Pattern Recognition 43 (2010) 3364–3373

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

SEP/COP: An efficient method to find the best partition in hierarchical clustering based on a new cluster validity index

Ibai Gurrutxaga *, Iñaki Albisua, Olatz Arbelaitz, José I. Martín, Javier Muguerza, Jesús M. Pérez, Iñigo Perona

Department of Computer Architecture and Technology, University of the Basque Country, Manuel Lardizabal 1, 20018 Donostia, Spain

ARTICLE INFO

Article history:
Received 8 October 2009
Received in revised form
4 March 2010
Accepted 29 April 2010

Keywords:
Hierarchical clustering
Post-processing
Cluster validity index

ABSTRACT

Hierarchical clustering algorithms provide a set of nested partitions called a cluster hierarchy. Since the hierarchy is usually too complex it is reduced to a single partition by using cluster validity indices. We show that the classical method is often not useful and we propose SEP, a new method that efficiently searches in an extended partition set. Furthermore, we propose a new cluster validity index, COP, since many of the commonly used indices cannot be used with SEP. Experiments performed with 80 synthetic and 7 real datasets confirm that SEP/COP is superior to the method currently used and furthermore, it is less sensitive to noise.

1. Introduction

Clustering is widely used in many fields such as psychology [1], biology [2,3], pattern recognition [4], image processing [5,6] and computer security [7]. The aim any clustering algorithm is to identify natural structures in data [4,8–13]. It is an unsupervised pattern classification method that partitions the input data into groups or clusters. The goal of a clustering algorithm is to perform a partition where objects within a group are similar and objects in different groups are dissimilar.

Many clustering algorithms exist, but they can be broadly classified into two groups: partitional [14,15] and hierarchical [16–20]. Partitional algorithms process the input data and create a partition that groups the data into clusters. In contrast, hierarchical algorithms build a nested partition set called a cluster hierarchy. At the lower level of the hierarchy all the data points are usually clustered in singleton clusters, i.e. clusters composed of just one object or data point. The topmost partition is usually the cluster containing all the data points. The remaining intermediate partitions define the path followed from one end to the other. The procedures to obtain this set of partitions are classified as either agglomerative (bottom-up) or divisive (top-down). The former begins from the singleton clusters and obtains the hierarchy by successively merging clusters. In contrast, the latter begins with a single cluster containing all the points and proceeds by iteratively splitting the clusters.

Although a cluster hierarchy offers more information than a single partition, hierarchies are often too complex to analyse. Therefore, it is usual to post-process a hierarchy to find the best partition in it. The usual approach is to evaluate all the partitions in the hierarchy—or just a subset of them—based on a cluster validity index (CVI) and to select the best one. A CVI is an index that evaluates a partition, usually by analysing the cohesion and separation of its clusters. These types of indices are mostly used to compare different partitions in environments where the actual clusters are unknown [10,21–25]. In some cases, the best partition is selected while the algorithm is still building the hierarchy. In these cases the CVIs are often referred as stopping criteria [26,27].

In this paper we show that this post-processing approach is not valid in many cases because the correct partition is often not explicitly in the cluster hierarchy. Therefore, the correct partition is often ignored in the process. We define a new partition set that can be computed from a hierarchy—the extended partition set—and show that in many cases the correct partition is in this extended partition set and not in the hierarchy. Moreover, we describe an algorithm to efficiently search in the extended partition set of a hierarchy. Furthermore, we show that the common CVIs are not suitable for the search algorithm and consequently, we propose a new cluster validity index.

An experiment carried out with synthetic and real datasets showed that the algorithm we propose, SEP, is better than the usual hierarchy post-processing method and it also showed that the method is robust in noisy environments. Furthermore, it also shows that COP, the cluster validity index we propose to be used

* Corresponding author.
E-mail addresses: i.gurrutxaga@ehu.es (I. Gurrutxaga), inaki.albisua@ehu.es (I. Albisua), olatz.arbelaitz@ehu.es (O. Arbelaitz), j.martin@ehu.es (J.I. Martín), j.muguerza@ehu.es (J. Muguerza), txus.perez@ehu.es (J.M. Pérez), inigo.perona@ehu.es (I. Perona).

0031-3203/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2010.04.021

with SEP, works as well as the better known indices with the usual approach.

In the next section we illustrate the problem with an example and define the notation used in the rest of the paper. Section 3 describes the search algorithm, SEP, and shows why the usual CVIs are not suitable for it. A new cluster validity index, COP, suitable for the search algorithm is described in Section 4. Sections 5 and 6 are devoted to showing the experimental work performed and the results obtained. Finally, we discuss the results in Section 7 and we draw some conclusions in Section 8.

2. Problem definition and notation

In this section we first illustrate the problem we face with a graphical example. Next, we introduce some definitions and the notation used throughout the paper. Finally, we formally state our goal.

2.1. Illustrative example

The usual approach to working with a cluster hierarchy is to represent it as a graphical tree-like structure called a dendrogram. Clusters in the partitions are represented as nodes in the dendrogram. Similarly, branches in the dendrogram represent the merging/splitting of clusters in the hierarchy. Fig. 1b shows the dendrogram corresponding to the hierarchy obtained by applying a well-known agglomerative clustering algorithm to the dataset in Fig. 1a. Obviously, this representation is considerably more helpful for the end user than a nested partition set.

The order of the partitions is represented in the dendrogram by the height of the nodes. Merges produced in the initial steps of the hierarchical algorithm can be found in the lower part of the dendrogram, while the final merge producing a partition with a single cluster is represented at the top of the tree—the root node. Notice that any partition of the hierarchy can be represented in the dendrogram by a horizontal cut. As an example, we added a line cutting the dendrogram in Fig. 1b where the correct partition of four clusters can be found. As we move the line down (up) it will cut partitions with more (less) clusters.

Let us now build a hierarchy and its corresponding dendrogram from a different dataset. This dataset and the corresponding dendrogram are shown in Fig. 2. It can be easily seen that the correct partition of three clusters is not in the hierarchy, since there is no horizontal line that cuts the three correct branches. However, we can easily see that the dendrogram shows the correct partition. We have added a broken line in the figure to show that a non-horizontal cut can specify the correct partition. This example shows that in some cases, although the correct partition is not explicitly in the hierarchy it can be implicitly described by it. In fact, some authors propose not build the whole hierarchy but to prune a subtree when it can be considered a compact cluster [17] and therefore, this would be equivalent to cutting the dendrogram with a broken line.

Nevertheless, the aim of this work is to propose a tool to post-process and help interpreting a complete hierarchy. Moreover, since our tool does not modify the building process of the hierarchy, it can be used combined with many hierarchical clustering algorithms.

Since the usual approach to post-processing a hierarchy only considers horizontal cuts, we argue that in many cases the correct partition will not even be in the search space of the post-processing algorithm. Therefore, we define a new search space containing all the partitions in a dendrogram—those that can be generated with any type of cut—and call it the extended partition set. A search in this space will necessarily produce results that are equal to or better than the results produced by a search in the hierarchy, since all partitions in the hierarchy are also in the extended partition set.

We are aware that it is not feasible to exhaustively search the extended partition set of any practical hierarchy. For instance, the extended partition sets of the two dendrograms shown above contain 4.97×10^{31} and 4.90×10^{23} partitions. In Section 3 we describe SEP, an algorithm that can search in such a space in an efficient way.

2.2. Notation

In this section we formally define some terms and describe the notation used in this paper.

