BIOINFORMATICS ORIGINAL PAPER Vol. 23 no. 13 2007, pages 1607–1615
doi:10.1093/bioinformatics/btm158

Gene expression

Weighted rank aggregation of cluster validation measures: a Monte Carlo cross-entropy approach

Vasyl Pihur, Susmita Datta and Somnath Datta*

Department of Bioinformatics and Biostatistics, University of Louisville, Louisville, KY 40202, USA

Received on December 7, 2006; revised on April 9, 2007; accepted on April 18, 2007

Advance Access publication May 5, 2007

Associate Editor: Joaquin Dopazo

ABSTRACT

Motivation: Biologists often employ clustering techniques in the explorative phase of microarray data analysis to discover relevant biological groupings. Given the availability of numerous clustering algorithms in the machine-learning literature, an user might want to select one that performs the best for his/her data set or application. While various validation measures have been proposed over the years to judge the quality of clusters produced by a given clustering algorithm including their biological relevance, unfortunately, a given clustering algorithm can perform poorly under one validation measure while outperforming many other algorithms under another validation measure. A manual synthesis of results from multiple validation measures is nearly impossible in practice, especially, when a large number of clustering algorithms are to be compared using several measures. An automated and objective way of reconciling the rankings is needed.

Results: Using a Monte Carlo cross-entropy algorithm, we successfully combine the ranks of a set of clustering algorithms under consideration via a weighted aggregation that optimizes a distance criterion. The proposed weighted rank aggregation allows for a far more objective and automated assessment of clustering results than a simple visual inspection. We illustrate our procedure using one simulated as well as three real gene expression data sets from various platforms where we rank a total of eleven clustering algorithms using a combined examination of 10 different validation measures. The aggregate rankings were found for a given number of clusters k and also for an entire range of k.

Availability: R code for all validation measures and rank aggregation is available from the authors upon request.

Contact: somnath.datta@louisville.edu

Supplementary information: Supplementary information are available at http://www.somnathdatta.org/Supp/RankCluster/supp.htm.

1 INTRODUCTION

1.1 Motivation

Cluster validation techniques in bioinformatics gained some attention in the recent years. With the widespread application of clustering to the post-genomic data analysis, a thorough validation of the results became necessary. To fill the void, classical validation measures from the data-mining literature have been carried over to bioinformatics (Handl *et al*., 2005), as well as new validation indexes, both internal (Datta and Datta, 2003) and external (Datta and Datta, 2006), have been developed specifically for the microarray data clustering assessment.

Choosing a suitable clustering method, an appropriate measure of dissimilarity (similarity), and a reasonable number of clusters for a particular data at hand remain today the biggest challenges of the clustering process. Our expectation that with the help of validation techniques we would be able to alleviate, if not eliminate, some of those challenges, faced further complications. When applied to the real-world data sets, the results from multiple validation measures are inconclusive, to say the least. Often times they are utterly contradictive. Optimized to assess a certain aspect of a partitioning, both clustering algorithms and cluster validation techniques exhibit biases towards the particular property they optimize. Since none of the clustering algorithms performs uniformly best under all scenarios, it is not enough to use a single algorithm and/or a single validation measure when the real underlying structure of data is unknown which is true in many cases. The most recent recommendations for performing cluster analysis (Handl *et al*., 2005) are to use a number of algorithms and validation measures that optimize different aspects of a partitioning (compactness, connectedness and spacial separation) on the appropriate range of cluster sizes. An objective visual analysis of the results, however, is nearly impossible when the number of validation measures is large.

The goal of multiobjective clustering is to identify a partitioning that, in some sense, would be optimal with respect to several objective functions. A number of different approaches have been proposed in the literature. Law *et al*. (2004) suggest to combine the results from multiple clustering algorithms that optimize conceptually different objective functions into a single partitioning (*post*-clustering optimization). Handl and Knowles (2005; also see Handl and Knowles, 2004), on the other hand, suggest a new clustering algorithm called MOCK (multiobjective clustering with automatic determination of the number of clusters), which identifies multiple trade-off solutions optimized with respect to two objective functions from the Pareto front *during* the clustering process and has the ability to automatically determine the number of clusters.

*To whom correspondence should be addressed.

In this article, we introduce a new technique that can select an optimal algorithm amongst a collection of clustering algorithms that the user is considering. It is based on a weighted rank aggregation that allows us to combine the results from multiple validation measures in a comparative analysis of a given collection of clustering algorithms. Thus, we do not introduce a new algorithm but provide an automatic and objective evaluation of a set of clustering algorithms all of which are considered to be potentially useful by the user.

1.2 Related work

Lin *et al*. (2006) used two cross-entropy Monte Carlo methods to combine the lists of MicroRNAs (miRs) targets predicted by three different algorithms. Datta and Datta (2003) and Datta and Datta (2006) together introduced five novel cluster validation measures for gene expression data which we consider here. An excellent account of traditional cluster validation measures is given in Handl *et al*. (2005). We use five of those in this work.

1.3 Outline and summary

In the System and Methods section, we provide descriptions of the four gene expression data sets (1 simulated, 3 real) along with their corresponding GO annotations. Brief discussions of the clustering methods and the cluster validation measures used to analyze the performance of the algorithms on these data sets follow. Weighted rank aggregation approach of cluster validation measures is considered in the Algorithm and Implementation section where some background information on the cross-entropy Monte Carlo algorithm is given. The Results section summarizes our findings of consolidating ranks from different validation indexes using the algorithm described in the previous section for each of the data sets. The article ends with a general discussion of rank aggregation and our overall conclusions.

2 SYSTEM AND METHODS

2.1 Data sets

We consider one simulated and three real data sets in this article. In each case, we also have a reference set of functional classes to calculate certain biological validation indices (external measures) that we use for evaluation.

