

Feature selection algorithms can be broadly divided into two categories [3, 10]: filters and wrappers. Filters evaluate the relevance of each feature (subset) using the data set alone, while wrappers invoke the learning algorithm to evaluate the quality of each feature (subset). Both approaches, filters and wrappers, usually involve combinatorial searches through the space of possible feature subsets. Anyway, wrappers are usually more computationally demanding, but they can be superior in accuracy when compared with filters.

Most of the literature on feature selection pertains to supervised learning, and not much work has been done for feature selection in unsupervised learning [13, 6, 11, 8, 14, 2].

In this paper we propose a wrapper approach to gene selection in clustering of gene expression data. The combinatorial search is performed using the Simulated Annealing (SA) method [9] which is a global search method technique derived from Statistical Mechanics and based on the Metropolis algorithm [12], while the learning algorithm is the Fuzzy C-Means (FCM) that is one of most popular clustering algorithms (for a detailed description of the FCM see [1]).

In the next section we describe the proposed SA algorithm for gene selection. In Sect. 3 an evaluation index of gene relevance is presented. Sect. 4 describes the experimental validation of our method on the data set by Golub et al. [7]. Conclusions are presented in Sect. 5.

2 SA for gene selection

The method for feature selection we propose makes use of Simulated Annealing (SA) technique [9] that is a global search method technique derived by Statistical Mechanics.

SA is based on the Metropolis algorithm [12] that has been proposed to simulate the behavior and small fluctuations of a system of atoms starting from an initial configuration, by the generation of a sequence of iterations. In the Metropolis algorithm each iteration is composed by a random perturbation of the actual configuration and the computation of the corresponding energy variation (ΔE). If $\Delta E < 0$ the transition is unconditionally accepted, otherwise the transition is accepted with probability given by the Boltzmann distribution:

$$P(\Delta E) = exp\left(\frac{-\Delta E}{KT}\right) \tag{1}$$

where K is the Boltzmann constant and T the temperature.

In SA this approach is generalized to the solution of general optimization problems [9] by using an *ad hoc* selected cost function (*generalized energy*), instead of the physical energy. SA works as a probabilistic hill-climbing procedure searching for the global optimum of the cost function. The temperature T takes the role of a control parameter of the search area (while K is usually set to 1), and is gradually lowered until no further improvements of the cost function are noticed. SA can work in very high-dimensional searches, given enough computational resources.

In Tab. 1, a detailed description of the proposed Simulated Annealing Feature Selection (SAFS) algorithm is presented.

The system state (configuration) is represented by a binary mask $\mathbf{g} = \left(g_1, g_2, \ldots, g_q\right)$, where each bit g_i (with $i = 1, \ldots, q$) corresponds to the selection ($g_i = 1$) / deselection ($g_i = 0$) of a feature (if we want to select a set of s features, at each time only s bits will be set to 1). The initialization of the vector mask $\mathbf{g}$ (Step 2) is done by generating s integer numbers with uniform distribution in the interval $[1, q]$ and setting the corresponding bits to 1 of $\mathbf{g}$ and the remaining ones to 0. A perturbation or move (Step 5c) is done by switching $2 \times r$ bits of $\mathbf{g}$, by randomly selecting r bits set to 0 and r bits set to "1", and flipping their values.

The unsupervised clustering (Steps 3 and 5d) is performed in the sub-space of selected features defined by the vector mask $\mathbf{g}$. After each run of the unsupervised clustering algorithm we can obtain an evaluation of E as a function of either the cost function associated to the clustering algorithm, clustering validation indexes [4], or, when the data set is labeled, the *Representation Error* (RE) defined as the percentage of data points belonging to the same cluster sharing the same label.

The initial value of temperature T is obtained as the average value of ΔE computed over an assigned number p of random perturbations of the mask $\mathbf{g}$.

Table 1. Simulated Annealing Feature Selection (SAFS) Algorithm.