A dataset, $X = \{x_1, x_2, \ldots, x_N\} \in \Re^F$, is a set of N objects or data points represented as feature vectors in a F-space. A cluster, $C \subseteq X$ is a subset of objects in the dataset. The centroid of a cluster, $\overline{C} = (1/|C|)\sum_{x \in C} x$, is the mean vector of all vectors in the cluster.

A partial partition of a dataset, $P^Y = \{C_1, C_2, \ldots, C_K\}$ s.t. $\bigcup_{C_k \in P^Y} C_k = Y, C_k \cap C_l = \emptyset \ \forall k \neq l$ and $Y \subseteq X$, is a set of disjoint clusters of a subset of the dataset. A total partition of a dataset, P^X, is a set of disjoint clusters of the complete dataset. For clarity,

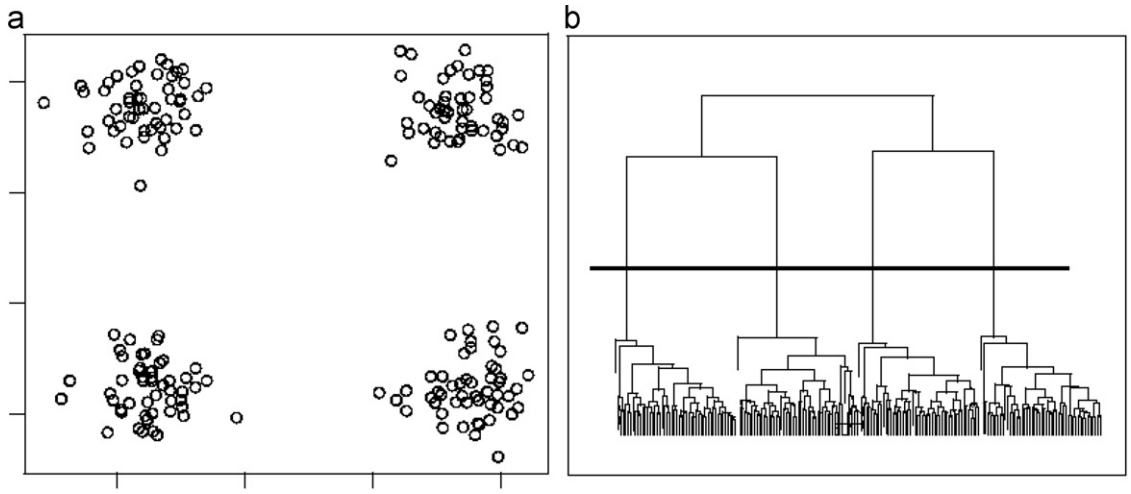

Fig. 1. (a) 2 × 2 dataset. (b) Dendrogram obtained from 2 × 2 dataset.

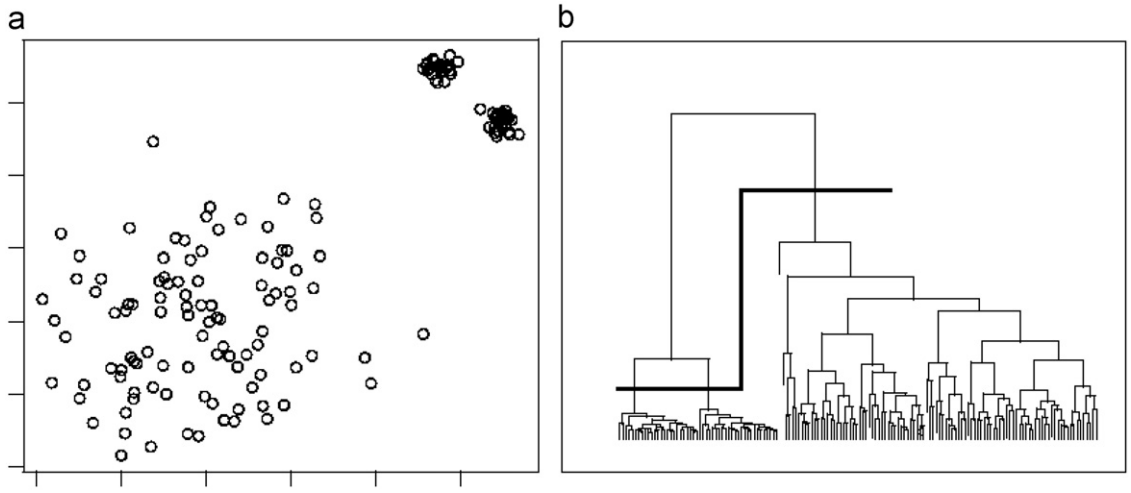

Fig. 2. (a) 1 + 2 dataset. (b) Dendrogram obtained from 1+2 dataset.

a partial partition or total partition will be referred to as "a partition" where it is either irrelevant which kind of partition we are referring to or it is clear by the context. The correct partition of a dataset, P^*, is the partition that describes the natural groups in the dataset: i.e. each cluster in P^* will represent a group of points in the dataset.

A hierarchy, $H=\{P_1,P_2,\ldots,P_R\}\forall P_r,P_s r<s\Leftrightarrow\forall C_k\in P_r,\exists C_l\in P_s$ such that $C_k\subseteq C_l$, is a set of nested partitions where any partition—except the first one—can be described as a set of cluster merges performed in the previous partition in the set. The extended partition set of a hierarchy, $E_H=\{P:P\subseteq T,\bigcup_{C\in P}C=X$ and $\forall C_k,C_l\in P\ C_k\cap C_l=\emptyset\}$ where $T=\bigcup_{P\in H}\bigcup_{C\in P}C$, is the set of partitions that can be built with any combination of clusters found in the hierarchy.

A cluster validity index is a function, V(P), that evaluates a partition. Unless otherwise explicitly stated we will assume that a lower value of V denotes a better partition. Let us define a function that finds the best partition in a partition set as $S(V,H)=\text{argmin}_{P\in H}V(P)$ where V is a CVI and H is a partition set.

In this context the problem we want to solve is the following: given a dataset, X, and a hierarchy, H, find $\hat{P}=S(V,E_H)$, the best partition, according to V, in the extended partition set of H.

3. SEP: an algorithm to search in the extended partition set of a hierarchy

As noted in Section 2.1 an exhaustive search in an extended partition set is not feasible for all but trivial hierarchies. In contrast, an exhaustive search in a hierarchy is usually possible, although, due to the limitations of CVIs, the search is usually restricted to some of the topmost partitions. Therefore, we can more precisely define the usual approach to find the best partition in a hierarchy H as $S(V,H')$ where $H'=\{P:P\in H,\ 1<|P|\leq\alpha\}$ where α is a user defined threshold. Thus, just the partitions containing up to a fixed number of clusters are analysed. It is common practice to set this threshold to $\sqrt{N}$ [21,28] or a fixed value between 10 and 25 [27,29]. This practice allows an exhaustive search since the search space is reduced to a few dozen partitions. Anyway, even the exhaustive analysis of the whole hierarchy is usually feasible since the size of the hierarchy is bounded by N, $|H|\leq N$.

In the following, we describe SEP (Search over the Extended Partition set); an algorithm that finds the best partition in the extended partition set of a hierarchy. SEP is a recursive algorithm described in Algorithm 1.

