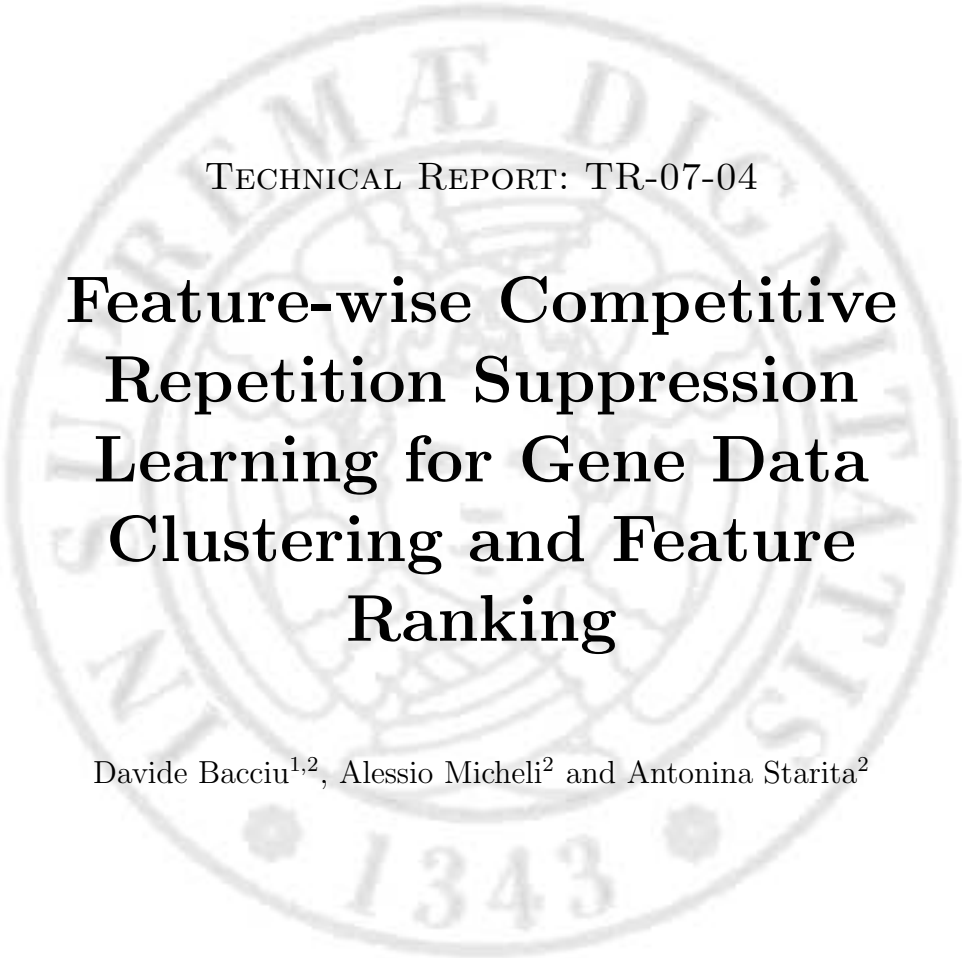
Università di Pisa
Dipartimento di Informatica



Technical Report: TR-07-04

Feature-wise Competitive Repetition Suppression Learning for Gene Data Clustering and Feature Ranking

Davide Bacciu[1,2], Alessio Micheli[2] and Antonina Starita[2]

February 9, 2007

Address: Largo B. Pontecorvo 3, 56127 Pisa, Italy. Tel: +39 050 2212700 Fax: +39 050 2212726

Feature-wise Competitive Repetition Suppression Learning for Gene Data Clustering and Feature Ranking

Davide Bacciu[1,2], Alessio Micheli[2] and Antonina Starita[2]

[1] IMT Lucca Institute for Advanced Studies,
Via San Micheletto 3, 55100 Lucca, Italy,
davide.bacciu@imtlucca.it
[2] Dipartimento di Informatica, Università di Pisa,
Largo B. Pontecorvo 3, 56127 Pisa, Italy
{micheli,starita}@di.unipi.it

Abstract. The paper extends Competitive Repetition-suppression (CoRe) learning to deal with high dimensional data clustering. We show how CoRe can be applied to the automatic detection of the unknown cluster number and the simultaneous ranking of the features according to learned relevance factors. The effectiveness of the approach is tested on two freely available data sets from gene expression data and the results show that CoRe clustering is able to discover the true data partitioning in a completely unsupervised way, while it develops a feature ranking that is consistent with the state-of-the-art lists of gene relevance.

1 Introduction

DNA microarray technology has made available extended collections of data, comprising information related to the expression levels of thousands of genes with respect to the most important biological processes. The discovery of the patterns hidden in gene expression data and the identification of the most informative portions of this consistent collections have became two major issues in genomic data analysis. Clustering has long being used to discover group structures in a set of data and constitutes an essential tool for the unsupervised exploration of information collections. However its effectiveness in this field is highly limited by the the nature of the genomic datasets, that are characterized by a small cardinality (i.e. the number of samples) in a high dimensional space (i.e. the number of genes). For instance, classical distance-based clustering algorithms like k-means, FCM and RPCL suffer when applied to these problems, since they seek for areas where the data is especially dense [1]. Moreover, often most of the features are irrelevant for the determination of the cluster membership: their presence may impair the detection of the "correct" data partition (see the analysis in [2]). Recently, several algorithms have been proposed to deal with the peculiarities of the genomic data and, in particular, with feature selection. However, most of them, are based on supervised learning, while the few

unsupervised models often exploit computationally intensive approaches such as resampling [3], ensemble methods or combinatorial searches [1] to obtain good clustering results.