1. Assign s (number of features to be selected), r (number of bits to be switched for making a move), T (initial temperature of the system), α (cooling parameter), f_{max} (maximum number of iteration at each T), h_{min} (minimum number of success for each T), c (number of clusters), m (fuzzification parameter);
2. Initialize **g** at random (binary mask);
3. Perform unsupervised clustering and evaluate the generalized system energy E;
4. **do**
5. Initialize $f = 0$ (number of iterations), h=0 (number of success);
   (a) **do**
   (b) Increment number of iterations f;
   (c) Perturb mask **g**;
   (d) Perform unsupervised clustering and evaluate the generalized system energy E;
   (e) Generate a random number rnd in the interval [0,1];
   (f) **if** $rnd < P(\Delta E)$ **then**
      i. Accept the new **g** mask;
      ii. Increment number of success h;
   (g) **endif**
   (h) **loop until** $h \leq h_{min}$ **and** $f \leq f_{max}$;
6. update $T = \alpha T$;
7. **loop until** $h > 0$;

SAFS is a very computational intensive algorithm, but it is able to work with every kind of features (e.g., continuous, ordinal, binary, discrete, nominal).

It is worth noting that each time we run the SAFS algorithm we can find a sub-optimal sub-set of s features from the original q. In principle, different independent runs of SAFS can lead to different sub-sets of s features.

3 Ranking feature relevance

SA is an algorithm implementing a stochastic time-varying dynamical system where the state vector evolves in the direction of the minima of the generalized energy function.

In our case during the evolution of the SAFS algorithm the bits set in the state vector **g** will be related to the more relevant features (genes) with increasing probability.

Table 2. Parameters choice.

Meaning	Symbol	Value
Number of random perturbations of **g** used to estimate the initial value of T	p	10000
Number of features to be selected	s	20
Number of bits to be switched for making a move	r	3
Cooling parameter	α	0.9
Aging constant	γ	0.98
Maximum number of iteration at each T	f_{max}	10000
Minimum number of success for each T	h_{min}	1000
FCM algorithm repetitions for each move	l	5
Number of clusters	c	2
Fuzzification parameter	m	2

The features more relevant in cluster discrimination should appear soon in the set of bits set to 1 and will be more frequent in the following iterations of the algorithm.

In order to estimate the relevance of features, we can include in the SAFS an aging algorithm. To this aim,we can define a vector $\mathbf{r} = \left(r_1, r_2, \ldots, r_{iq}\right)$. At Step 2 of the SAFS algorithm, we set $r_i = 1/q \ \forall i$. Every time a perturbation is accepted (Step 5.f), according to the Boltzmann distribution, we update **r** using this formula:

$$\mathbf{r} = \gamma\mathbf{r} + \mathbf{g} \tag{2}$$

where γ is the aging constant chosen in the interval [0,1], and then we normalize the vector **r** using the following constraint:

$$\sum_{i=1}^{N} r_i = 1 \tag{3}$$

At the end of the SAFS the vector **r** tells us how long each feature has belonged to it in the last few successful moves of the algorithm. We give then to vector **r** an interpretation as vector of feature relevances.

4 Experimental validation

The method was tested on the publicly available Leukemia data by Golub et al. [7]. The Leukemia problem consists in characterizing two forms of acute leukemia, Acute Lymphoblastic Leukemia (ALL) and Acute Mieloid Leukemia (AML). The original work proposed both a supervised classification task ("class prediction") and an unsupervised characterization task ("class discovery"). Here we obviously focus on the latter, but we exploit the diagnostic information on the type of leukemia to assess the goodness of the clustering obtained.

The data set contains 38 samples for which the expression level of 7129 genes has been measured with the DNA microarray technique (the interesting human genes are

Table 3. The ten most relevant genes found in a run ranked in order of relevance.