Algorithm 1. SEP(V,*Node*)

```
1:  C ← cluster in Node
2:  if Node is a leaf node then
3:    return {C}
4:  else
5:    Union ← ∅
6:    for all Child ∈ Node do
7:      Union ← Union ∪ SEP(V,Child)
8:    end for
9:    if V({C}) < V(Union) then
10:     return {C}
11:   else
12:     return Union
13:   end if
14: end if
```

The algorithm compares two partitions (line 9): (a) the partition with just the cluster corresponding to the current node, and (b) the union of the best possible partition in each child node of the current node. The decision about which is the best possible partition in each child node is made recursively (line 7). The recursive calls are repeated through the whole tree until reaching the leaf nodes, where a partition containing just the cluster at the node is always returned as the best one (line 3). In order to analyse the whole tree the algorithm must begin with the root node.

This algorithm makes the search affordable from the point of view of computational cost. The CVI must be computed twice at each node: once for the cluster in the node and once for the union of the best partition of each child. In any case, many indices can easily evaluate the union of some partial partitions combining the individual results for each combined partition. Therefore, the CVI can be usually computed just once for each node. Note that the number of nodes in a dendrogram is bounded by $2N-1$ (N leaf nodes and $N-1$ internal nodes).

The key idea in the algorithm is to analyse each subtree independently. In this way, we can begin at the leaf nodes and go up through the tree deciding locally which is the best possible partition in each subtree. In other words, the algorithm decides which is the best partial partition for the data in each node. Unfortunately, this procedure implies some limitations: (a) the CVI must be able to evaluate partial partitions, even those with just one cluster and (b) the CVI must ensure that once a partial partition is selected as the best one for a particular subtree, it will still be the best one after merging it with another partial partition—

corresponding to different data points. We discuss these limitations below and show that many CVIs are not suitable for SEP.

3.1. Properties CVIs must satisfy

The CVI passed as an input argument to SEP must satisfy two properties:

- *Partiality*: The CVI must be able to evaluate a partial partition. Although CVIs are designed to evaluate total partitions, it is easy to adapt a CVI to evaluate a partial partition: evaluate the partial partition as if it were a total partition. In other words, a CVI can be described as $V(P^X,X)$, but we need a CVI defined as $V(P^Y, X)$. An easy way to adapt a usual CVI would be to redefine it as $V(P^Y,Y)$; i.e. reducing the data it analyses to the data in the partial partition it must evaluate. Unfortunately this solution is not good since the CVI would only "see" the data in the evaluated partial partition, ignoring the rest of the data. In this case the CVI would not be able to evaluate a partition with a single cluster, as needed by SEP. Certainly, the separability concept implicit in most CVIs would hardly be estimated in a single cluster partition if data outside of the cluster could not be considered.
- *Context-independent optimality*: The CVI must ensure that if a partial partition is better than another one, it will be better in any context. We define this property as follows:

  $$V(P_1^Y) < V(P_2^Y) \Leftrightarrow V(P_1^Y \cup P^Z) < V(P_2^Y \cup P^Z), \quad Y \cap Z = \emptyset$$

  In this way, we can combine the best partitions from two subtrees and ensure that there is no better partition for the union of the data in the two subtrees. Note that this property requires the index to be independent of possible clusterings of the data outside of the evaluated partial partition. Therefore, the combination of the two properties requires the CVI to analyse the data outside of the evaluated partial partition, ignoring any possible clustering of it.

3.2. Analysis of some widely used CVIs

Next we describe, following the notation described in Section 2.2, five widely used CVIs. Then, we show that none of them satisfies the partiality property.

- Calinski–Harabasz (CH) [30]:

  $$CH(P) = \frac{(N-|P|)\text{inter}_{CH}(P)}{(|P|-1)\text{intra}_{CH}(P)}$$

  where $\text{inter}_{CH}(P) = \sum_{C \in P} |C| d(\overline{C},\overline{X})$ and $\text{intra}\,CH(P) = \sum_{C \in P} \sum_{x \in C} d(x,\overline{C})$.
- C-Index [31]:

  $$CI(P) = \frac{S(P)-S_{\min}(P)}{S_{\max}(P)-S_{\min}(P)}$$

  where $S(P) = \sum_{C \in P} \sum_{x_i,x_j \in C} d(x_i,x_j)$ is the sum of the distances between all pairs of objects in the same cluster. Let n_w be the number of such pairs of objects. Then, $S_{\min}$ is the sum of the n_w smallest distances over the whole dataset, while $S_{\max}$ is the sum of the n_w largest distances.
- Gamma [32]:

  $$\text{Gamma}(P) = \frac{\sum_{C \in P}\sum_{x_i,x_j \in C} dl(x_i,x_j)}{n_w(N(N-1)/2-n_w)}$$

  where $dl(x_i,x_j)$ denotes the number of object pairs that are more similar than x_i and x_j and are in different clusters. The denominator is used for normalization purposes and ensures that the index will not exceed 1. n_w denotes the number of pairs of objects in the same cluster and N is the number of objects in the whole dataset.
- Davies–Bouldin (DB) [33]:

  $$DB(P) = \frac{1}{|P|}\sum_{C_k \in P} \max_{C_l \in P, k \neq l} \left\{ \frac{S(C_k)+S(C_l)}{d(\overline{C_k},\overline{C_l})} \right\}$$

  where $S(C) = 1/|C| \sum_{x \in C} d(x,\overline{C})$.
- Dunn [34]:

  $$\text{Dunn}(P) = \frac{\text{inter}_{Dunn}(P)}{\text{intra}_{Dunn}(P)}$$

  where

  $$\text{inter}_{Dunn}(P) = \min_{C_k \in P}\left\{ \min_{C_l \in P, k \neq l} \{\delta(C_k,C_l)\} \right\}$$

  $$\delta(C_k,C_l) = \min_{x_i \in C_k}\left\{ \min_{x_j \in C_l} \{d(x_i,x_j)\} \right\}$$

  $$\text{intra}_{Dunn}(P) = \max_{C \in P}\left\{ \max_{x_i,x_j \in C} \{d(x_i,x_j)\} \right\}$$

Note that in the case of the Calinski–Harabasz and Dunn indices a bigger value denotes a better partition.

Now, we will briefly discuss whether or not these indices satisfy the properties required by the SEP algorithm.

- *Partiality*: Although all the indices can be used to process a partial partition, none of them can satisfactorily evaluate a partial partition with a single cluster. Calinski–Harabasz has the term $|P|-1$ in the denominator, so the value of the CVI would be infinity in such a case. In the case of C-Index the terms S, $S_{\min}$ and $S_{\max}$ would be equal, leading to a 0/0 indetermination. The same indetermination would occur with Gamma, since the sum of all $dl(x_i,x_j)$ would be 0 and $n_w = N(N-1)/2$. Finally, Davies–Bouldin and Dunn require the comparison of at least two clusters. In conclusion, none of the five indices described above are suitable for the SEP algorithm.
- *Context-independent optimality*: The analysis of this property is more complex to perform and is not actually needed in this case, since none of the analysed CVIs satisfies the partiality property. In any case, for most of the indices we found counter-examples that show they do not satisfy the context-independent optimality property.

4. COP: a new cluster validity index

We defined a new cluster validity index that satisfies the two desired properties. First we define the new CVI and then describe some of its properties.