In this work, we present an unsupervised learning model that deals with the cluster number identification problem and that achieves contemporaneous clustering and feature ranking in a completely unsupervised and incremental way. Our model inspires to the characteristics of a cortical memory mechanism, called *repetition suppression* [4] (RS). This mechanism induces long-lasting changes to the visual cortex, decreasing the neural activity as a consequence of the repeated presentation of similar stimuli. In brief, it induces a sharpening of the neural representation of items by means of an overall reduction of the number of active neurons which is counterbalanced by the steepening of the response of the most item-selective neurons. We modeled these RS characteristics in an unsupervised learning algorithm, named *competitive repetition-suppression* (CoRe) *learning* [5], that trains pools of selective pattern detector units, generating compact neural representations of the input data. The present work extends the original model to deal with feature selection: as a by-product of its execution, CoRe produces a relevance ranking that is used to select the most informative features. Moreover, we exploit the RS mechanism to reduce the effect of irrelevant and redundant features and to obtain good performance also when clustering on the whole feature space. In addition, we exploit the robustness of CoRe with respect to low density clusters to correctly determine the unknown cluster number. In our intention, this model should serve as computationally simple tool (i.e. in the order of magnitude of k-means or FCM) for the unsupervised exploration of high dimensional data, producing incremental evaluations of the data partitioning and features relevance as learning proceeds.

2 Competitive Repetition Suppression Learning

Competitive repetition-suppression (CoRe) learning is a soft-competitive [6] model that allows only a subset of the most active units to learn in proportion to their activation strength, while it penalizes the least active units, reducing their responses to the patterns that produced a low firing strength. Each CoRe unit $u_i \in U$ is characterized by a prototype $\boldsymbol{c}_i$ that determines its preferred stimulus, and by an activation function $\varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i)$ (parametrical w.r.t. to the vector $\boldsymbol{\lambda}_i$) that determines the unit's response to the d-dimensional input pattern $\boldsymbol{x}_k \in \chi$.

The CoRe competition is engaged between two sets of units: at each step the most active units are selected to form the *winners pool*, i.e.

$$win_k = \{i \mid \varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i) \geq \theta_{win},\ u_i \in U\}, \tag{1}$$

while the remainder is inserted into the *losers pool*, that is

$$lose_k = \{i \mid \varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i) < \theta_{win},\ u_i \in U\} \tag{2}$$

where the winners threshold θ_{win} is a meta-parameter determining the level of competition between the units.

Modeling the stimuli repetition is a key issue for implementing a learning scheme based on repetition suppression. In our model we define a parameter, named *stimulus predominance*, that represents an approximate measure of the pattern frequency. The stimulus predominance at the time t is defined as

$$\nu_i^t = \frac{1}{|\chi_t|} \sum_{\boldsymbol{x}_k \in \chi_t} \frac{\varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i)}{z_U^k}, \tag{3}$$

where χ_t is the set of the input patterns presented to the network up to time t and the fraction in the sum is the relative activation of the i-th unit upon the presentation of input $\boldsymbol{x}_k$, where z_U^k acts as a normalization factor. For instance, the term z_U^k can be chosen as the output of the maximally active neuron, from the units set U, on the pattern x_k, i.e. $z_U(x_k) = max_{u_j \in U}\{\varphi_j(x_k|\lambda_j)\}$. The general objective of the definition in (3), is to measure the frequency of the patterns that are preferred by the unit with prototype c_i, scaled by the relative activation strength with respect to the contribution of the maximally similar prototype z_U^k. The stimulus predominance is used to regulate the amount of penalization that is applied to the losers, defined as

$$RS_k^t = \frac{1}{M|win_k|} \sum_{i \in win_k} \nu_i^t \varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i), \tag{4}$$

that is the repetition suppression produced by the pattern $\boldsymbol{x}_k$ at time t, where M is the maximum of the activation function φ_i.

The expression in (4) is used to calculate the penalization for the loser neurons. For instance, we can use it to define a pseudo-target activation for units in the losers pool as $\hat{\varphi}_i(\boldsymbol{x}_k) = \varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i)(1 - RS_k^t)$. This reference signal forces the losers to reduce their activation proportionally to the amount of repetition suppression RS_k^t that is applied.. The error of the i-th loser can thus be written as

$$\underline{E}_{i,k} = \frac{1}{2}\left(\hat{\varphi}_i(\boldsymbol{x}_k) - \varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i)\right)^2 = \frac{1}{2}\left(\varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i) RS_k^t\right)^2. \tag{5}$$

Conversely, the responses of the winner units are strengthened by applying a learning rule that minimizes the winners' error

$$\overline{E}_{i,k}^t = (M - \varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i)) \tag{6}$$

that is, the target activation for the neurons $u_i \in win_k$ is M, i.e. the maximum of the activation function φ_i. The vector of unit parameters $\boldsymbol{\lambda}_i$ can be adapted by an optimization procedure that minimizes the error functions defined in (5) and (6).

Since the task of CoRe learning is to train highly selective units, it is important to define a metric for identifying the most significant (selective) units which have been produced by the learning process. For this reason we define the *relevance factor* for the unit u_i as

$$\hat{\nu}_i^t = \frac{1}{\nu_i^t |\chi_t|} \sum_{\boldsymbol{x}_k \in win_{u_i}^t} \frac{\varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i)}{z_{win_k}^k}, \tag{7}$$

where $z^k_{win_k}$ follows the definition given above and $win^t_{u_i}$ is the set of patterns $\boldsymbol{x}_k \in \chi_t$ for which unit u_i was in the winners pool, i.e.

$$win^t_{u_i} = \{\boldsymbol{x}_k \mid \varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i) \geq \theta_{win},\ \boldsymbol{x}_k \in \chi_t\}. \tag{8}$$

In other words, the relevance factor defines a soft-measure of the frequency with which u_i was the most active unit in the winners pool. This measure can be used to determine which units are less significant for the representation of the input patterns and can be used, for instance, for deciding whether a unit has to be removed from the pool.