2.1.1 Simulated data The simulated data set contains 300 'genes' with seven time points. Their (log-transformed) temporal profiles were assumed to follow a multivariate normal distribution which were generated using *mvrnorm* function available in the R (http://www.r-project.org/) library *MASS*. With all means set to 0, the variance structure reflected the existence of six distinct classes of same size in the data. Correlations between any two profiles not in the same class were set to 0.1, while correlations within each individual classes were gradually decreasing by ~0.01 starting from 0.8 as one was moving away from the main diagonal of the variance–covariance matrix. The entries of the main diagonal were set to 1.

For the construction of the reference set of functional classes, 10 out of 50 'genes' in each group were arbitrarily selected whose class membership information were made known to reflect partial biological information.

2.1.2 Mouse data The mouse data which is publicly available at http://hugheslab.med.utoronto.ca/Zhang/, consists of 41 699 genes whose mRNAs levels were measured in 55 mouse tissues (hybridization repeated twice for each tissue). For our illustration, we selected a subset of 1000 genes that had the largest mean squared errors across profiles. Only 16 out of 55 tissues (110 in total since all are duplicated) that did not have missing values were selected. Using these 16 dimensional expression profiles, 1000 genes were clustered using the 11 clustering algorithms.

Zhang *et al*. (2004) provide two sets of functional annotations for this data, GO and SuperGO. Using the GO annotations, we were able to annotate 422 genes (SuperGO provided a smaller number) out of 1000 that we chose. The following functional classes were constructed with corresponding sizes in parentheses: Behavior (5), Biogenesis (28), Bone Remodelling (4), Cell Activation (2), Cell Adhesion (32), Cell Cycle (13), Cell Death (11), Cell Growth (3), Cell Motility (14), Development (56), Differentiation (8), Excretion (5), Immune System (31), Metabolism (117), Regulation (5), Reproduction (3), Response (47), Transport (38).

2.1.3 Sporulation data This is a well known data set collected and analyzed by Chu *et al*. (1998) that records mRNA levels at seven different time points during the sporulation process in budding yeast. For our purposes, only a subset consisting of 513 genes that were positively expressed is analyzed. A gene was considered to be positively expressed if $\sum \log R > 0$ where the sum is taken over all time points.

To obtain a set of functional classes, we used the FunCat webtool available at http://fatigo.bioinfo.cipf.es/. Out of 513 genes 503 were annotated into the following 16 functional classes: metabolism (138), energy (27), cell cycle and DNA processing (152), transcription (50), protein synthesis (10), protein fate (72), protein with binding function or cofactor requirement (81), protein activity regulation (16), transport (63), cell communication (12), defense (36), interaction with environment (33), cell fate (17), development (13), biogenesis (77) and cell differentiation (82). The same reference set of functional classes was used in (Datta and Datta, 2006).

2.1.4 Breast cancer data Human breast cancer progression data set consists of 258 genes (SAGE tags) that were considered to be differentially expressed at 5% significant level between four normal and seven ductal carcinoma in situ (DCIS) samples. Abba *et al*. (2004) provide further details in their article.

Functional classes were successfully mined from the AmiGo web-based tool available at http://www.godatabase.org/cgi-bin/amigo/go.cgi for 113 genes. The following 11 classes were obtained: biogenesis (24), transport (7), cell communication (15), cellular metabolism (48), cell cycle (6), cell motility (7), immune response (7), cell death (7), development (5), cell differentiation (5) and cell proliferation (5).

2.2 Clustering methods

In this work, a clustering technique is referred to as a clustering method; if it is used with more than one dissimilarity measure, each one is called an algorithm. For a clustering technique that does not use a general dissimilarity, the terms 'method' and 'algorithm' are synonymous. A total of eleven clustering algorithms consisting of seven different clustering methods were considered in this study. Many of the clustering methods are available in the R base distribution or the *cluster* package. R package *kohonen* provides the *SOM* method and *mclust* package provides the Model-based method. Our own implementations of SOTA was developed which is available upon request. The methods UPGMA, Diana, PAM and SOTA can be used with a general dissimilarity measure; we used both Euclidean and (Pearson's) correlation-based dissimilarities.

2.2.1 UPGMA Unweighted pair group method with arithmetic mean (Sneath and Snokal, 1973) is probably one of the most frequently encountered clustering methods. It owes its widespread popularity to its conceptual simplicity and software availability. Over the years, researchers successfully applied UPGMA to many different situations, both biological and non-biological in nature.

2.2.2 K-Means This is another classical clustering method many consider when clustering data (Hartigan and Wong, 1979). It is a partitioning technique that uses the within-class sum of squares as a criterion. An initial set of cluster centers needs to be provided or randomly generated which, of course, requires to set the number of clusters to be generated beforehand.

2.2.3 Diana Diana is a divisive hierarchical method that iteratively splits clusters into two smaller ones until a desired number of clusters is generated or each cluster contains a single observation (Kaufman and Rousseeuw, 1990). Diana is one of a few representatives of the divisive hierarchical approach to clustering (SOTA is another one) as most hierarchical methods are agglomerative. In addition, it also provides the divisive coefficient which is a measure of how much structure was found in the process (many methods do not provide a measure of the significance of the results).

2.2.4 PAM Partitioning around medoids is considered to be a more robust version of K-Means (Kaufman and Rousseeuw, 1990). In many respects, these two methods are very similar.

2.2.5 Model-based clustering Under this approach, a statistical model (mixtures of normals) is fit to the data using the maximum likelihood method (EM algorithm) (Banfield and Raftery, 1993).

2.2.6 SOM This is also a popular method amongst computational biologists and machine-learning researchers. Self-organizing maps uses neural network settings under which numerous observations compete for the current object and the one whose weight vector is closest wins. The winner and its neighbors learn by having their weight vectors readjusted according to pre-specified rules (Kohonen, 1997).