4.1. Index definition

We defined a ratio-type index [21] with an estimate of the intra-cluster variance (cohesion) in the numerator and an estimate of the inter-cluster variance (separation) in the denominator. We called it COP because it satisfies the Context-independent Optimality and Partiality properties. We defined the index as

$$COP(P^Y,X) = \frac{1}{|Y|}\sum_{C \in P^Y} |C| \frac{\text{intra}_{COP}(C)}{\text{inter}_{COP}(C)}$$

where

$$\text{intra}_{COP}(C) = 1/|C| \sum_{x \in C} d(x,\overline{C})$$

$$\text{inter}_{COP}(C) = \min_{x_i \notin C} \max_{x_j \in C} d(x_i,x_j)$$

and $d(x_i,x_j)$ is the Euclidean distance between points x_i and x_j. Since COP cannot evaluate the root and leaf nodes, we set its value to 1 in these cases. Thus, it is able to analyse a partial partition, P^Y, using the entire dataset, X, but ignoring any possible clustering of points in X–Y.

The intra-variance estimate is the average Euclidean distance between the points in a cluster and its centroid. The inter-variance estimate is the distance between the cluster and its nearest point, measured by the complete-linkage proximity measure. The complete-linkage proximity measure defines the distance between two clusters as the distance between the furthest points in the clusters [11]. Note that each cluster is evaluated separately and all the partial evaluations are averaged by their weighted mean.

4.2. Properties

COP possesses some interesting properties: it is bounded between 0 and 1 and it satisfies the two properties required to be used in the SEP algorithm. Furthermore, the COP value of a node can be computed by combining the COP values of its children. Therefore, SEP needs to compute COP just once in each node (see Section 3).

It can be shown that $COP \geq 0$ because it is a ratio between positive values—set cardinalities and Euclidean distances. We can also show that $COP \leq 1$ because the denominator is always greater than or equal to the numerator. More precisely $\text{intra}_{COP}(C) + d(\overline{C},y) \leq \text{inter}_{COP}(C)$ where y is the closest point to C, according to complete-linkage proximity measure.

Proof. The average Euclidean distance of all the points in a cluster C, to y, is $(1/|C|)\sum_{x\in C}\sum_{i=1}^{F}(x_i-y_i)^2 = (1/|C|)\sum_{x\in C}\sum_{i=1}^{F}((x_i-\overline{C}_i)+(\overline{C}_i-y_i))^2 = (1/|C|)\sum_{x\in C}\sum_{i=1}^{F}(x_i-\overline{C}_i)^2 + (1/|C|)\sum_{x\in C}\sum_{i=1}^{F}(\overline{C}_i-y_i)^2 + (2/|C|)\sum_{x\in C}(x-\overline{C})(\overline{C}-y)$. The first term is the intra-variance estimate of COP and the third term is 0 since $\sum_{x\in C}(x-\overline{C})=0$. Thus, since the mean value is always less than or equal to the maximum, we can state that $(1/|C|)\sum_{x\in C} d(x,y) = \text{intra}_{COP}(C) + d(\overline{C},y) \leq \max_{x\in C} d(x,y) = \text{inter}_{COP}(C)$. □

Thus, COP takes its maximum value in the improbable case where the closest point not in the cluster is in the centroid of the cluster.

Next we show that COP satisfies the two properties required by the SEP algorithm: partiality and context-independent optimality. First, it can naturally evaluate partial partitions. It can even evaluate a partial partition with just one cluster, since the inter-variance estimate considers the data not in the partial partition evaluated. Second, we will show that COP is a context-independent optimal index. First of all, let us describe how to compute COP for the union of two partial partitions:

$$COP(P^Y \cup P^Z,X) = \frac{1}{|Y|+|Z|}\left(\sum_{C\in P^Y}|C|\frac{\text{intra}_{COP}(C)}{\text{inter}_{COP}(C)} + \sum_{C\in P^Z}|C|\frac{\text{intra}_{COP}(C)}{\text{inter}_{COP}(C)}\right) = \frac{1}{|Y|+|Z|}(|Y|COP(P^Y)+|Z|COP(P^Z)) \quad (1)$$

Now, we can prove that $COP(P_1^Y,X) < COP(P_2^Y,X) \Leftrightarrow COP(P_1^Y \cup P^Z,X) < COP(P_2^Y \cup P^Z,X)$.

Proof. $COP(P_1^Y \cup P^Z,X) < COP(P_2^Y \cup P^Z,X)$ can be written, following Eq. (1), as $(1/(|Y|+|Z|))(|Y|COP(P_1^Y)+|Z|COP(P^Z)) < (1/(|Y|+|Z|))(|Y|COP(P_2^Y)+|Z|COP(P^Z))$. Then, removing equal terms on each side we obtain $COP(P_1^Y) < COP(P_2^Y)$, proving the equivalence of both inequalities. □

To conclude the analysis of COP we would like to remark that if a set of partial partitions, U, has been evaluated by COP, then the union of these partitions can be easily evaluated. Generalizing Eq. (1):

$$COP\left(\bigcup_{P\in U}P,X\right) = \frac{1}{\sum_{P\in U}n(P)}\sum_{P\in U}n(P)COP(P)$$

where $n(P)=\sum_{C\in P}|C|$ is the number of objects in a partition and all partitions in U are disjoint partitions.

This means that COP must be computed just once for each node in the SEP algorithm.

5. Experimental setup

In order to evaluate the SEP algorithm—the new method we propose to efficiently find the best partition in the extended partition set of a cluster hierarchy—we performed an extensive experiment and compared SEP with the usual approach. We built 261 hierarchies using a well-known agglomerative hierarchical algorithm known as the sequential agglomerative hierarchical nonoverlapping (SAHN) method [8]. This method begins with the all-singleton partition and iteratively joins the two closest clusters. The proximity measure used to compute the distance between two clusters is a user-defined parameter, but the usual measures are single-linkage (nearest neighbour), complete-linkage (furthest neighbour) and average-linkage (group average) [8]. We built a hierarchy using SAHN with each of the aforementioned proximity measures for 80 synthetic and 7 real datasets.

We built the synthetic datasets as follows. We randomly selected 10 points in a two-dimensional square between (0,0) and (50,50). We created a Gaussian cluster centred at each random point. The covariance matrix for each cluster was $\sigma_k \times I$ where I is the two-dimensional identity matrix. We randomly selected σ_k between 0.1 and 3 for each cluster, with a distribution similar to an inverted Gaussian bell, so that extreme variances were favoured. The number of points in each cluster was randomly selected from a uniform distribution between 25 and 50.

We built 40 datasets with the previously described method; 10 of them without noise and the rest with randomly added noise points: 10 datasets with $0.05 \times N$, 10 with $0.1 \times N$ and 10 with $0.2 \times N$ noisy points, where N is the total number of points in the dataset. As an example, the left part of Fig. 3 shows one of the noiseless datasets used in the experiment.

In some cases these types of datasets can lead to misleading results. If several clusters overlap significantly it is not clear whether the clustering method should separate them or not. In some cases it may be reasonable to allow the clustering method to consider a set of clusters as a single cluster. Unfortunately, this is usually a subjective matter, so there is no clear criterion for this problem. The left part of Fig. 3 shows a clear example where not all the clusters are clearly defined and therefore, the correct partition is debatable.

Thus, in order to avoid debate about the subjectiveness of "natural" groups in a dataset, we built another set of 40 datasets for which we ensured that there were no two clusters too close together. More precisely, we removed the centre of cluster C_k and randomly created another cluster centre if $d(C_k,C_l) < 3\times(\sigma_k+\sigma_l)\ \forall l \neq k$.

The seven datasets from real domains were drawn from the UCI repository [35]: Ecoli, Ionosphere, Iris, Page-blocks, Segmentation, Vehicle and Yeast. Table 1 summarizes the characteristics of the real datasets.