3 Learning Feature Relevance with CoRe Clustering

The general CoRe learning formulation described in the previous section can be readily applied to clustering problems and, in particular, to the issue of automatically determining the unknown cluster number. For instance, each CoRe unit can be interpreted as a possible cluster detector, where the prototype $\boldsymbol{c}_i$ defines the cluster representative and the activation function $\varphi_i(\boldsymbol{x}_k|\boldsymbol{\lambda}_i)$ (e.g. a gaussian centered in $\boldsymbol{c}_i$) determines whether the pattern $\boldsymbol{x}_k$ belongs to the i-th cluster. In this sense, CoRe clustering works essentially by evolving a small set of highly selective cluster detectors out of an initially larger population by means of a reward-punishment procedure that resembles the rival penalization mechanism of the RPCL [7] algorithm.

The current CoRe formulation performs cluster identification without considering the relevance of the features composing the input vectors neither in its decision procedure nor at the learning stage. In particular, the repetition suppression effect, along with the stimulus predominance and the relevance factor, are defined and used only on per-unit basis. In other words, this means that RS^t_k, ν^t_i and $\hat{\nu}^t_i$ are all scalars. Neglecting the contribution of relevant features w.r.t. irrelevant dimensions impairs the performance of CoRe clustering when dealing with high dimensional spaces. However, the CoRe framework can be seamlessly extended, introducing the repetition suppression competition also on a per-feature basis. In our intention this should lead to an algorithm that, by means of the same mechanism, is capable of automatically determining the unknown cluster number using an iterative unit pruning strategy that selects a set of relevant units, corresponding to the identified clusters. Moreover, by using an iterative feature pruning strategy similar to that used for the units, the CoRe algorithm should be able to select the subset of features that is most relevant for cluster membership. On the other hand, the relevance score can be used to rank the features and to apply a standard top-k selection step at the end of training, without resorting to feature pruning.

First, we redefine the stimulus predominance of unit u_i at time t to be the d-dimensional vector $\boldsymbol{\nu}^t_i = [\nu^t_{i1}, \ldots, \nu^t_{il}, \ldots, \nu^t_{id}]^T$, where d is the cardinality of the input space χ and ν^t_{il} is the stimulus performance restricted to the l-th feature,

that is

$$\nu_{il}^t = \frac{1}{|\chi_t|} \sum_{\boldsymbol{x}_k \in \chi_t} \left\{ \frac{\varphi_{il}(x_k(l)|\lambda_{il})}{\varphi_{z(U,\boldsymbol{x}_k)l}(x_k(l)|\lambda_{z(U,\boldsymbol{x}_k)l})} \right\}_{\mathbb{1}} \text{ with } \{v\}_{\mathbb{1}} = \begin{cases} 1 & v > 1 \\ v & v \leq 1 \end{cases}. \quad (9)$$

The term $\varphi_{il}(x_k(l)|\lambda_{il})$ in (9) represents the activation strength of the i-th unit restricted to the l-th component of the input $\boldsymbol{x}_k$, while $z(U, \boldsymbol{x}_k)$ is the index of the unit that activated most for the pattern $\boldsymbol{x}_k$, that is

$$z(U, \boldsymbol{x}_k) = \arg\max_{u_j \in U} \{\varphi_j(\boldsymbol{x}_k|\lambda_j)\}. \quad (10)$$

The function $\{\cdot\}_{\mathbb{1}}$ is used in (9) to enforce the condition $\boldsymbol{\nu}_i^t \leq 1$ and, consequently, $\boldsymbol{RS}_k^t \leq 1$. Here we assumed that also the repetition suppression is applied feature-wise: therefore $\boldsymbol{RS}_k^t = [RS_{k1}^t, \ldots, RS_{kl}^t, \ldots, RS_{kd}^t]^T$ is a vector that applies different levels of suppression to the single components of the losers activation function. Adapting (4) to comply with the new formulation, we rewrite the equations for the $\boldsymbol{RS}_k^t$ components as

$$RS_{kl}^t = \frac{1}{M|win_k|} \sum_{i \in win_k} \nu_{il}^t \varphi_i(x_k(l)|\boldsymbol{\lambda}_i). \quad (11)$$

Finally, the relevance factor for the unit u_i is the vector $\hat{\boldsymbol{\nu}}_i^t = [\hat{\nu}_{i1}^t, \ldots, \hat{\nu}_{il}^t, \ldots, \hat{\nu}_{id}^t]^T$ where $\hat{\nu}_{il}^t$ is the relevance of the l-th feature for the i-th prototype, defined as follows

$$\hat{\nu}_{il}^t = \frac{1}{\nu_{il}^t |\chi_t|} \sum_{\boldsymbol{x}_k \in win_{u_i}^t} \left\{ \frac{\varphi_{il}(x_k(l)|\lambda_{il})}{\varphi_{z(win_k,\boldsymbol{x}_k)l}(x_k(l)|\lambda_{z(win_k,\boldsymbol{x}_k)l})} \right\}_{\mathbb{0}} \quad (12)$$

where

$$\{v\}_{\mathbb{0}} = \begin{cases} 0 & v > 1 \\ v & v \leq 1 \end{cases}. \quad (13)$$