2.2.7 SOTA Self-organizing tree algorithm is an unsupervised network with a binary tree topology (Herrero *et al*., 2001). It combines the advantages of both hierarchical clustering and SOM. It has much better running times than UPGMA when the number of items (genes) is over 600. In all our analysis, SOTA is used with the neighborhood parameter set to zero.

2.3 Validation measures

An excellent summary of different types of validation measures can be found in Handl *et al*. (2005). Based on the classification discussed in the article, we select at least one validation measure from each category: for assessing compactness and separation properties of a partitioning, Dunn Index and Silhouette Width are used; for assessing connectedness, Connectivity is selected; and for predictive power assessment, the Stability measure is chosen which is supplemented by Figure of Merit (Yeung *et al*., 2001) and three other stability measures developed by Datta and Datta (2003). Our list also contains two novel external (biological) measures developed by Datta and Datta (2006).

2.3.1 Dunn index Dunn index is a ratio of the smallest cluster distance to the largest intra-cluster distance, where the smallest cluster distance is defined as the minimum distance between two observations that belong to different clusters (Dunn, 1974). This index is limited to the interval $[0, +\infty]$ and should be maximized.

2.3.2 Silhouette width Silhouette width is a combination measure that considers both intra and inter cluster distances (Rousseeuw, 1987). It takes values in the interval $[-1, 1]$ and should be maximized.

2.3.3 Connectivity Connectivity captures the degree to which observations are connected within a cluster by keeping track of whether the neighboring observations are put into the same cluster (Handl *et al*., 2005). The closer the neighbor that is put into a different cluster, the greater the penalty that is incurred. Neighborhood size (=10 in our analysis) is a required parameter. This index should be minimized.

2.3.4 Stability Under this scheme, the data is randomly divided into two equal groups on which clustering algorithm is separately used. Cluster assignments from the first (training) group are then used to predict cluster assignments for the items in the second (test) group on the basis of the nearest-neighbor classifier. The correspondence between the two assignments is reflected with this stability index which, in our case, was the average over 20 such divisions. This measure should be maximized on the interval [0, 1] (Handl *et al*., 2005).

2.3.5 FOM Figure of merit is another stability-based measure that estimates the predictive power of a clustering algorithm (Yeung *et al*., 2001). Small values for FOM are desirable.

2.3.6 Average proportion of non-overlap The idea behind this and the next two stability-based measures is that a clustering algorithm should produce consistent results even when one deletes one of the dimensions (e.g. time points, replicates, etc.) from the original data set. Average proportion of non-overlap computes the average proportion of genes that are grouped into different clusters based on the complete data and the reduced data (one column missing). The smaller the values of this measure, the more stable the algorithm seems to be (Datta and Datta, 2003).

2.3.7 Average distance between means This measure computes the average distance between the mean expression ratios of all genes clustered together based on the complete and reduced data (Datta and Datta, 2003). It should be minimized.

2.3.8 Average distance Average distance measure computes the average distance between all genes clustered together on the basis of the complete and reduce data (Datta and Datta, 2003). This measure also should be minimized.

2.3.9 BHI Biological homogeneity index is a measure of how biologically homogeneous clusters are based on the available (usually incomplete) biological information. This is an external validation measure because previous knowledge is necessary to compute the index (Datta and Datta, 2006). It should be maximized.

2.3.10 BSI Biological stability index is a measure of consistency with regard to previous biological knowledge when the complete and reduced (by one dimension) data are used in clustering (Datta and Datta, 2006). This index, just like BHI, should also be maximized.

3 ALGORITHM AND IMPLEMENTATION

When evaluating algorithms with the above 10 validation measures, we obtain 10 different ordered lists of 11 clustering algorithms for each total number of clusters under consideration. One can make an attempt to visually analyze them with the goal of finding the best algorithm for a particular data. A completely objective evaluation, however, is not possible with this traditional approach.

Thinking in the direction of discovering a super-list that would be simultaneously as 'close' as possible to the 10 lists

produced by each validation measure, we can formalize our goal within the framework of the following minimization problem. Find δ^* such that

$$\delta^* = \arg\min \Phi(\delta), \quad (1)$$

where $\Phi(\delta) = \sum_M d(\delta, L_M)$ in which L_M is an ordered list produced by validation measure M, d is an appropriate distance function, and the minimization is carried over all possible ordered lists δ of size $k = |L_M|$.

We use Spearman's footrule distance (Fagin *et al.*, 2003) to this end. However, we modify it by including certain weights to reflect how varied the scores are under validation measure M for different ranked positions. Also, we consider the slightly more general problem of combing the top k lists of n clustering algorithms under multiple validation measures M, where $k \leq n$.

Let $L_M = \{A_1^M, \ldots, A_k^M\}$, where $k \geq 1$, denote an ordered list of top k algorithms produced by the validation measure M. Let $M(1), \ldots, M(k)$ be the scores for the top k algorithms in L_M, where $M(1)$ is the best score given by measure M and so on. Let $r^M(A)$ be the rank of A under M (1 means "best") if A is within top k, and be equal to $k+1$, otherwise; $r^\delta(A)$ is defined likewise. The weighted Spearman's footrule distance between L_M and any ordered list δ of k algorithms is given by

$$d(\delta, L_M) = \sum_{t \in L_M \cup \delta} |M(r^\delta(t)) - M(r^{L_M}(t))| \times |r^\delta(t) - r^{L_M}(t)|.$$

One can intuitively think of this distance as a penalty for moving algorithm t from one position to another within the list M(second term of the products) which is adjusted by the difference in scores between the two positions (first term). We also normalize the scores under each measure before computing d.