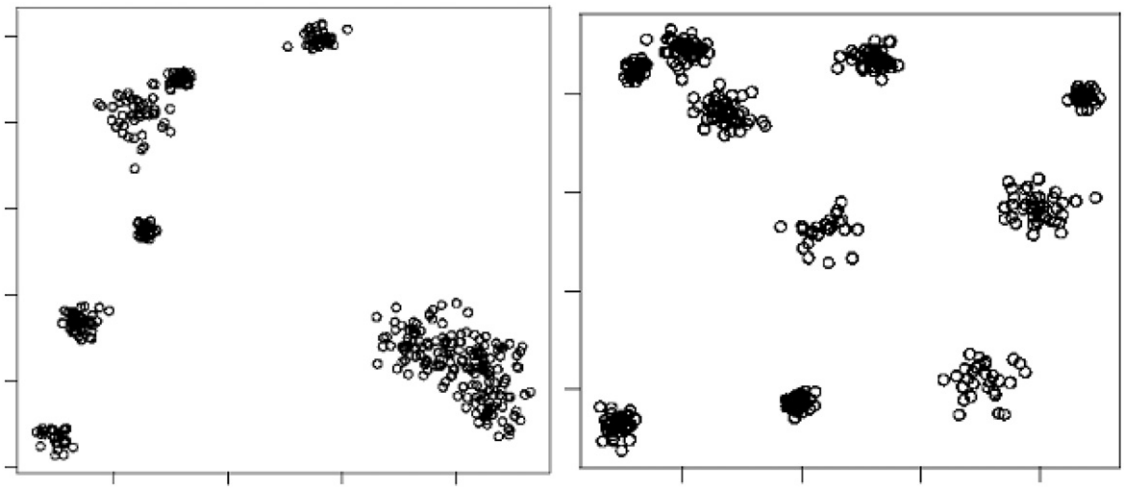

Fig. 3. Two of the noiseless datasets used in the experiment: with overlapping clusters (left) and without overlapping clusters (right). Note that both datasets have 10 clusters.

Table 1
Characteristics of the real datasets used in the experiment.

Dataset	Clusters	Dimensions	Data points
Ecoli	8	7	336
Ionosphere	2	34	351
Iris	3	4	150
Page-blocks	5	10	5473
Segmentation	7	19	2310
Vehicle	4	18	846
Yeast	10	8	1484

Once the hierarchies had been created, we first compared all the partitions in a hierarchy with the correct partition using a similarity measure. We selected the most similar, $P' = \text{argmax}_{P \in H}(\text{sim}(P,P^*))$, where sim is a similarity measure. We called this partition the ms-partition. Then we used the five CVIs described in Section 3.2 to compute the best partition in the hierarchy and checked which indices were able to find the ms-partition. As mentioned in Section 3, many CVIs tend to incorrectly favour partitions with many small clusters. Therefore, we did a reduced search with $\alpha = \sqrt{N}$. In other words, we limited the search to the partitions between 2 and $\sqrt{N}$ clusters [21,28].

Although we designed the COP index to work together with the SEP algorithm, it can also be used as a normal CVI. In order to analyse the quality of COP, we included it in the experiment described in the previous paragraph. In other words, we also used the COP index to perform the usual hierarchy post-processing approach. Thus, we were able to compare COP with five widely used CVIs. Nevertheless, the main contribution in this paper is the combination of the SEP algorithm and the COP index. Measuring the quality of the COP index allowed us to better estimate the benefits introduced by the SEP algorithm.

Finally, we used the SEP algorithm and COP index to conduct a search in the extended partition set of each hierarchy and checked if the selected partition, $\hat{P}$, was the ms-partition. Note that in this case the partition found could be even more similar to the correct partition (remember that the search is conducted over a wider space) and therefore, we checked if $\text{sim}(\hat{P},P^*) \geq \text{sim}(P',P^*)$.

There are many similarity measures that compare partitions [36–39] and it is not easy to decide which of them should be used. To obtain more robust results we decided to use two similarity measures. The first one is the widely used adjusted rand (AR) index [40]. The second one is an index based on the properties of information theory and is called variation of information (VI). Meilă showed in [41,42] that VI possesses many significant properties.

6. Results

In this section we show the results obtained in the experiment described. We first focus on the synthetic datasets and then we show the results for the real datasets. To reduce the size of the tables we refer to the single-linkage, average-linkage and complete-linkage algorithms as "sl", "al" and "cl", respectively.

6.1. Synthetic datasets

In this section we show the aggregate results for each group of datasets that were created using the same process. Thus, we have eight groups of 10 datasets with different levels of overlapping (overlapped and nonoverlapped) and noise (0%, 5%, 10% and 20%). We distributed the results in four tables (one per noise level) and show how many times each CVI was able to find the best possible partition in the hierarchy; i.e. the ms-partition. Remember that we used two similarity measures to compare partitions: adjusted rand and variation of information. The upper bound was 10 for each dataset group and algorithm. The goal for SEP/COP was to find a partition at least as "good" as the ms-partition. In the results for SEP/COP we included a superscript indicating the number of partitions "better" than the ms-partition. Tables 2, 3, 4 and 5 show the results for 0%, 5%, 10% and 20% noise level, respectively.

If we focus on the noiseless datasets and the usual approach, the results show that the Calinski–Harabasz index has the best behaviour for overlapped datasets. The remaining indices show a similar behaviour, except for COP and Dunn. COP shows a better behaviour than the rest although it is slightly worse than Calinski–Harabasz. The results for Dunn are clearly the worst. The results for nonoverlapped datasets are better, as expected, but the conclusions drawn from the overlapped datasets remain true.

If we focus on the SEP algorithm combined with the COP method we can observe that results for overlapped and non-overlapped datasets differ. While the results for overlapped datasets do not show any improvement over COP, SEP shows almost perfect behaviour for nonoverlapped datasets.

Table 2
Number of times each CVI found the ms-partition in the noiseless datasets.

Overlapping	Overlapped						Nonoverlapped					
Similarity	AR			VI			AR			VI		
Algorithm	sl	al	cl	sl	al	cl	sl	al	cl	sl	al	cl
CH	4	3	3	6	6	2	6	7	6	8	8	6
C-Index	3	1	1	3	2	3	3	3	5	4	4	6
Gamma	3	1	1	3	2	3	2	3	5	3	4	6
DB	2	1	2	4	2	3	4	5	6	6	6	7
Dunn	0	0	0	0	1	1	1	1	3	3	2	4
COP	2	2	5	4	3	5	5	6	7	7	7	8
SEP/COP	3^1	2^1	3^2	5^1	3^1	3^1	9^5	9^5	10^6	10^5	10^5	10^6

Table 3
Number of times each CVI found the ms-partition in the 5% noise datasets.

Overlapping	Overlapped						Nonoverlapped					
Similarity	AR			VI			AR			VI		
Algorithm	sl	al	cl	sl	al	cl	sl	al	cl	sl	al	cl
CH	1	3	1	2	7	4	3	3	7	5	4	6
C-Index	1	2	3	2	3	3	3	4	6	4	5	7
Gamma	1	3	2	2	4	2	3	4	6	4	5	7
DB	1	2	1	2	2	2	3	2	3	4	3	3
Dunn	0	0	0	1	1	0	1	1	4	2	2	4
COP	1	2	1	2	3	2	3	3	6	5	4	7
SEP/COP	2^1	2^1	3^2	5^2	3^1	3^3	8^5	7^4	9^5	9^5	8^4	9^5

Table 4
Number of times each CVI found the ms-partition in the 10% noise datasets.