The term $z(win_k, \boldsymbol{x}_k)$ in (12) follows the definition in (10), while $win_{u_i}^t$ defines the set of patterns $\boldsymbol{x}_k \in \chi_t$ for which unit u_i was in the winners pool, i.e. $win_{u_i}^t = \{\boldsymbol{x}_k \mid \varphi_i(\boldsymbol{x}_k|\lambda_i) \geq \theta_{win},\ \boldsymbol{x}_k \in \chi_t\}$. The function $\{\cdot\}_{\mathbb{0}}$ in (13), squashes its argument to zero if it exceeds 1. The rationale behind this choice is to penalize the relevance of a feature l from a unit $u_i \in win_k$ whenever it produces an high feature activation $\varphi_{il}(x_k(l))$ in correspondence to a low feature activation $\varphi_{wl}(x_k(l))$ in the unit u_w that is the maximally active neuron for the pattern x_k, i.e. $w = \max_j \varphi_j(\boldsymbol{x}_k, \boldsymbol{\lambda}_j)$ (see Fig. 1).

The algorithm for feature-wise CoRe clustering is completed by deriving the update rules for the unit parameters $\boldsymbol{\lambda}_i$ (including the prototype $\boldsymbol{c}_i$). Table 1 reports the pseudo-code for the CoRe clustering algorithm with gaussian activation functions having mean $\boldsymbol{c}_i$ and variance $\boldsymbol{\sigma}_i$. The update rules for the gaussian parameters $\boldsymbol{\lambda}_i = \{\boldsymbol{c}_i, \boldsymbol{\sigma}_i\}$ can be derived by differentiating the CoRe errors with respect to each parameter component $l = 1, \ldots, d$, that is $\triangle \underline{\lambda}_{il} = \frac{\partial \underline{E}_{il,k}^t}{\partial \lambda_{il}}$ for the loser units and $\triangle \overline{\lambda}_{il} = \frac{\partial \overline{E}_{il,k}^t}{\partial \lambda_{il}}$ for the winners, where

$$\underline{E}_{il,k}^t = \frac{1}{2}(\varphi_{il}(x_k(l)|\lambda_{il}) RS_{kl}^t)^2 \quad (14)$$

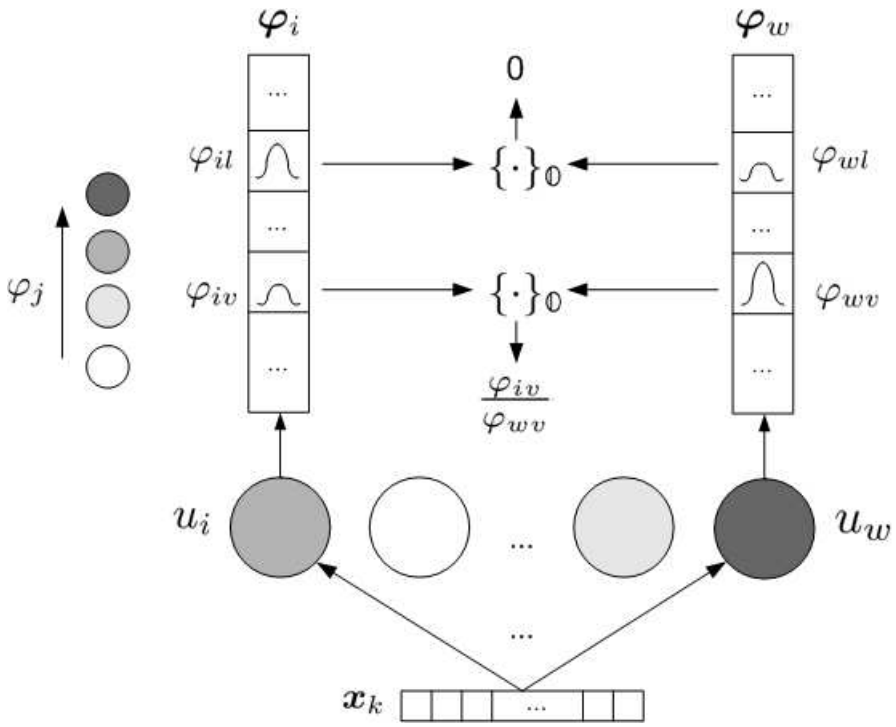


Fig. 1. Feature-wise calculation of the relevance factor for the unit u_i on the input $\boldsymbol{x}_k$: the gray-levels identify increasing levels of unit activation φ_j; the most active unit u_w is identified by the darkest color. The relevance of the l-th feature of u_i is squashed to 0 by the function $\{\cdot\}_0$ since the unit u_w that is most selectively tuned to $\boldsymbol{x}_k$ has a low l-feature activation φ_{wl}. Conversely the relevance of the v-th feature is reinforced proportionally to φ_{iv} because the v-th component of the activation function in u_w is high (see φ_{wv}).

and

$$\overline{E}^t_{il,k} = (M - \varphi_{il}(x_k(l)|\lambda_{il})\ ). \tag{15}$$

For an in-depth description of the learning equations and the related derivation steps, refer to Appendix A.