Having defined the distance for our minimization problem, we can easily solve it when the number of items in our lists is not large or, in case, only a k-top combined list is needed where $k < n$. Since the cardinality of the set of all possible solutions is $n!/(n-k)!$, a brute force approach is certainly feasible for small k's. To solve this combinatorial problem for larger k's, we use cross-entropy Monte Carlo algorithm originally proposed by Rubinstein (1997) for estimating probabilities of rare events in complex stochastic networks and soon after extended to solving difficult combinatorial optimization problems (Rubinstein, 1999, 2001). The algorithm is suitable for solving the minimization problem of finding the optimal list δ^*.

3.1 Cross-entropy (CE) Monte Carlo algorithm

Before we present the algorithm, a short introduction of notation is necessary. Let $(X)_{n\times k}$ be a random matrix whose entries are 0 or 1 with the constraints of its columns summing up to 1 and its row summing to at most 1. Under this setup, each realization of X, x, uniquely determines an ordered list of size k by the position of 1's in each column from left to right (Lin *et al.*, 2006). For example, if the full list was (A, B, C), a 3×2 matrix

$$x = \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}$$

would translate into a candidate top 2 list of (C, A). We introduce a stochastic search algorithm to find an x^* that corresponds to an optimal δ^* satisfying (1).

Assume that the random matrix X that has a probability mass function $P(x)$ that is indexed by the parameter matrix $(v)_{n\times k} = ((p_{jr}))$ as follows:

$$P_v(x) \propto \prod_{j=1}^{n}\prod_{r=1}^{k}(p_{jr})^{x_{jr}} \times I\left(\sum_{r=1}^{k} x_{jr} \leq 1, 1 \leq j \leq n; \sum_{j=1}^{n} x_{jr} = 1, 1 \leq r \leq k\right).$$

Note that any realization x of X will satisfy the above two conditions and hence would correspond to a size k ordered list.

CE Monte Carlo algorithms is a 2-step 'simulate-update' iterative procedure (De Boer *et al.*, 2005; Rubinstein and Kroese,2004):

(i) Generate a random sample from $P_v(x)$.
(ii) Update parameters v based on the drawn sample to produce a 'better' sample in the future which is concentrated around an x^* that corresponds to an optimal δ^*.

Next, we present a more detailed account of the procedure in our context. The procedure has a number of user selectable parameters N (a positive integer), $\rho \in [0, 1]$, $w \in [0, 1]$ and $\epsilon \in [0, 1]$ whose selection is discussed following the description of the algorithm.

(1) **Initialization:** set $t = 0$. Set the initial parameter matrix v^0 of the random distribution of X with constant entries; i.e. let each $p_{jr}^0 = 1/n$. Thus, at this stage, each of the n clustering algorithms has equal chance of being included in the lists of k algorithms for which the objective function Φ will be evaluated.

(2) **Sampling:** at stage t, draw a sample of size N from $P_{v^t}(x)$. Find the corresponding k lists and the values of the objective function Φ, $\Phi(\delta_i), 1 \leq i \leq N$. Sort the $\Phi(\delta_i)$'s in ascending order, say, $\Phi_{(1)} \leq \cdots \leq \Phi_{(N)}$ and find a ρ-quantile, $y^t = \Phi_{([\rho N])}$, where $[a]$, for any real number a, is the integer part of a.

(3) **Updating:** update the parameter vector as follows

$$p_{jr}^{(t+1)} = (1-w)p_{jr}^t + w\frac{\sum_{i=1}^{N} I(\Phi(\delta_i) \leq y^t)x_{ijr}}{\sum_{i=1}^{N} I(\Phi(\delta_i) \leq y^t)},$$

where x_{ijr} is the value at the jrth position of the ith sample and w is a weight parameter introduced to avoid convergence to a local maxima.

(4) **Convergence:** if $||v^{t+1} - v^t|| < \epsilon$, then stop iterating in which case $\Phi_{(1)}$ is taken to be the minima of the objective function and the corresponding ordered subset is the optimal k list, otherwise go back to Sampling. We define $||v^{t+1} - v^t||$ as

$$||v^{t+1} - v^t|| = \frac{1}{nk}\sum_j\sum_r |p_{jr}^{t+1} - p_{jr}^t|.$$

In the Sampling step, we use a Markov chain Monte Carlo (MCMC) procedure to generate samples from $P_v(x)$. It is a sequential procedure (a Markov chain) where at each iteration b, a proposed matrix $x_{n\times k}$ is generated by a random shuffling of 1's in the current realization of the Markov chain $x^{(b)}$ subject to the two constraints discussed above. It is then accepted as the next realization $x^{(b+1)}$ of the Markov chain if the acceptance probability p_A defined by

$$p_A = \frac{P_v(x)}{P_v(x^{(b)})} = \prod_{r=1}^{k}\prod_{j=1}^{n}(p_{jr})^{x_{jr}-x_{jr}^{(b)}}$$

is greater than u, where u is another (uniform) random number in (0, 1) that is independently generated; otherwise, x is rejected and we let $x^{(b+1)} = x^{(b)}$.

The CE algorithm requires users to set a number of parameters. Convergence to a global optimal solution in many ways depends on the parameters chosen. It is recommended that the number of samples N is to be set to at least $10k^2$ and the rarity parameter ρ is to be set to 0.01 if N is relatively large or 0.1 if N is small (<100). The weight parameter w has somewhat a lesser impact on the convergence and values from 0.25 to 0.75 are common. The ϵ parameter should be small; a value of 10^{-4} was used by us.

3.2 Standardization of validation scores

Validation measures employed in this work return scores on different scales and, thus, proper standardization is necessary to equalize the contribution of each measure to the calculation of the objective function. We tried numerous standardization schemes to this end and decided to use a simple transformation of the form: $(x - \min(x))/\max(x - \min(x))$, where x is either a matrix (if multiple number of clusters considered) or a vector (if a number of cluster is fixed beforehand) of scores returned by a particular measure. The main reason for picking this transformation over others lies in its ability to spread out standardized scores over the whole interval [0, 1] which ensures equal importance of each measure in the aggregation process. Aggregation tables using other standardization techniques can be found on the Supplementary Material.