Name	**Description**
M33680_at	26-kDa cell surface protein TAPA-1 mRNA
J03801_f_at	LYZ Lysozyme
X04085_rna1_at	Catalase *EC*1.11.1.6 5'flank and exon 1 mapping to chromosome 11, band p13 (and joined CDS)
S71043_rna1_s_at	Ig alpha 2=immunoglobulin A heavy chain allotype 2 {constant region, germ line} [human, peripheral blood neutrophils, Genomic, 1799 nt]
M19722_at	FGR Gardner-Rasheed feline sarcoma viral $v-fgr$ oncogene homolog
AB002332_at	KIAA0334 gene
M10942_at	Metallothionein-Ie gene (hMT-Ie)
HG2238-HT2321_s_at	Nuclear Mitotic Apparatus Protein 1, Alt. Splice Form 2
S34389_at	HMOX2 Heme oxygenase (decycling) 2
M96956_at	TDGF1 Teratocarcinoma-derived growth factor 1

6817, and the other are controls required by the technique). These expression levels have been scaled by a factor of 100. Of these samples, 27 are cases of ALL and 11 are cases of AML. Moreover, it is known that the ALL class is in reality composed of two different diseases, since they are originated from different cell lineages (either T-lineage or B-lineage). In the data set, ALL cases are the first 27 objects and AML cases are the last 11. Therefore, in the presented results, the object identifier can also indicate the class (ALL if id ≤ 27, AML if larger).Using those data (with dimensionality $q = 7129$), Golub et al. [7] selected a set of 50 most relevant genes.

We describe here the results obtained using the SAFS algorithm. In the implementation of SAFS we used as the clustering algorithm the Fuzzy C-Means (FCM) [1] that is one of most popular clustering algorithms. Moreover, as the Leukemia data base contains is labeled, we used the *Representation Error* (RE) as an evaluation the generalized energy E.

It is worth noting that the FCM is an unstable algorithm, as his results depend not only from even small perturbations of the data set, but also from the initialization of his parameters (i.e., number of clusters c, clusters centroids y_k and fuzziness parameter m). For this reason at the beginning of the SAFS algorithm (Step 3) and for each perturbation (Step 5c) of SAFS we run the FCM $l = 5$ times and we choose the solution corresponding to the minimum of the generalized energy E.

SAFS has been implemented in R-language (*http://www.r-project.org/*) under Linux operating system. On a Pentium IV 1900 Mhz personal computer a complete running of SAFS least about 10 hours (involving the run of about one million FCMs).

We done 10 independent runs of SAFS using the assumptions in Tab. 2. For each run we obtained a different sets of 20 genes giving a $RE = 0$, containing at the least one gene found by Golub et al. [7]. In Tab. 3, we list the ten must relevant genes found in a run.

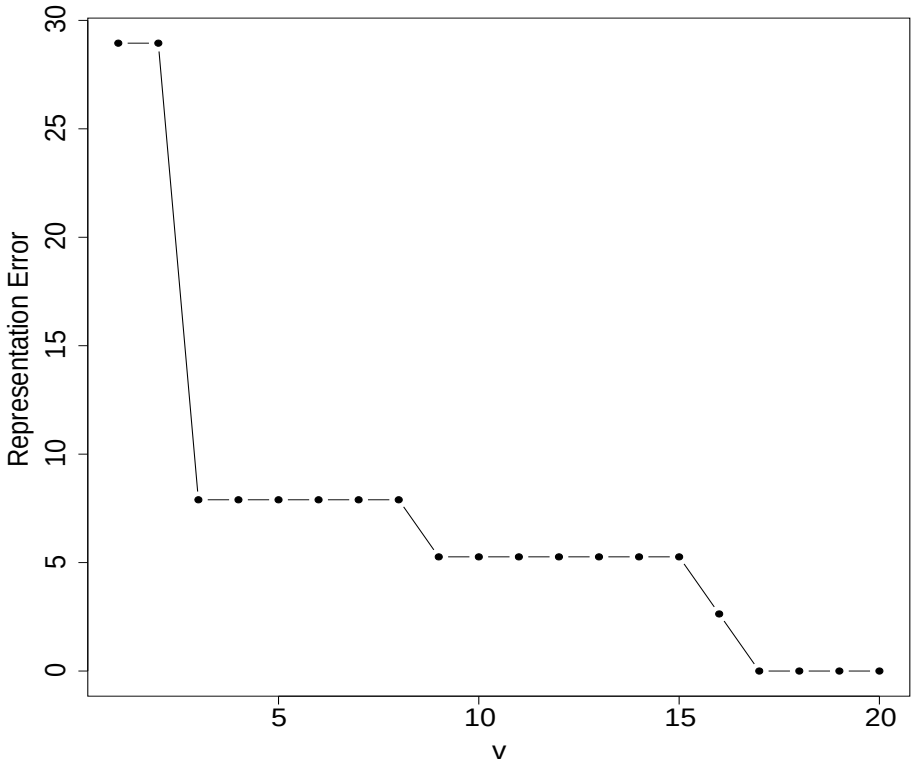

Fig. 1. Representation Error versus the size (v) of gene subsets.