4 Experimental Evaluation

The aim of this section is to describe the behavior and the performance of the CoRe clustering algorithm on simulated and real gene expression data collected with DNA microarrays. All the results described in this section have been obtained for gaussian CoRe units having the parameters described in Table 2. The trials have been repeated 10 times and the results averaged. The algorithm

Given: $\chi = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ *(the input data),* θ_{win} *(the winners threshold),* θ_{pr-u} *(the unit pruning threshold),* θ_{pr-d} *(the dimension pruning threshold),* K *(the initial unit number), initialize* $\{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K\}$ *(gaussian centers) randomly and* $\{\boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_K\}$ *proportionally to the data covariance, set* $\theta_{decay} = 1 - \frac{1}{N}$ *(the relevance factor exponential decay),* $\hat{\boldsymbol{\nu}}_i^t = 1$ *and* $\hat{\boldsymbol{\nu}}_i^t = 1$ *for* $i = 1, \ldots, K$*. Do*

Step 0 At each learning time t, randomly pickup a sample $\boldsymbol{x}_k$ from the dataset χ.
Step 1 For each unit u_i do
  a. Calculate the features activation $\varphi_{il}^t(x_k(l)|\lambda_i) = \exp\left(-\frac{(x_k(l)-c_{il})^2}{2\sigma_{il}^2}\right)$ and the unit activation $\varphi_i^t(\boldsymbol{x}_k|\boldsymbol{\lambda}_i) = \exp\left(-\sum_{l=1}^{d^t} \frac{(x_k(l)-c_{il})^2}{2\sigma_{il}^2}\right)$;
  b. If $\varphi_i^t(\boldsymbol{x}_k|\boldsymbol{\lambda}_i) \geq \theta_{win}$ then add i to win_k^t and $\boldsymbol{x}_k$ to $win_{u_i}^t$, else add i to $lose_k^t$.
Step 2 If win_k^t is empty, set $win_k^t = \{w\}$, where $w = \arg\max_{u_j \in U}\{\varphi_j^t(\boldsymbol{x}_k|\boldsymbol{\lambda}_j)\}$, and remove w from $lose_k^t$.
Step 3 For each unit u_i
  a. Update the stimulus predominance $\boldsymbol{\nu}_i^t$ as in (3);
  b. If $i \in win_k^t$ update the relevance factor $\hat{\boldsymbol{\nu}}_i^t$ as in (7)
  c. If $i \in lose_k^t$ apply the relevance factor decay $\hat{\boldsymbol{\nu}}_i^t = \theta_{decay} \cdot \hat{\boldsymbol{\nu}}_i^{t-1}$
Step 5 Compute the repetition suppression $\boldsymbol{RS}_k^t$.
Step 6 For each u_i do
  a. Update the unit parameters $\boldsymbol{c}_i$ and $\boldsymbol{\sigma}_i$ by gradient descent;
Step 7 Feature and unit pruning
  a. If $\hat{\nu}_{il}^t \leq \theta_{pr-d}$ for all $i = 1, \ldots, n^t$ then remove the l-th feature from all the units;
  b. If $\hat{\nu}_{il}^t \leq \theta_{pr-u}$ for all $l = 1, \ldots, d^t$ then prune the unit;
  c. If pruning has occurred, reset $\boldsymbol{\nu}_i^t$ and $\hat{\boldsymbol{\nu}}_i^t$.

Iterate until convergence.

Table 1. Feature-wise CoRe Clustering Algorithm. Note that the time index t has been omitted from the unit parameters λ_i to ease the notation.

stopping condition depends on the convergence to zero of the Mean Applied Repetition Suppression (MARS), that is

$$MARS^t = \frac{1}{|\chi^t|} \sum_{k=1}^{|\chi^t|} \left[\sum_{i \in lose_k} \left(\sum_{l=0}^{d^t} \frac{RS_{kl}^t \cdot \varphi_{il}(x_k(l))^2}{d^t} \right) \right]. \tag{16}$$

The MARS score measures the level of competition that is ongoing into the CoRe network, by summing up the repetition suppression, scaled by the loser unit activation, that is generated during one learning epoch. CoRe stopping condition is determined by the convergence of MARS score to zero and the contemporaneous satisfaction of a clustering stability criterion (i.e. the cluster assignment remains the same for at least 4 epochs).

First, we considered the `Simulated6` [3] dataset, comprising 60 samples consisting of 600 genes, that is known to be partitioned in 6 clusters of various sizes and that are characterized by gene markers of different strength. In particular, the dataset can be partitioned into 6 classes consisting of 8, 12, 10, 15, 5 and 10 samples, respectively, each marked by 50 distinct genes uniquely up-regulated for

Table 2. CoRe meta-parameters

Description	Name	Value
Winners threshold	θ_{win}	0.9
Unit pruning threshold	θ_{pr-u}	0.005
Dimension pruning threshold	θ_{pr-d}	0.01
Initial variance	σ_i	1
Winners' prototype learning rate	α_{cw}	0.05
Winners' variance learning rate	α_{sw}	0.0005
Losers' prototype learning rate	α_{cl}	0.005
Losers' variance learning rate	α_{sl}	0.0001

that class [3]. In addition, it contains 300 noisy genes not "coding" any particular class. The CoRe clustering algorithm was run on the `Simulated6` dataset, with $K = 10$. The results in Table 3 show that CoRe clustering is able to correctly identify the six clusters, pruning the irrelevant units, and correctly partitioning the data. If we compare our results with those obtained by Hierarchical Clustering and k-means, we see that CoRe achieves better results in terms of cluster number identification and classification accuracy. In particular, the parsimonious CoRe cluster allocation policy does not allow the creation of a singleton cluster in correspondence of sample 8, that is characterized by up-regulated genes both in class 1 and class 2 [3]. Figure 2 shows a graphical interpretation of the relevance factor corresponding to the 6 CoRe clusters ($C_1, \ldots, C_6$): the darker areas correspond to genes with low relevance, while the whiter parts indicate high relevance values. As one would have expected, we see that each cluster is characterized by a set of 50 relevant genes: some clusters, e.g. C_1 and C_2, show a sharp distinction between relevant and irrelevant genes, while others, e.g. C_5 and C_6, are characterized by weaker markers. The relevance factor gives an interesting visual insight into the features properties and can be used to select a subset of highly informative genes. For instance, clustering on the basis of the top-50 genes (w.r.t. $\hat{\boldsymbol{\nu}}_i^t$) characterizing each cluster, automatically discards 300 non-informative genes and raises k-means clustering performance to $Rand_{k-m} = 1$.

