A New, Fast and Accurate Algorithm for Hierarchical Clustering on Euclidean Distances

Elio Masciari[1], Giuseppe Massimiliano Mazzeo[1], and Carlo Zaniolo[2]

[1] ICAR-CNR
[2] UCLA
{masciari,mazzeo}@icar.cnr.it, zaniolo@cs.ucla.edu

Abstract. A simple hierarchical clustering algorithm called CLUBS (for CLustering Using Binary Splitting) is proposed. CLUBS is faster and more accurate than existing algorithms, including k-means and its recently proposed refinements. The algorithm consists of a divisive phase and an agglomerative phase; during these two phases, the samples are repartitioned using a least quadratic distance criterion possessing unique analytical properties that we exploit to achieve a very fast computation. CLUBS derives good clusters without requiring input from users, and it is robust and impervious to noise, while providing better speed and accuracy than methods, such as BIRCH, that are endowed with the same critical properties.

1 Introduction

The clustering challenge. Cluster analysis represents a fundamental and widely used method of knowledge discovery, for which many approaches and algorithms have been proposed over the years [9]. Among the most popular methods, we find partition-based clustering (e.g. *k-means*[10]), density based clustering (e.g. *DBScan*[6]), hierarchical methods (e.g. *BIRCH*[14]) and grid-based methods (e.g. *STING* [13]). The continuous stream of clustering algorithms proposed over the years underscores the fact that the logical and algorithmic complexities of this many-facet problem have yet to be tamed completely, and that along with the great progress achieved in the past, significant progress should be expected in the future. In particular, it is well known that no clustering algorithm completely satisfies both accuracy and efficiency requirements, thus a good clustering algorithm has to be evaluated w.r.t. some external criteria that are independent from the metric being used to compute clusters. Indeed, in this paper we propose an algorithm that significantly improves the state of the art in clustering analysis, with respect to *speed, repeatability,* and *accuracy* whose performances have been evaluated using widely accepted clustering validity metric.

Current Solutions: Who is the best in terms of speed and accuracy? In order to compare our performances we preliminary tested existing clustering solutions. We chose algorithms that are widely used by data miners due to their general purpose nature. More in detail, we evaluated algorithms that satisfy user needs in a wide variety of application scenarios. In terms of *speed* of computation, the standard of paragon is set by the k-means [10] algorithm that has as objective minimizing the average distance of the samples from their cluster centroids. Owning to its efficiency, simplicity, and the

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 111–122, 2013.

naturalness of its objective, k-means has become the most widely used clustering algorithm in practical applications. But k-means also suffers from serious shortcomings: in particular, the algorithm does not assure *repeatability* of results, which instead depends on the choice of the initial k points. In practice, therefore, the data mining analyst will have to select the value of k, and the initial k points, via some exploratory dry runs. k-means' shortcomings have motivated much research work [11,2,4], seeking to reduce the variability of the results it produces and bring it closer to the 'unsupervised learning' archetype. A second line of research has instead produced different mining algorithms to overcome k-means' problem while keeping with its general objective of clustering the points around centroids. Approaches such as grid-based clustering work by partitioning the data space into cells arranged in a grid and then merging them to build clusters, while density based approaches search for fully-connected dense regions. Finally, hierarchical clustering methods work by performing a hierarchical decomposition of data in either bottom-up (agglomerative) or top-down (divisive) way. In this respect hierarchical approaches offer good performances w.r.t. the accuracy of clustering. All the techniques mentioned here present advantages and weakness that will be discussed in detail in the related work section. For the goal of assessing the quality of our approach, it is worth noticing that the hierarchical clustering algorithm BIRCH [14] is *stable* (i.e. its results do not vary depending on some initial parameter setting), *accurate*, *impervious to noise* and *scalable to very large data sets*. However, BIRCH is typically not as fast as k-means.

Our Solution. The above thought-provoking discussion guided our search for a new clustering algorithm. Indeed, in this paper we propose a new hierarchical algorithm called CLUBS (for CLustering Using Binary Splitting) whose *speed* performances are better than k-means *and* whose *accuracy* overcomes previous hierarchical algorithms while operating in a *completely unsupervised* fashion. The first phase of the algorithm is divisive, as the original data set is split recursively into miniclusters through successive binary splits: the algorithm's second phase is agglomerative since these miniclusters are recombined into the final result. Due to its features our algorithm can be used also for refining other approaches performances. As an example it can be used to overcome k-means initial assignment problem since its low complexity will not affect the overall complexity while the accuracy of our results will guarantee an excellent initial assignment of cluster centroids. Further, our approach induces during execution a *dynamic hierarchical* grid that will better fit the dataset w.r.t. classical grid approaches that exploit a fixed grid instead. Finally, the algorithm exploits the analytical properties of the Quadratic Sums of Squares (SSQ in the following) function to minimize the cost of merge and split operations, and indeed the approach results really fast. One may argue that many different measures could be used for cluster computation but the accuracy of SSQ is as good as other cluster distance measures (e.g. Single Link, Complete Link, Average) for real case scenarios *and* its computation can be made faster than other measures. These properties are discussed in Section 2.