4 RESULTS

For each of the 11 algorithms, we compute the 10 validation measures over an appropriate range of cluster sizes. Parameters for the CE algorithm were selected as follows: the number of samples generated N was set to 2000, the weight parameter $w = 0.5$ and the rarity parameter $\rho = 0.01$. Throughout, *UE* stands for UPGMA with the Euclidean distance, *UC* for UPGMA with the correlation distance, *K* for K-Means, *DE* for Diana with the Euclidean distance, *DC* for Diana with the correlation distance, *PE* for PAM with the Euclidean distance, *PC* for PAM with the correlation distance, *M* for model-based, *SM* for SOM, *SE* for SOTA with the Euclidean distance and *SC* for SOTA with the correlation distance.

We begin the presentation of our findings with the simulated data. We considered from four to eight clusters, but for the sake of brevity, only illustrate the performance of the algorithms for the six clusters which is a 'true' cluster structure by design. Figure 1 shows the standardized scores for the 10 validation measures (similar plots are available on the Supplementary Material illustrating different standardization techniques that can also be used). Scores closer to one indicate a better performance under the measure. Careful visual inspection of the plot seems to indicate that *SOM* and *Model-based* algorithms perform quite well. It is not as trivial, however, to see which one of these two algorithms performs better overall. Even in this relatively simple situation, obtaining aggregated ranks of all 11 algorithms by visual inspection alone is quite a difficult task. The proposed CE Monte Carlo rank aggregation approach greatly simplifies and automates the process of obtaining the holistic picture of the clustering results, providing us with a list of algorithms objectively ordered according to the weighted Spearman's footrule criteria. The combined list returned by the CE algorithm in this particular situation (considering six clusters) ranked from best to worst is: (*M*, *SM*, *K*, *PE*, *UE*, *DE*, *SE*, *PC*, *UC*, *SC*, *DC*). *Model-based* algorithm does outperform its rival *SOM*.

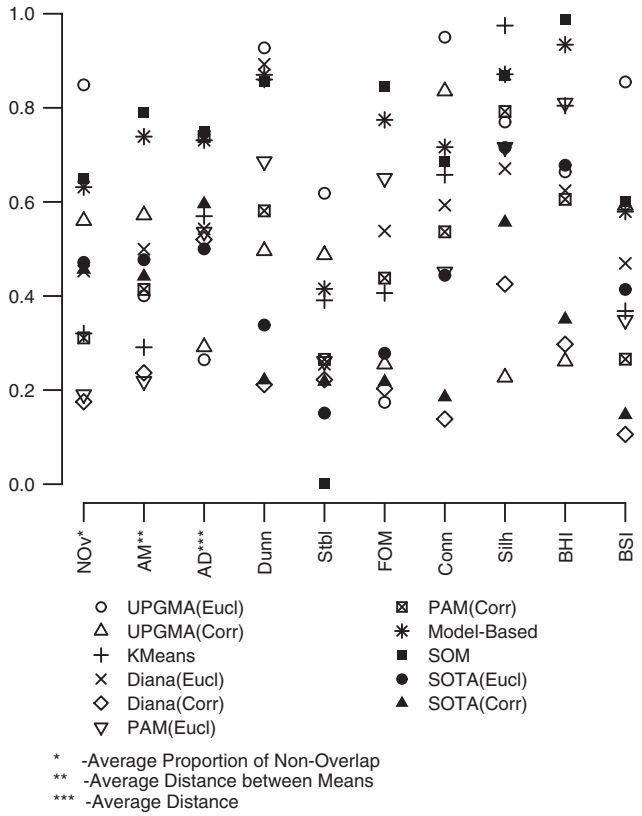

Fig. 1. Standardized scores for the simulated data (six clusters). Large scores correspond to a better performance under a measure. Here, based on the 10 validation measures (*x*-axis), it is very difficult to identify the winner or rank the algorithms from best to worst by visual inspection alone. None of the algorithms performs well under all measures.

Next, we would like to look at a more complicated and general case when the true number of clusters in the data is unknown. Mouse data is a good candidate to consider. Table 1 summarizes the rankings produced by the CE algorithm for the mouse data. For each number of clusters under consideration, a separate combined list is given with the overall ranking at the bottom of the table. Each row in Table 1 represents the aggregation over the 10 validation measures for a particular number of clusters and the overall ranking represents the aggregation over the validation measures and the number of clusters. For the mouse data, e.g. the overall ranking is the ordered list δ^* which minimizes the objective function $\Phi(\delta) = \sum_M d(\delta, L_M)$, where L_M consists $10 \times 7 = 70$ ordered lists. Here, each validation measure is represented by seven ordered lists (one for each number of clusters).

In Figure 2, we show five validation measures selected from different categories (internal, stability-based and biological) and plot their standardized scores against the number of clusters. One immediately begins to appreciate the complexity of each individual plot. To pick an algorithm that performs best considering these five measures is nearly impossible. The Supplementary Material offers five more plots for the other five validation measures which makes any attempts to visually analyze them hopeless. Inspecting Table 1, we see that *UPGMA* with the correlation distance is the overall winner. *PAM*, with correlation, is ranked second and also seems to be a good choice for this data.

It is worth mentioning here that if it is of interest to find a top k list where $k < n$, let's say a top five list, e.g. it is possible to do so using the same algorithm with a parameter $k = 5$. We obtain a top five list for the mouse data and compare it to the top five algorithms from the complete list (the last row of Table 1). The top five list ranked from best to worst is the following: (PC, UC, PE, K, SC). The two lists are somewhat different which is due to using partial information (the first five positions) when constructing a top five list. It is preferred to use complete information if possible but when n is very large, computational considerations come into play; in such cases, it may be more practical to only consider a top k list for a k smaller than n.