The second set of simulations was run on real gene expression data from a widely adopted benchmark dataset, that is the Leukemia dataset by Golub [9]. The training data consists of 38 bone marrow samples comprising the expression levels of 7129 genes obtained from acute leukemia patients at the time of the

Table 3. Estimated number of clusters and classification accuracy for Hierarchical Clustering (HC), k-means($k-m$) and CoRe in terms of the corrected Rand index [8].

Dataset	K_{true}	K_{HC}	K_{k-m}	K_{CoRe}	$Rand_{HC}$	$Rand_{k-m}$	$Rand_{CoRe}$
`Simulated6`	6	7	6	6	0.986	0.962	1
`Leukemia`	3	5	2	3	0.648 (0.46)	0.419	0.845
	2	-	2	2	-	0.351	0.946

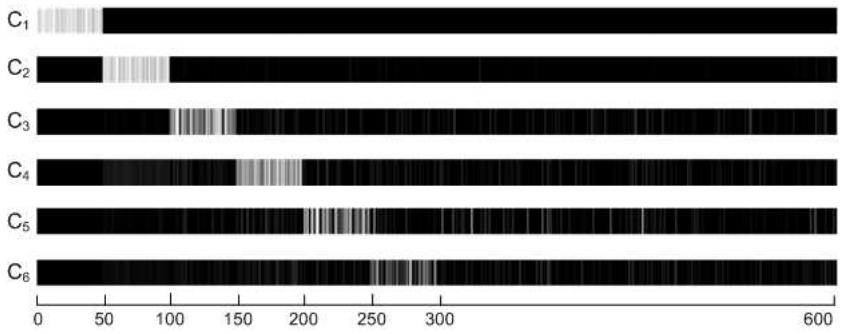


Fig. 2. Expression patterns for the 6-class simulated dataset. The graylevel scales represent the relevance factor $\hat{\boldsymbol{\nu}}_i^t$ of the single genes for each of the identified clusters: black indicate zero relevance while white represents full relevance. The relevance factors have been mean subtracted and scaled in $[0, 1]$.

diagnosis. The dataset can be partitioned into 3 relevant classes: 8 T-Lineage Acute Lymphoblastic Leukemia (ALL) samples, 19 B-lineage ALL samples and 11 Acute Myeloid Leukemia (AML) samples. Most of the supervised learning algorithms tested on the Leukemia data focused on a reduced problem, that is partitioning the dataset into 11 AML and 27 ALL samples. In this work, we concentrate on unsupervised class discovery, so we let the algorithm decide whether to partition the data into 2 or 3 clusters. The results in Table 3 show that initially CoRe learning identifies 3 clusters. In other words, with the MARS stopping criterion used so far, CoRe clustering converged having identified the original T-ALL, B-ALL and AML classes (see the MARS score at the "3 Class" point in Figure 3), with a clustering performance of 0.845. However, if we relax the stopping condition and we let the CoRe algorithm run beyond its first convergence point, we notice that the amount of applied repetition suppression starts raising again (see Figure 3), indicating an ongoing competition between units and features. After 30/35 epochs the algorithm converges again, this time identifying 2 clusters (the "2 Class convergence" point in Figure 3), corresponding to the 2 ALL/AML classes ($Rand_{CoRe} = 0.946$). The Hierarchical Clustering algorithm, on the other hand, identifies 5 partitions, with 2 clusters identifying correctly the AML and T-ALL classes while the remaining 3 clusters partitioned further the B-class [3]. Moreover, forcing the HC method to partition the data into 3 classes reduces further its clustering precision (see the Rand index in brackets in Table 3). Again, the k-means algorithm obtained the lowest Rand scores within the methods, both when run with $k = 3$ and $k = 2$.

The drastic reduction in the clustering performance of k-means and HC can be explained by looking more in depth at the nature of the Leukemia data. In [2] it is shown that the leukemia classes overlaps significantly in the 7129-dimensional space, while they are neatly separable when a subset of informative features are used. The CoRe algorithm exploits the feature-wise competition mechanism to suppress the contribution of the irrelevant genes, achieving better

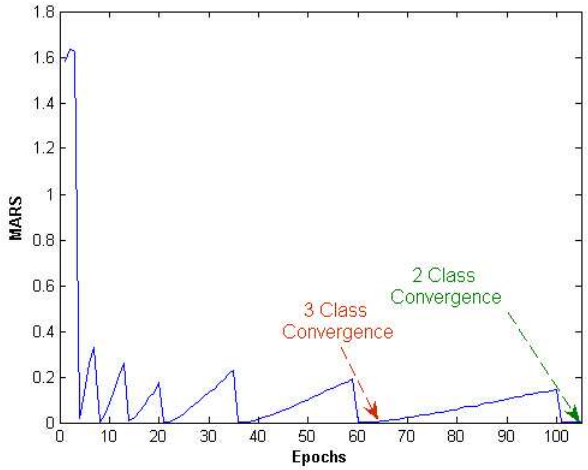
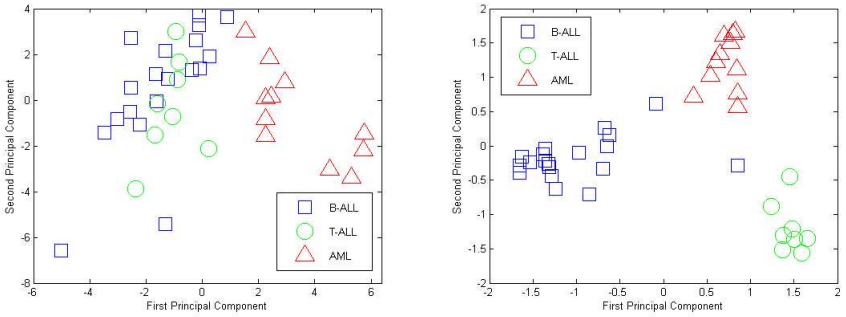