Main Difference of CLUBS w.r.t. other approaches. CLUBS works in a completely unsupervised way and overcomes the main limitations that beset other algorithms. In particular, we have that (1) CLUBS is not tied to a fixed grid, (2) it can backtrack on previously wrong calculation, and (3) it performs also well on non-globular clusters where

clusters are not spherical in shape, this feature will be intuitively understood after the partitioning and recombination strategy will be detailed in Section 3 (BIRCH does not perform as well, because it uses the notion of radius or diameter to control the boundary of a cluster, and the same drawback also affects k-means like algorithms). Moreover we have that (4) CLUBS can detect the natural clusters present in data, while in Birch each node in the auxiliary tree exploited (called CF tree) can hold only a limited number of entries due to its size thus a CF tree node does not always correspond to what a user may consider a natural cluster. Finally, (5) density based algorithms like DBSCAN are very sensitive to clustering parameters like Minimum Neighborhood Points and they fail to identify clusters if density varies and if the data set is too sparse and different sampling affects density measures, however we compared CLUBS against OPTICS that allows to detect clusters with different densities instead. As will be clear by experimental evaluation, CLUBS does not suffer these limitations due to unique features of SSQ and the two-phase algorithm.

2 Background

After recalling some basic notions used in our algorithm, we discuss binary partitioning and the cluster quality measures there used. Throughout the paper, for each dataset a d-dimensional data distribution D is assumed. D will be treated as a multi-dimensional array of integers with volume n^d (without loss of generality, we assume that all dimensions of D have the same size). The number of non-zero elements of D will be denoted as N. A *range* ρ_i on the i-th dimension of D is an interval $[l..u]$, such that $1 \leq l \leq u \leq n$. Boundaries l and u of ρ_i are denoted by $lb(\rho_i)$ (*lower bound*) and $ub(\rho_i)$ (*upper bound*), respectively. The size of ρ_i will be denoted as $size(\rho_i) = ub(\rho_i) - lb(\rho_i) + 1$. A *block* b (of D) is a d-tuple $\langle \rho_1, \ldots, \rho_d \rangle$ where ρ_i is a range on the dimension i, for each $1 \leq i \leq d$. Informally, a block represents a "hyper-rectangular" region of D. A block b of D with all zero elements is said to be a *null block*. The volume of a block $b = \langle \rho_1, \ldots, \rho_d \rangle$ is given by $size(\rho_1) \times \ldots \times size(\rho_d)$ and will be denoted as $vol(b)$. Given a point in the multidimensional space $\mathbf{x} = \langle x_1, \ldots, x_d \rangle$, we say that $\mathbf{x}$ belongs to the block b (written $\mathbf{x} \in b$) if $lb(\rho_i) \leq x_i \leq ub(\rho_i)$ for each $i \in [1..d]$.

Given a block $b = \langle \rho_1, \ldots, \rho_d \rangle$, let x be a coordinate on the i-th dimension of b such that $lb(\rho_i) \leq x < ub(\rho_i)$. Coordinate x divides the range ρ_i of b into $\rho_i^{low} = [lb(\rho_i)..x]$ and $\rho_i^{high} = [(x+1)..ub(\rho_i)]$, thus partitioning b into $b^{low} = \langle \rho_1, \ldots, \rho_i^{low}, \ldots, \rho_d \rangle$ and $b^{high} = \langle \rho_1, \ldots, \rho_i^{high}, \ldots, \rho_d \rangle$. The pair $\langle b^{low}, b^{high} \rangle$ is said to be the *binary split* of b along the dimension i at the position x; dimension i and coordinate x are said to be the *splitting dimension* and the *splitting position*, respectively.

Informally, a binary partition can be obtained by performing a binary split on D (thus generating the two sub-blocks D^{low} and D^{high}), and then recursively partitioning these two sub-blocks with the same binary hierarchical scheme.

Definition 1. *Given a d-dimensional data distribution D with volume n^d, a* binary partition *BP of D is a binary tree such that the root of BP is the block $\langle [1..n], \ldots, [1..n] \rangle$ and for each internal node p of BP the pair of children of p is a binary-split of p.* □

Clustering Computation Preliminaries. Given a dataset $\mathcal{DS}$ cluster analysis aims at producing a clustering $C = \{C_1, \cdots, C_n\}$ that is a subset of the set of all subsets of $\mathcal{DS}$ such that C contains disjoint (non-overlapping) subsets, covering the whole object set (we refer in this paper exclusively to hard clustering problem, where every data point belongs to one and only one cluster). Consequently, every point $x \in \mathcal{DS}$ is contained in exactly one and only one set C_i. These sets C_i are called clusters.