Rankings of the 11 algorithms applied to the yeast data are summarized in Table 2. It appears that *UPGMA* with the Euclidean distance is the overall winner in this case. *UPGMA* with the correlation distance, and *SOTA* with the Euclidean distance capture the next two spots. For the cancer data, *UPGMA*, *Diana* and *SOTA* all with the Euclidean distance perform reasonably well. Table 3 shows the complete rankings.

Table 1. Rankings of various clustering algorithms for mouse data

	Ranks (1 is best)										
	1	2	3	4	5	6	7	8	9	10	11
6 Clusters	UC	K	PC	PE	SC	UE	DC	M	SM	SE	DE
7 Clusters	PC	UC	K	SC	DC	SM	PE	M	UE	SE	DE
8 Clusters	PC	K	PE	SC	UC	DC	SM	M	UE	SE	DE
9 Clusters	PC	UC	K	PE	M	SC	UE	DC	SM	SE	DE
10 Clusters	UC	K	PC	PE	DC	UE	SC	M	SM	DE	SE
11 Clusters	UC	PE	PC	K	DC	UE	SC	M	SM	DE	SE
12 Clusters	UC	PE	PC	DC	K	UE	M	SC	SM	DE	SE
Overall	UC	PC	K	PE	SC	DC	UE	M	SM	DE	SE

The graphs of all 10 validation measures for the mouse, yeast, cancer and simulated data are available on the Supplementary Material. Additional aggregation tables similar to Table 1–3 computed using different standardization methods are also available on the Supplementary Material
. With reasonable transformations (see Supplementary Material for examples) $\Re^+ \to [0, 1]$, ranks returned by the CE algorithm do not differ substantially.

The CE algorithms took 10–15 iterations to converge in each case. Everything else remaining equal, the number of iterations will depend on ϵ in the convergence criterion which we set to 10^{-4}.

5 DISCUSSION

The problem of rank aggregation is certainly not new. Back in 1770, a French mathematician and nautical astronomer Jean-Charles de Borda, proposed the solution based on the average position of items in the lists. It found its famous application in the voting theory where each candidate's overall rank was a simple average of the number of candidates beaten by him/her over all voters' rankings. This simple solution, however, had a major drawback in a sense that a candidate with the largest number of pairwise wins could lose the race. Here's a small example that illustrates the situation in which this can happen. Let's assume that three candidates A, B and C have been ranked by five different voters as in the Table 4.

Using Borda's method, we would obtain the combined ranking (B, A, C) where B is ranked ahead of A despite the fact that A is preferred to B more often. Spearman's footrule distance is minimized when the overall ranking is: (A, B, C) and declares A to be a winner, which is in agreement with the principle that a candidate with the largest number of pairwise wins should win (also known as a Condorcet criterion). In a case of a complete symmetry when each candidate beats the same number of his/her adversaries, Borda's method in its original form is not very helpful in determining the winner.

The voting example is inherently dichotomous and in that sense is different from the settings of ranking clustering algorithms where validation scores provide additional information that can be integrated into the ranking process. Incorporating the weights into the Spearman's footrule distance turned out a very useful and, in many examples, a necessary property for the convergence of the CE algorithm. When two candidate lists have the same value of the objective function, the CE algorithm struggles in choosing between the two, taking a long time to converge arbitrarily to one of them. Weights almost eliminated the possibility of such a tie and made the solution unique. Let us consider an example. Assume that the three lists with corresponding weights are given as in Table 5. Combining them without considering weights produces two distinct aggregated lists with the same values of the objective function equal to 18, (D, A, F, E) and (A, D, F, E). Obviously, the winner cannot be determined in this situation. When the weights are used, however, the solution, (D, A, F, E), is unique with a unique minimum value of the objective function of 6.75.

D does beat A and, in this case, one can confirm that without much difficulty by looking at Table 5. This example clearly demonstrates the advantages of using the weighted rank aggregation introduced in this article over a conventional one that uses ranks alone.

Researchers may choose to have a greater control over the aggregation process by weighing different validation measures differently. This can be easily done by introducing the weight function $W(M)$ into our objective function $\Phi(\delta) = \sum_M W(M)d(\delta, L_M)$. As for example, if multiple validation indices that measure similar characteristics of a clustering algorithm are included for rank aggregation, each can be down-weighted by the inverse of the total number of such similar indices.

It is important to remember that clustering results should be scrutinized and checked before running the CE algorithm to avoid potential biases and pitfalls. *Fanny*, a fuzzy clustering method, which we initially considered for this study, was consistently returning a smaller number of hard clusters than anticipated. We then had to remove it because its performance was biased in a positive direction due to validation measures' sensitivity towards the number of clusters. Handl *et al*. (2005) discuss this in more details in their article. A trivial example of a validation measure being effected by the increasing number of clusters would be connectivity which is minimized (is zero) when only one cluster is considered and tends to increase when the number of clusters in a partitioning becomes larger. Another issue one can potentially encounter when clustering