Fig. 3. Behavior of the Mean Applied Repetition Suppression Score (MARS) for the Leukemia dataset. The MARS score drops to zero each time a unit is pruned. The epoch identified by the "3 Class Convergence" label denotes the initial convergence point of the CoRe algorithm (i.e. $MARS = 0$ and stable clustering for at least 4 epochs). The epoch identified by the "2 Class Convergence" label denotes the second convergence point of the CoRe algorithm corresponding to the identification of 2 classes in the Leukemia data.

results when the full set of features is used. Figure 4 gives a visual interpretation of the problem, by projecting the Leukemia data along its first two principal components. In figure 4.a it is shown a plot of components when all the 7129 genes are used: the two classes in the ALL samples (B-linkage and T-linkage) are completely mixed, while the AML class is not clearly identifiable as a single cluster. Figure 4.b shows the PCA result when only the 30 most relevant genes identified by CoRe are used: the AML data is tightly clustered, while the T-ALL and B-ALL classes are almost completely separated already in two dimensions.

The feature ranking produced by CoRe clustering can be confronted with the list of the 50 most relevant genes for the Leukemia data as identified by Golub [9]. Table 4 presents an example of 8 genes from the top-20 list generated by CoRe clustering that are present in Golub's list: this result confirms that CoRe relevance ranking produces results that are consistent with the common knowledge concerning informative genes in the Leukemia problem. The same top-20 genes selected by CoRe have been used to perform again k-means clustering on the ALL/AML task, obtaining a neat improvement in clustering performance (i.e. $Rand_{k_m} = 0.7901$). If we calculate the clustering performance in terms of the Q-accuracy (i.e. the number of correctly classified sample w.r.t the total sample number), top-20 k-means obtains an accuracy of 0.947, that is the same value obtained by max-min cut hierarchical clustering in [2] using the top-50



(a) PCA on 7129 components (b) PCA on CoRe top-30 components

Fig. 4. 2D principal components for the Leukemia dataset with full genes (a) and CoRe top-30 features (b).

Table 4. Relevant features identified by CoRe clustering and listed in Golub's top-50 [9].

Code	Description
M27891_at	CST3 Cystatin C (amyloid angiopathy and cerebral hemorrhage)
M28130_rna1_s_at	Interleukin 8 (IL8) gene
M57710_at	LGALS3 Lectin, galactoside-binding, soluble
Y00787_s_at	INTERLEUKIN-8 PRECURSOR
U05259_rna1_at	MB-1 gene
M96326_rna1_at	Azurocidin gene
U22376_cds2_s_at	C-myb gene extracted from Human (c-myb) gene, complete primary cds, and five complete alternatively spliced cds
X1704_at2	PRG1 Proteoglycan 1, secretory granule

genes. CoRe Q-accuracy score on the 7129-dimensional ALL/AML task is 0.987, while the accuracy drops to 0.947 when only the top-20 genes are used. These results show that CoRe outperforms other filter approaches, such as the two-way unsupervised feature selection algorithm [10] whose score for the full-feature and on the top-50 case is 0.763 and 0.790, respectively. Recently, in [1] was presented a wrapper approach, based on simulated annealing, that obtains full Q-accuracy on the leukemia data with less than 20 genes. However, this approach does not evaluate the feature relevance from the dataset alone, whereas it explicitly searches for subsets of 20 genes minimizing the representation error, requiring a considerable effort for the combinatorial search.

5 Conclusion

We presented an extension of the CoRe learning algorithm that enables CoRe competition on a feature-wise basis. This model overcomes previous limitations of the CoRe model when dealing with high-dimensional data. In particular, we

showed how it can be applied to the automatic detection of the unknown cluster number and the simultaneous ranking of the data features with respect to a given relevance measure. The proposed approach was applied to two freely available datasets from gene expression data and the results showed that CoRe clustering is able to discover the true data partitioning in a completely unsupervised way, while it develops a feature ranking that is consistent with the state-of-the-art lists of gene relevance. CoRe obtains clustering performances that are comparable with that of complex ensemble and resampling based approaches [3, 1, 11], while retaining the low computational complexity of an incremental filter model. In particular, we advise that CoRe can be used for the development of a tool for interactive gene data exploration that produces visual interpretations of feature ranking as learning proceeds. Moreover, it would be interesting to study further the incremental approach, analyzing the effect of the introduction of fresh data on a consolidated training base.

A Derivation of the Learning Rules

In this section we describe the learning equations for the CoRe parameters. All the results described hereunder refers to gaussian units with diagonal variance matrices: hence, σ_i is a vector containing all the diagonal elements of the variance matrix of unit u_i.