Overlapping	Overlapped						Nonoverlapped					
Similarity	AR			VI			AR			VI		
Algorithm	sl	al	cl	sl	al	cl	sl	al	cl	sl	al	cl
CH	0	1	5	1	2	8	0	3	5	1	4	5
C-Index	0	1	3	1	2	5	0	3	6	1	3	6
Gamma	0	1	2	1	2	4	0	3	6	1	3	6
DB	0	0	1	0	1	2	0	2	4	1	2	4
Dunn	0	0	1	0	1	1	0	1	1	1	1	1
COP	0	1	2	1	2	5	0	3	6	0	4	2
SEP/COP	3^1	2^1	4^2	6^2	3^1	6^3	9^6	7^5	10^9	10^6	8^5	9^8

The results for the 5% noise datasets show that all CVIs except Dunn behave similarly. The only exception is the good result for overlapped datasets and average-linkage algorithm achieved by Calinski–Harabasz. The results for Dunn are again the worst. If we compare Tables 2 and 3 we can conclude that noise affects the results since all CVIs show lower scores in general terms. However, SEP/COP does not appear to be significantly affected by noise and, therefore, the results for overlapped datasets are now similar to or slightly better than the rest. Obviously, the best results for nonoverlapped datasets are again obtained by SEP/COP.

Let us now focus on the effects of a noise increment from 5% to 10% (Table 4): the scores of every CVI are clearly reduced. Nevertheless, the relative differences between them remain similar with a few exceptions. Calinski–Harabasz showed a slight improvement on overlapped datasets and Davies–Bouldin showed poor results, similar to those for Dunn.

Table 5
Number of times each CVI found the ms-partition in the 20% noise datasets.

Overlapping	Overlapped						Nonoverlapped					
Similarity	AR			VI			AR			VI		
Algorithm	sl	al	cl	sl	al	cl	sl	al	cl	sl	al	cl
CH	0	1	2	0	3	6	0	3	6	0	3	6
C-Index	0	1	2	0	3	5	0	1	1	0	1	1
Gamma	0	1	2	0	3	5	0	1	1	0	1	1
DB	0	1	0	0	2	0	0	0	3	0	1	3
Dunn	0	0	0	0	1	2	0	3	2	0	4	2
COP	0	1	1	0	3	6	0	2	7	0	3	7
SEP/COP	2^1	2^1	4^3	5^2	4^1	4^2	9^6	8^5	8^7	10^6	8^5	8^7

In contrast, the results for SEP/COP remain similar and, therefore, it is clearly the best choice in this case. As an example, the results show that SEP/COP achieves a 64% score while the best CVI in this case (CH) obtains a 29% score.

Finally, to confirm the effects produced by noise, let us focus on Table 5. The results for the CVIs keep declining in general terms. Dunn and Davies–Bouldin still show the worst results for overlapped datasets, but, surprisingly, they show a slight improvement for nonoverlapped ones. Calinski–Harabasz and COP show the best results for nonoverlapped datasets, since the noise increment did not appear to affect them in this case. Once again, the results for SEP/COP remained similar, so that the differences from the rest increased. It is clear that for noisy environments SEP outperforms the usual approach.

6.2. Real datasets

In this subsection we show the results for real datasets. We would like to remark that the results for real datasets can be quite misleading, since labels in real datasets do not always describe "natural" clusters in the data. There are several reasons for this: inadequate data collection or feature selection, excessive cluster overlapping, labelling mistakes, incorrect labelling criteria, etc.

The analysis of the results for real datasets revealed an unexpected result. For the adjusted rand index, almost all partitions obtained by any method showed an extremely low similarity to the correct partition. We measured the similarity level of all the partitions in each hierarchy and checked that for the adjusted rand index most of the ms-partitions were nonsense (usually partitions with hundreds of clusters). However, VI selected partitions that were intuitively closer to the correct one.

Therefore, we discarded the results obtained by adjusted rand and we focused on the results obtained by the VI index. Since all the real datasets are different we did not group them and, hence, Table 6 shows the results for the seven real datasets. Note that VI is actually a dissimilarity measure, so that lower values of the index denote more similar partitions.

All similarity values were calculated with a precision of six decimal places, but due to the lack of space we show the values rounded to three decimal places. The first column of Table 6 refers to the dataset and algorithm. The second column shows the similarity value of the ms-partition and, therefore establishes an upper-bound for the usual approach. Remember that SEP/COP can improve this upper-bound. The values achieving the upper-bound—based on the six decimal places—are underlined. The values improving the upper-bound are indicated in bold and underlined. The last row shows the results in the same way it was

Table 6
Similarity level, regarding to VI index, between the partition selected for each method and the correct partition. The UB column shows the upper-bound for the classic method, i.e. the similarity level of the most similar partition in the hierarchy.

	UB	CH	C-Index	Gamma	DB	Dunn	COP	S/C
Ecoli								
sl	2.076	2.112	2.120	2.171	2.171	2.171	2.171	2.171
al	1.151	1.611	1.488	1.488	1.283	2.089	1.178	1.244
cl	1.301	1.301	1.311	1.546	1.687	2.295	1.301	1.301
Ionosphere								
sl	0.962	0.962	1.270	0.962	0.962	0.962	0.962	0.962
al	0.962	1.242	1.358	0.962	0.962	0.962	0.962	0.962
cl	1.412	2.306	2.804	2.804	2.804	1.829	2.805	3.175
Iris								
sl	0.667	0.667	0.667	0.667	0.667	0.667	0.667	0.667
al	0.609	0.609	1.512	0.667	0.667	0.667	0.667	0.667
cl	0.854	1.067	1.785	1.067	0.854	1.785	0.854	0.854
Page-blocks								
sl	0.634	0.636	0.636	0.636	0.636	0.636	0.636	0.636
al	0.635	2.093	2.572	0.635	0.854	0.635	0.635	**0.634**
cl	0.635	3.305	3.459	0.635	0.764	0.635	0.635	**0.633**
Segmentation								
sl	2.218	2.814	2.814	2.814	2.816	2.814	2.814	2.814
al	2.150	2.237	2.814	2.814	2.814	2.814	2.814	2.814
cl	2.267	3.588	2.814	2.814	2.814	2.814	2.814	2.814
Vehicle								
sl	2.018	2.135	2.135	2.135	2.260	2.135	2.018	2.018
al	2.501	2.526	4.537	4.537	2.526	4.521	2.526	2.526
cl	2.523	3.214	4.800	4.800	2.523	4.985	2.523	2.523
Yeast								
sl	2.505	2.505	2.555	2.527	2.586	2.514	2.514	2.511
al	2.495	3.300	2.505	2.505	2.505	2.495	2.505	2.505
cl	2.495	3.768	5.220	2.505	2.505	2.495	2.505	2.505
Count		5	1	5	5	7	8	9^2

done for synthetic datasets; i.e. it indicates how many times each CVI was able to find the ms-partition. In the case of the SEP/COP algorithm the superscripts indicate how many times it found a partition better than the ms-partition.

The results for the usual approach show that COP and Dunn are the best CVIs for the real datasets. It is surprising how well the Dunn index performs, since its behaviour was really poor for synthetic datasets. In contrast, Calinski–Harabasz, the best CVI for the synthetic datasets, showed average results for the real datasets. C-Index is without question the CVI with the poorest results.