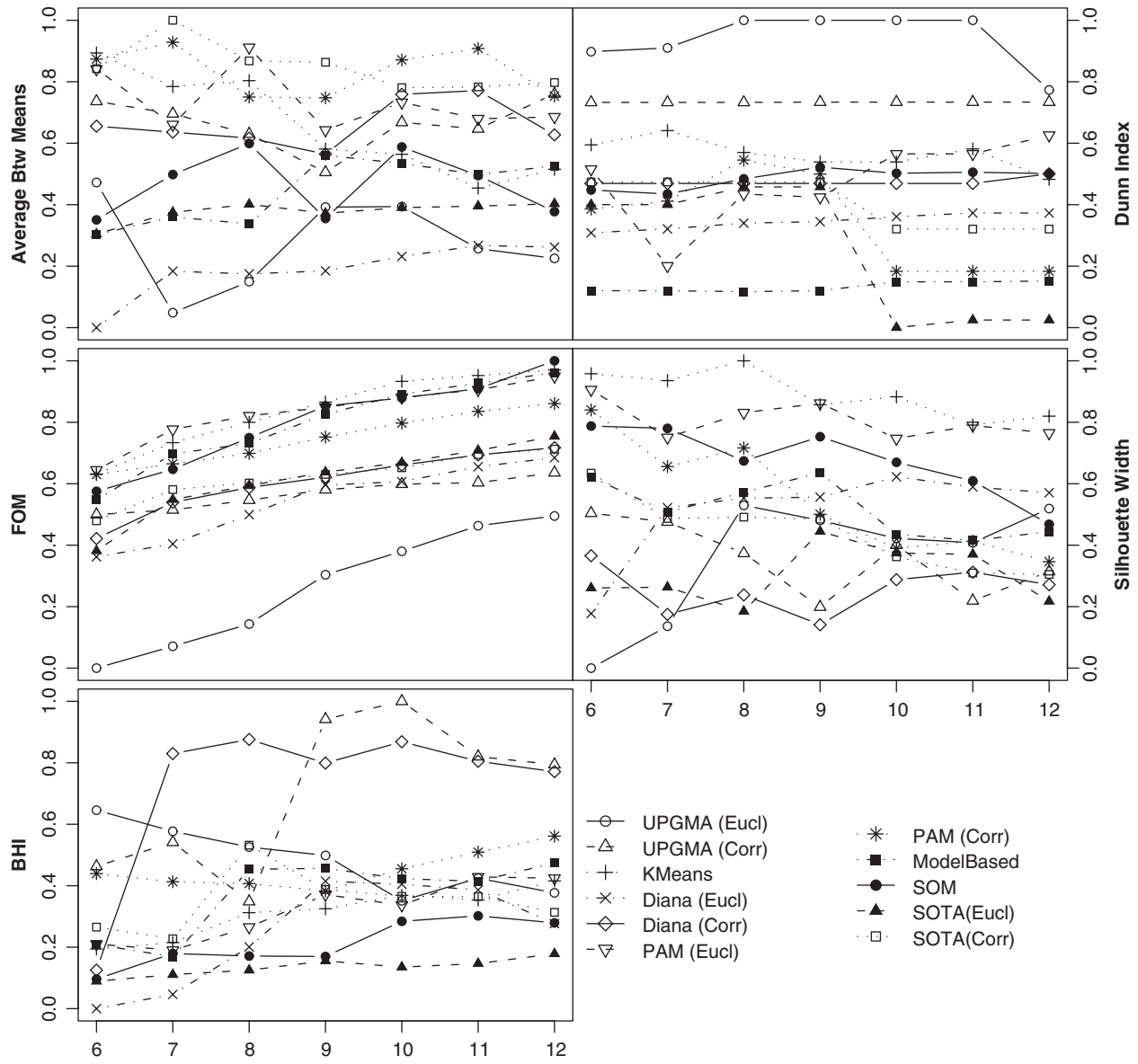

Fig. 2. Standardized cluster validation scores for various cluster validation measures across a range of different number of clusters (mouse data). Again, large scores imply better performance. Without knowing the true number of clusters in a given data set, multiple 2D plots have to be considered simultaneously to judge the performance of algorithms. It is virtually impossible to analyze these plots visually.

Table 2. Rankings of various clustering algorithms for yeast data

	Ranks (1 is best)										
	1	2	3	4	5	6	7	8	9	10	11
6 Clusters	UE	K	DE	SE	SM	PE	M	SC	UC	DC	PC
7 Clusters	SM	UE	DE	K	SE	PE	M	UC	SC	DC	PC
8 Clusters	UE	UC	SE	DE	PE	K	SM	SC	DC	PC	M
9 Clusters	SC	UC	UE	SE	PE	K	DE	SM	DC	M	PC
10 Clusters	SE	UC	UE	M	DE	SM	K	SC	PE	DC	PC
11 Clusters	SE	UC	UE	M	DE	K	SM	PE	SC	DC	PC
12 Clusters	UE	SE	UC	DE	SM	M	K	PE	SC	DC	PC
Overall	UE	UC	SE	DE	SM	K	M	PE	SC	DC	PC

Table 3. Rankings of various clustering algorithms for cancer data

	Ranks (1 is best)										
	1	2	3	4	5	6	7	8	9	10	11
4 Clusters	UE	DE	SE	SM	PE	DC	K	UC	M	SC	PC
5 Clusters	UE	DE	SE	K	PE	SC	SM	UC	DC	M	PC
6 Clusters	UE	DE	SE	PE	K	SM	UC	SC	DC	M	PC
7 Clusters	DE	UE	SE	PE	K	SM	UC	DC	SC	M	PC
8 Clusters	DE	UE	SE	PE	SM	K	UC	SC	M	DC	PC
9 Clusters	UE	DE	SE	PE	SM	UC	M	SC	K	DC	PC
10 Clusters	UE	DE	SE	PE	SM	UC	K	SC	M	DC	PC
Overall	UE	DE	SE	PE	SM	K	UC	SC	M	DC	PC

Table 4. Rankings of the three candidates by five voters

	Ranks		
	1	2	3
Voter 1	A	B	C
Voter 2	A	B	C
Voter 3	A	B	C
Voter 4	B	C	A
Voter 5	B	C	A

This voting example illustrates the differences in the rankings obtained using Borda's method versus the aggregation method proposed in this article. Borda's approach declares B to be the winner despite the fact that A beats B three out of five times. Our method prefers A to B.

data is having a partitioning with a single large cluster and a few small clusters (1–2 profiles). Similarly to the previous scenario, a clustering algorithm that returns a partitioning of this type will be favored by many of the validation measures that we discuss in this article.