The incremental learning rules for prototype adjustment and spread tuning can be obtained by applying gradient descent to the error measure in (14) and (15). The parameter increments for the units $u_i \in lose_k$ can be obtained by differentiating (14) with respect to each parameter c_{il} and σ_{il} in $\boldsymbol{\lambda}_i$. Therefore, the prototype update at time t can be calculated as

$$\triangle \underline{c}_{il,k}^t = \frac{\partial \underline{E}_{il,k}^t}{\partial c_{il}} = \varphi_{il}(x_k(l)|\lambda_{il}) RS_{kl}^t \frac{\partial(\varphi_{il}(x_k(l)|\lambda_{il}) RS_{kl}^t)}{\partial c_{il}} \tag{17}$$

where the differentiation on the right side can be expanded by chain rule as

$$\begin{aligned} \frac{\partial(\varphi_{il}(x_k(l)|\lambda_{il}) RS_{kl}^t)}{\partial c_{il}} &= \frac{\partial(\varphi_{il}(x_k(l)|\lambda_{il}) RS_{kl}^t)}{\partial \varphi_{il}(x_k(l)|\lambda_{il})} \frac{\partial(\varphi_{il}(x_k(l)|\lambda_{il}))}{\partial c_{il}} \\ &= \left(RS_{kl}^t + \varphi_{il}(x_k(l)|\lambda_{il}) \frac{\partial(RS_{kl}^t)}{\partial \varphi_{il}(x_k(l)|\lambda_{il})} \right) \cdot \\ &\quad \cdot \frac{\partial(\varphi_{il}(x_k(l)|\lambda_{il}))}{\partial c_{il}} \\ &= RS_{kl}^t \varphi_{il}(x_k(l)|\lambda_{il}) \frac{(x_k(l) - c_{il})}{\sigma_{il}^2} \end{aligned} \tag{18}$$

in which we have used $\frac{\partial(RS_{kl}^t)}{\partial \varphi_{il}(x_k(l)|\lambda_{il})} = 0$ if $u_i \in lose_k$ at time t (follows from the definition of RS in (11)). Substituting the results of (18) in (17) we obtain

$$\triangle \underline{c}_{il,k}^t = \left(\frac{\varphi_{il}(x_k(l)|\lambda_{il}) RS_{kl}^t}{\sigma_{il}} \right)^2 (x_k(l) - c_{il}). \tag{19}$$

Similarly, the spread update at time t can be calculated as

$$\begin{aligned}\triangle\underline{\sigma}_{il,k}^{t} &= \frac{\partial \underline{E}_{il,k}^{t}}{\partial \sigma_{il}} = \varphi_{il}(x_k(l)|\lambda_{il}) RS_{kl}^{t} \frac{\partial(\varphi_{il}(x_k(l)|\lambda_{il}) RS_{kl}^{t})}{\partial \sigma_{il}} \\ &= (\varphi_{il}(x_k(l)|\lambda_{il}) RS_{kl}^{t})^2 \frac{(x_k(l)-c_{il})^2}{\sigma_{il}^3}\end{aligned} \tag{20}$$

while the parameter increments for the units $u_i \in win_k$ can be written as follows

$$\triangle\overline{c}_{il,k}^{t} = \frac{\partial \overline{E}_{il,k}^{t}}{\partial c_{il}} = -\varphi_{il}(x_k(l)|\lambda_{il}) \frac{(x_k(l)-c_{il})}{\sigma_{il}^2} \tag{21}$$

$$\triangle\overline{\sigma}_{il,k}^{t} = \frac{\partial \overline{E}_{il,k}^{t}}{\partial \sigma_{il}} = -\varphi_{il}(x_k(l)|\lambda_{il}) \frac{(x_k(l)-c_{il})^2}{\sigma_{il}^3}. \tag{22}$$

The update rule for the unit parameters $\lambda_i^t = (c_i^t, \sigma_i^t)$ is

$$\lambda_i^t = \begin{cases} \lambda_i^{t-1} - \alpha_\lambda^{lose} \triangle\underline{\lambda}_{i,k}^{t} \text{ for } i \in lose_k \\ \lambda_i^{t-1} - \alpha_\lambda^{win} \triangle\overline{\lambda}_{i,k}^{t} \text{ for } i \in win_k \end{cases} \tag{23}$$

where α_λ^{win} and α_λ^{lose} are the learning and de-learning rate, respectively, with $\alpha_\lambda^{win} \gg \alpha_\lambda^{lose}$.

References

1. Filippone, M., Masulli, F., Rovetta, S.: Unsupervised gene selection and clustering using simulated annealing. In: LNCS. Volume 3849., Springer (2005) 229–235
2. Ding, C.H.Q.: Analysis of gene expression profiles: class discovery and leaf ordering. In: RECOMB '02: Proceedings of the sixth annual international conference on Computational biology, New York, NY, USA, ACM Press (2002) 127–136
3. Monti, S., Tamayo, P., Mesirov, J., Golub, T.: Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. Mach. Learn. **52**(1-2) (2003) 91–118
4. Grill-Spector, K., Henson, R., Martin, A.: Repetition and the brain: neural models of stimulus-specific effects. Trends in Cognitive Sciences **10**(1) (2006) 14–23
5. Bacciu, D., Starita, A.: Competitive repetition suppression learning. In: Proceeding of ICANN 2006. Lecture Notes in Computer Science, Springer (2006)
6. Nowlan, S.: Soft Competitive Adaptation: Neural Network Learning Algorithms based on Fitting Statistical Mixtures. Phd thesis, Carnegie-Mellon University, Pittsburg (1991)
7. Xu, L., Krzyzak, A., Oja, E.: Rival penalized competitive learning for clustering analysis, rbf net, and curve detection. IEEE Trans. on Neur. Net. **4**(4) (1993)
8. Hubert, L., Arabie, P.: Comparing partitions. Journal of Classification (1985)
9. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science **286**(5439) (1999) 531–537

10. Ding, C.H.Q.: Unsupervised feature selection via two-way ordering in gene expression analysis. Bioinformatics **19**(10) (2003) 1259–1266
11. Dugas, M., Merk, S., Breit, S., Dirschedl, P.: mdclust—exploratory microarray analysis by multidimensional clustering. Bioinformatics **20**(6) (2004) 931–936