If we focus on SEP/COP we can see that it outperforms any CVI used with the usual approach. Moreover, it is able to find a partition better than the ms-partition in two cases. Thus, it showed that it is the best choice for real datasets as well as for synthetic datasets.

7. Discussion

In this section we analyse the results obtained from the experimental work and discuss them from an overall point of view. Then, we briefly justify the validity of the SAHN method.

The experimental work showed that the usual approach to obtain a single partition from a cluster hierarchy is not satisfactory and that the SEP algorithm is a better solution. With regard to the experiments carried out for synthetic datasets with nonoverlapped clusters the combination of SEP and COP was shown to be almost optimal even with a high noise level. This result contrasts with the usual approach, since the results for the usual approach are always below those for SEP. Moreover, since the results for the usual approach decline as noise increases the differences are greater in noisy datasets.

In many environments datasets with nonoverlapping clusters can be considered as not being realistic. However, we claim that these datasets have a key advantage: there is no doubt about the natural groups in the datasets. Therefore the interpretation of the results is straightforward and the noise added by external factors is minimized.

Nevertheless, in order to perform a robust experiment, we replicated the experimental work over datasets for overlapped clusters. In this case SEP was not the best option in the noiseless datasets, but it was one of the best. Calinski–Harabasz showed the best behaviour for noiseless datasets with overlapped clusters and the results for COP were also satisfactory. When noise was added the results for the usual approach declined while SEP confirmed its low sensitivity to noise. As a result, SEP was again the best option.

In summary, the experiment with synthetic datasets shows that SEP/COP is the best option. Furthermore, in many cases it is able to find a partition better than the ms-partition—the upper-bound for the usual approach. This happens 51%, 56%, 64% and 65% of the times that SEP/COP finds a satisfactory partition for 0%, 5%, 10% and 20% noise level, respectively. With regard to the usual approach Calinski–Harabasz is the CVI achieving the best results although the results for COP are quite similar. Dunn is the CVI obtaining the poorest results.

The datasets used in the last experiment should be considered the most realistic since they are datasets from real applications. However, the knowledge about the "true" clusters is limited in these datasets. As usual, we used the class labels to define the "true" clusters. However, this approach can sometimes be misleading since labels do not always represent natural clusters.

The results for datasets from real applications showed little difference between most of the CVIs used with the usual approach. COP was the best, but, surprisingly, Dunn—the CVI with worst results for synthetic datasets—also showed a satisfactory behaviour. In contrast, C-Index found the ms-partition just once. The remaining CVIs showed an average behaviour.

The combination of SEP and COP showed slightly better results than the best results for the usual approach and therefore, it is confirmed as the best choice. In addition, the results showed that COP, the CVI proposed in this work, also behaves satisfactorily when it is not used together with SEP. Moreover, it was shown to be one of the best CVIs analysed in this work.

This work focuses on providing tools to interpret hierarchies generated by SAHN. The SAHN method has been used for several decades and has been shown to be useful in many environments. Nevertheless, to ensure that it worked satisfactorily in our experiment, we compared it to the well-known k-means algorithm [8]. The comparison was not straightforward, since the number of clusters must be specified to k-means and setting it to the actual number of clusters does not guarantee optimal behaviour. Thus, we decided to compare the upper-bounds of both algorithms. We computed k-means for each dataset and we set k to all values between 2 and $\sqrt{N}$, as we did when finding the best partition in the hierarchy based on the usual approach. Then, we compared all the partitions to the correct partition using the adjusted rand and VI indices, and selected the best values as the upper-bound.

Finally, we compared the upper-bounds for k-means with the upper-bounds for SAHN. The results confirmed that both algorithms obtain similar results for synthetic datasets, since the upper-bound for SAHN improved the upper-bound for k-means in 52% of the cases. However, for real datasets the upper-bounds for SAHN were higher in 90% of the cases. Therefore, the results confirmed that the experiment was carried out for reasonable hierarchies.

8. Conclusions

In this work we showed that the usual procedure used to post-process a cluster hierarchy to reduce it to a single partition is not always effective. Furthermore, we proposed a new post-processing method, called SEP, which can search efficiently in a more extensive partition set obtaining satisfactory results. In addition, we defined a new cluster validity index, COP, which is suitable for the proposed method.

The results showed that SEP/COP is superior to the commonly used method. The results with synthetic datasets were conclusive. Moreover, they showed that SEP is not significantly affected by noise in the data, while the usual process and CVIs are negatively affected even by a low noise level.

The results with real datasets were not as conclusive although, due to imprecisions in this kind of data, the results must be analysed with caution. However, the results again showed that SEP/COP is the best option for these datasets.

Consequently, we claim that SEP is a better hierarchy post-processing method than the usually used one. Moreover, we claim that COP is a good cluster validity index either to be used with SEP or as a normal CVI. However, we encourage further research on CVIs suitable for SEP. Just as previous research efforts improved CVIs, we consider that research work focused on CVIs suitable for SEP will produce positive results and consolidate the SEP algorithm.

We are now working on a variation of the SEP algorithm that can output a reduced set of optimal partitions instead of a single partition. This approach would allow a complex hierarchy to be reduced to a simplified hierarchy with just the most significant partitions. A dendrogram obtained from such a reduced hierarchy would clearly be easier for a human user to analyse, while keeping the most relevant information about the hierarchical nature of the data.

Acknowledgments

This work was funded by the University of the Basque Country, DAMISI Project (EHU 08/39), the Diputación Foral de Gipuzkoa and the E.U.