The advantage of using the proposed approach to obtain a full ranking of clustering algorithms based on their performance evaluated by numerous validation measures is its flexibility. Any computationally reasonable number of different clustering algorithms can be considered with arbitrary

Table 5. Illustration of the advantages of using weighted rank aggregation when scores are available

	Ranks			
	1	2	3	4
List 1	B (0.8)	A (0.75)	D (0.72)	C (0.7)
List 2	E (0.95)	D (0.9)	F (0.85)	A (0.83)
List 3	D (0.8)	A (0.8)	F (0.8)	E (0.6)

While ignoring the scores, no unique winner exists. Incorporating scores into the rank aggregation process helps to break the tie between A and D in favor of D.

combinations of tuning parameters and/or distance functions. In addition, a researcher can decide what and how many validation measures he/she wants to use to sufficiently evaluate the performance of the algorithms on a given data. This decision is subjective and, in many cases, may be data driven. Also, not only the winning algorithm is determined, which often is of primary interest, but a complete ranking of all clustering algorithms is returned, giving researcher the directions in which further exploration of the data should or should not be conducted. Inclusion of the additional clustering algorithm(s) may change the order of other algorithms previously ranked and we have verified that experimentally. This is certainly a disadvantage of the aggregation method, however, changes in the ordering of the algorithms are minimal and, if having place, they usually occur in the middle and tail portions of the aggregated lists.

As with any optimization procedure, care should be taken when using the CE procedure. It is recommended to run the algorithm multiple times with different seeds for the random number generator and tweaking the tuning parameters, if necessary, until a consistent convergence to the optimal value is achieved.

ACKNOWLEDGEMENTS

This research was supported by grants from the National Science Foundation (MCB-0517135) and the National Security Agency (H98230-06-1-0062). Reviewer's valuable comments lead to a better manuscript.

Conflict of Interest: none declared.

REFERENCES

Abba,M.C. *et al.* (2004) Transcriptomic changes in human breast cancer progression as determined by serial analysis of gene expression. *Breast Cancer Res*, **6**, R499.

Banfield,J.D. and Raftery,A.E. (1993) Model-based Gaussian and non-Gaussian clustering. *Biometrics*, **49**, 803–822.

Chu,S. *et al.* (1998) The transcriptional program of sporulation in budding yeast. *Science*, **282**, 699–705.

Datta,S. and Datta,S. (2003) Comparisons and validation of statistical clustering techniques for microarray gene expression data. *Bioinformatics*, **19**, 459–466.

Datta,S. and Datta,S. (2006) Methods for evaluating clustering algorithms for gene expression data using a reference set of functional classes. *BMC Bioinformatics*, **7**, 397.

De Boer,P. *et al*. (2005) A tutorial on the Cross-Entropy method. *Ann. Oper. Res.*, **134**, 19–67.

Dunn,J.C. (1974) Well separated clusters and fuzzy partitions. *J. Cybern.*, **4**, 95–104.

Fagin,R. *et al*. (2003) Comparing top k lists. *SIAM J. Discrete Math.*, **17**, 134–160.

Handl,J. and Knowles,J. (2004) Evolutionary multiobjective clustering. In Yao,X. *et al*. (eds.), *Proceedings of the Eighth International Conference on Parallel Problem Solving from Nature (PPSN VIII)*. Springer-Verlag, Berlin, pp. 1081–1091.

Handl,J. and Knowles,J. (2005) Exploiting the trade-off – the benefits of multiple objectives in data clustering. In Coello,L.A. *et al*. (eds.), *Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization*. Springer-Verlag, Berlin, pp. 547–560.

Handl,J. *et al*. (2005) Computational cluster validation in post-genomic data analysis. *Bioinformatics*, **21**, 3201–3212.

Hartigan,J.A. and Wong,M.A. (1979) A k-means clustering algorithm. *Appl. Stat.*, **28**, 100–108.

Herrero,J. *et al*. (2001) A hierarchical unsupervised growing neural network for clustering gene expression patterns. *Bioinformatics*, **17**, 126–136.

Kaufman,L. and Rousseeuw,P.J. (1990) *Fitting Groups in Data. An Introduction to Cluster Analysis*. Wiley, New York.

Kohonen, T. (1997) *Self-Organizing Maps*. 2nd edn. Springer-Verlag, Berlin.

Law,M. *et al*. (2004) Multiobjective data clustering. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. **Vol. 2**, pp. 424–430.

Lin,S. *et al*. (2006) Rank aggregation of putative microRNA targets with Cross-Entropy Monte Carlo methods. Preprint.

Rousseeuw,P.J. (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, **20**, 53–65.

Rubinstein,R.Y. (1997) Optimization of computer simulation models with rare events. *Eur. J. Oper. Res.*, **99**, 89–112.

Rubinstein,R.Y. (1999) The simulated Entropy method for combinatorial and continuous optimization. *Methodol. Comput. Appl. Probab.*, **2**, 127–190.

Rubinstein,R.Y. (2001) Combinatorial optimization Cross-Entropy, Ants, and rare events. In Uryasev,S. and Pardalos,P.M. (eds.), *Stochastic Optimization: Algorithms and Applications*. Kluwer, The Netherlands, pp. 304–358.

Rubinstein,R.Y. and Kroese,D.P. (2004) *The Cross-Entropy Method. A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer, New York.

Sneath,P.H. and Snokal,R.R. (1973) *Numerical Taxonomy*. Freeman, San Fransisco.

Yeung,K. *et al*. (2001) Validating clustering for gene expression data. *Bioinformatics*, **17**, 309–318.

Zhang,W. *et al*. (2004) The functional landscape of mouse gene expression. *J. Biol.*, **3**, 21.