Adding the relevance vectors r obtained in the 10 runs, the genes ranked at positions 1, 2, 4 are contained also in the set selected in unsupervised way by Golub et al. [7]. This is a symptom of a strong redundancy in the features of the data set.

In Fig. 1, we show the Representation Error (RE) computed using subset of genes including the v most relevant ones. As shown, at least the 17 most relevant genes must be considered in clustering in order to obtain $RE = 0$.

5 Conclusions

In this paper we have proposed a wrapper method for selecting features based on simulated annealing technique [9] and FCM algorithm [1]. The proposed approach, even if computationally intensive, permits to select the most relevant features (genes), and to rank their relevance, allowing to improve the results of clustering algorithms.

On the 7129-dimensional Leukemia data set by Golub et al. [7] the proposed feature selection method is able to find for each run a subset of 20 genes, that is sufficient to perform FCM clustering algorithm with null Representation Error.

It is worth noting that the proposed algorithm can work with every kind of features (e.g., continuous, ordinal, binary, discrete, nominal). Moreover, the proposed feature selection approach using simulated annealing can be used also with other learning machines, not only for unsupervised clustering, but also for supervised classification, regression, etc.

Acknowledgment

Work funded by the Italian Ministry of Education, University and Research (2004 "Research Projects of Major National Interest", code 2004062740), and the Biopattern EU Network of Excellence.

References

1. J.C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.
2. D.R. Bickel, Robust cluster analysis of microarray gene expression data with the number of cluster determined biologically, Bioinformatics, vol. 19, no. 7 pp. 818–824, 2003.
3. A. Blum and P. Langley, Selection of Relevant Features and Examples in Machine Learning, Artificial Intelligence, vol. 97, nos. 1–2, pp. 245–271, 1997.
4. N. Bolshakova and F. Azuaje, Cluster validation techniques for genome expression data Source, Signal Processing, vol.83, pp. 825–833, 2003.
5. R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.
6. J.G. Dy, C.E. Brodley, A. Kak, L.S. Broderick, and A.M. Aisen, Unsupervised Feature Selection Applied to Content-Based Retrieval of Lung Images, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 25, no. 3, pp. 373–378, 2003.
7. Golub, T., Slonim, D., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J., Coller, H., Loh, M., Downing, J., Caligiuri, M., Bloomfield, C., Lander, E.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science vol. 286, pp. 531–537, 1999.
8. R. Jornsten, B. Yu, Simultaneous gene clustering and subset selection for sample classification via MDL, Bioinformatics, vol. 19, no. 8, pp. 1100–1109, 2003.
9. S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. Science, vol. 220, pp.661–680, 1983.
10. R. Kohavi and G. John, Wrappers for Feature Subset Selection, Artificial Intelligence, vol. 97, nos. 1–2, pp. 273–324, 1997.
11. M.H. Law, M.A.T. Figueiredo, and A.K. Jain, Simultaneous Feature Selection in and Clustering Using Mixture Models, IEEE Trans. Pattern Analysis and Machine Intelligence, vol 28, no. 9, 2004.
12. N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of state calculations for fast computing machines. Journal of Chemical Physics, vol. 21, pp. 1087–1092, 1953.
13. P. Mitra and C.A. Murthy, Unsupervised Feature Selection Using Feature Similarity, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 3, pp. 301–312, Mar. 2002.
14. B. Mumey, L. Showe, M. Showe, A Combinatorial Approach to Clustering Gene Expession Data, Bioinformatics, 2003.
15. Q. Wang, Y. Shen, Y. Zhang, and JQ. Zhang, A quantitative method for evaluating the performances of hyperspectral image fusion, IEEE Trans. Instrumentation and Measurement vol. 52, pp. 1041–1047, 2003.