References

[1] K.J. Holzinger, H.H. Harman, Factor Analysis, University of Chicago Press, Chicago, 1941.
[2] P.A. Vijaya, M.N. Murty, D.K. Subramanian, Efficient bottom-up hybrid hierarchical clustering techniques for protein sequence classification, Pattern Recognition 39 (12) (2006) 2344–2355.
[3] S. Seal, S. Komarina, S. Aluru, An optimal hierarchical clustering algorithm for gene expression data, Information Processing Letters 93 (3) (2005) 143–147.
[4] B. Mirkin, Clustering for Data Mining: A Data Recovery Approach, Chapman & Hall, CRC, Boca Raton, FL, 2005.
[5] C.-H. Chou, M.-C. Su, E. Lai, A new cluster validity measure and its application to image compression, Pattern Analysis and Applications 7 (2) (2004) 205–220.
[6] A. Ma, I.K. Sethi, Distributed k-median clustering with application to image clustering, in: Proceedings of the 7th International Workshop on Pattern Recognition in Information Systems, , 2007, pp. 215–220.
[7] D. Barbará, S. Jajodia (Eds.), Applications of Data Mining in Computer Security, Kluwer Academic Publishers, Norwell, MA, 2002.
[8] A.K. Jain, R.C. Dubes, Algorithms for Clustering Data, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.
[9] A.K. Jain, Data clustering: 50 years beyond k-means, Pattern Recognition Letters 31 (18) (2009) 651–666.
[10] M. Halkidi, Y. Batistakis, M. Vazirgiannis, On clustering validation techniques, Journal of Intelligent Information Systems 17 (2001) 107–145.
[11] P.H.A. Sneath, R.R. Sokal, Numerical Taxonomy, Books in Biology, W. H. Freeman and Company, San Francisco, 1973.
[12] M. Liu, X. Jiang, A.C. Kot, A multi-prototype clustering algorithm, Pattern Recognition 42 (2009) 689–698.
[13] T.W. Liao, Clustering of time series data—a survey, Pattern Recognition 38 (2005) 1857–1874.
[14] H.B. Silva, P. Brito, J.P. da Costa, A partitional clustering algorithm validated by a clustering tendency index based on graph theory, Pattern Recognition 39 (5) (2006) 776–788.
[15] V. Chaoji, M. Al Hasan, S. Salem, M.J. Zaki, SPARCL: an effective and efficient algorithm for mining arbitrary shape-based clusters, Knowledge and Information Systems 21 (2) (2009) 201–229.
[16] J. Goldberger, T. Tassa, A hierarchical clustering algorithm based on the Hungarian method, Pattern Recognition Letters 29 (11) (2008) 1632–1638.
[17] A.L.N. Fred, M.N. Lao José, A new cluster isolation criterion based on dissimilarity increments, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (8) (2003) 944–958.
[18] S.-J. Oh, J.-Y. Kim, A hierarchical clustering algorithm for categorical sequence data, Information Processing Letters 91 (3) (2004) 135–140.
[19] G. Karypis, E.-H. Han, V. Kumar, CHAMELEON: a hierarchical clustering algorithm using dynamic modeling, IEEE Computer 32 (1999) 68–75.
[20] D.H. Widyantoro, T.R. Ioerger, J. Yen, An incremental approach to building a cluster hierarchy, in: ICDM '02: Proceedings of the 2002 IEEE International Conference on Data Mining, , 2002, pp. 705–708.
[21] M. Kim, R.S. Ramakrishna, New indices for cluster validity assessment, Pattern Recognition Letters 26 (15) (2005) 2353–2363.
[22] K.-L. Wu, M.-S. Yang, J.-N. Hsieh, Robust cluster validity indexes, Pattern Recognition 42 (11) (2009) 2541–2550.
[23] L.F. Lago-Fernández, F. Corbacho, Normality-based validation for crisp clustering, Pattern Recognition 43 (2010) 782–795.
[24] K. Mali, S. Mitra, Clustering and its validation in a symbolic framework, Pattern Recognition Letters 24 (14) (2003) 2367–2376.
[25] J.C. Bezdek, N.R. Pal, Some new indexes of cluster validity, IEEE Transactions on Systems, Man and Cybernetics, Part B 28 (3) (1998) 301–315.
[26] Y. Jung, H. Park, D.-Z. Du, B.L. Drake, A decision criterion for the optimal number of clusters in hierarchical clustering, Journal of Global Optimization 25 (2003) 91–111.
[27] G.W. Milligan, M.C. Cooper, An examination of procedures for determining the number of clusters in a data set, Psychometrika 50 (2) (1985) 159–179.
[28] U. Maulik, S. Bandyopadhyay, Performance evaluation of some clustering algorithms and validity indices, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (12) (2002) 1650–1654.
[29] N.R. Pal, J. Biswas, Cluster validation using graph theoretic concepts, Pattern Recognition 30 (6) (1997) 847–857.
[30] T. Calinski, J. Harabasz, A dendrite method for cluster analysis, Communications in Statistics 3 (1974) 1–27.
[31] L.J. Hubert, J.R. Levin, A general statistical framework for assessing categorical clustering in free recall, Psychological Bulletin 83 (1976) 1072–1080.
[32] F.B. Baker, L.J. Hubert, Measuring the power of hierarchical cluster analysis, Journal of the American Statistical Association 70 (1975) 31–38.
[33] D.L. Davies, D.W. Bouldin, A clustering separation measure, IEEE Transactions on Pattern Analysis and Machine Intelligence 1 (1979) 224–227.
[34] J.C. Dunn, A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters, Journal of Cybernetics 3 (1973) 32–57.
[35] A. Asuncion, D. Newman, UCI machine learning repository, 2007. URL: 〈http://www.ics.uci.edu/~mlearn/MLRepository.html〉.
[36] A.N. Albatineh, M. Niewiadomska-Bugaj, D. Mihalko, On similarity indices and correction for chance agreement, Journal of Classification 23 (2006) 301–313.
[37] D. Pfitzner, R. Leibbrandt, D. Powers, Characterization and evaluation of similarity measures for pairs of clusterings, Knowledge and Information Systems 19 (3) (2009) 361–394.
[38] V. Batagelj, M. Bren, Comparing resemblance measures, Journal of Classification 12 (1) (1995) 73–90.
[39] W.H. Day, The complexity of computing metric distances between partitions, Mathematical Social Sciences 1 (3) (1981) 269–287.
[40] L. Hubert, P. Arabie, Comparing partitions, Journal of Classification 2 (1985) 193–218.
[41] M. Meilă, Comparing clusterings by the variation of information, in: Proceedings of the Sixteenth Annual Conference on Computational Learning Theory COLT, , 2003, pp. 173–187.
[42] M. Meilă, Comparing clusterings—an axiomatic view, in: Proceedings of the Twenty-second International Conference on Machine Learning ICML 2005, , 2005, pp. 577–584.

About the Author—IBAI GURRUTXAGA received the M.Sc. degree in Computer Science from the University of the Basque Country in 2002. He is an Associate Professor in the Computer Architecture and Technology Department of the University of the Basque Country, and is currently pursuing the Ph.D. at the same department. He is working in data mining and pattern recognition, focusing on supervised and unsupervised classification (decision trees, clustering, computer security and intrusion detection).

About the Author—IÑAKI ALBISUA received the M.Sc. degree in Computer Science from the University of the Basque Country in 1998. He is an Associate Professor in the Computer Architecture and Technology Department of the University of the Basque Country, and is currently pursuing the Ph.D. at the same department. He is working in data mining and pattern recognition techniques, focusing on classifiers with explanation capacities, learning from imbalanced data and statistical analysis.

About the Author—OLATZ ARBELAITZ received the M.Sc. and Ph.D. degrees in Computer Science from the University of the Basque Country in 1993 and 2002, respectively. She is an Associate Professor in the Computer Architecture and Technology Department of the University of the Basque Country. She has worked in autonomous robotics, combinatorial optimization and supervised and unsupervised machine learning techniques, focusing lately in outlier detection techniques.

About the Author—JOSÉ IGNACIO MARTÍN received the M.Sc. and Ph.D. degrees in Computer Science from the University of the Basque Country in 1990 and 1994, respectively. He is an Associate Professor in the Computer Architecture and Technology Department of the University of the Basque Country. His main research is in the area of data mining and pattern recognition, and parallel programming.

About the Author—JAVIER MUGUERZA received the M.Sc. and Ph.D. degrees in Computer Science from the University of the Basque Country in 1990 and 1996, respectively. He is an Associate Professor in the Computer Architecture and Technology Department of the University of the Basque Country. His research interests include data mining and pattern recognition (decision trees, clustering, security and intrusion detection and optical character recognition), and parallel programming.

About the Author—JESÚS MARÍA PÉREZ received the M.Sc. and Ph.D. degrees in Computer Science from the University of the Basque Country in 1993 and 2006, respectively. He is an Associate Professor in the Computer Architecture and Technology Department of the University of the Basque Country. His research interests include data mining and pattern recognition techniques, focusing on classifiers with explanation capacities, learning from imbalanced data and statistical analysis.

About the Author—IÑIGO PERONA received the M.Sc. degree in Computer Science from the University of the Basque Country in 2008. He is granted to pursue the Ph.D. at the Computer Architecture and Technology Department of the University of the Basque Country. He is working in data mining and pattern recognition, focusing on supervised and unsupervised classification (outlier detection, clustering, computer security and intrusion detection).