Definition 2. *Let C_s be a cluster (set) of N d-dimensional points. Let $\mathbf{S} = (S_1, \ldots, S_d) = \sum_{\mathbf{p} \in C_s} \mathbf{p}$ be the vector representing the sum of points in C_s. The center of C_s is $\mathbf{C_s^0} = \frac{\mathbf{S}}{N}$. Let $\mathbf{Q} = (Q_1, \ldots, Q_d)$, where $Q_i = \sum_{\mathbf{p} \in C} p_i^2$, be the vector whose i-th coordinate is the sum of the squared i-th coordinates of the points in S. The SSQ (Sum of Squares) of C_s is defined as:*

$$SSQ(C_s) = \sum_{\mathbf{p} \in C_s} dist^2(\mathbf{p}, \mathbf{C_s^0}) = \sum_{\mathbf{p} \in C} \sum_{i=1}^{d} (p_i - C_s^0)^2 = \sum_{i=1}^{d} \sum_{\mathbf{p} \in C} (p_i - C_s^0)^2 = \sum_{i=1}^{d} \sum_{\mathbf{p} \in C} (p_i^2 - 2 \cdot p_i \cdot C_s^0 + (C_s^0)^2) = \sum_{i=1}^{d} \sum_{\mathbf{p} \in C} p_i^2 - 2 \cdot C_s^0 \cdot \sum_{\mathbf{p} \in C} p_i + N \cdot (C_s^0)^2)$$

we recall that N is the number of points in C and

$$\sum_{\mathbf{p} \in C} p_i = C_s^0 \cdot N$$

thus we obtain by substituting:

$$\sum_{i=1}^{d} \sum_{\mathbf{p} \in C} p_i^2 - \sum_{i=1}^{d} \frac{(\sum_{\mathbf{p} \in C} p_i)^2}{N}$$

finally by definition of Q_i and S_i we obtain:

$$SSQ(C_s) = \sum_{i=1}^{d} (Q_i - \frac{S_i^2}{N}) \quad (1)$$

From the latter, it is clear that, in order to quickly compute the SSQ of a cluster, we need only to store $\mathbf{Q}$, $\mathbf{S}$, and N. In the next section we will show how these information can be used effectively and efficiently to optimize the divisive and agglomerative steps of the CLUBS algorithm.

3 CLUBS: A New Clustering Algorithm

In order to obtain a good tradeoff between accuracy and efficiency we exploit in this paper a new really fast hierarchical approach. Among hierarchical algorithms, bottom-up approaches tend to be more accurate but have a higher computational cost than the top-down approaches [9]. The higher cost is due to the higher number of candidate clusters to be taken into account. To overcome this limitation, in our approach, the agglomerative step is only used on mini-clusters generated by a first divisive process, this results in a remarkable efficiency increase. Top-down partitioning exploiting greedy algorithms has been widely used in the multidimensional data compression due to its efficiency. Here we use a similar divisive approach to minimize the SSQ among the data belonging to clusters, we recall again that in literature many measure have been proposed (e.g. EES) that works in a similar way as SSQ but we chose SSQ since it offers a really fast

computation while maintaining an high accuracy in cluster model evaluation. Thus, our clustering algorithm consists of two steps, where in the first step we use binary hierarchical partitioning to produce a set of mini-clusters and in the second step, we pairwise merge the mini-clusters so obtained in a bottom-up fashion. In both steps the clusters are defined by a hierarchical partition of the multi-dimensional space. The partition can be compactly represented by a binary tree (BT in the following), where:1) each node is associated with a range of the multi-dimensional domain; 2) the root is associated with the whole data domain; 3)for each inner node n, its children are associated with a pair of ranges representing a (rectangular) partition of n.

Each node also maintains summary information about points inside its range, to expedite the clustering computation. The top-down splitting works as follows. As auxiliary structure, we maintain a priority queue of clusters whose elements are ordered on the basis of the SSQ of each cluster. At each iteration, the algorithm performs the following two steps: A) select the cluster C_s that exhibits the highest SSQ (i.e. the one on top of the priority queue), and then B) partition this C_s in such a way that the overall SSQ reduction, denoted ΔSSQ, is maximized. For step B, we compute $\Delta SSQ(i, j)$ for each dimension i and for each cutting position j; then we choose the position j that guarantees the maximum ΔSSQ. This computation can be done very efficiently since we pre-compute Q and S, and therefore we need a single scan of the data. We repeat these two steps, A and B above, while ΔSSQ is greater than the average SSQ. We recall that the partition (i.e., the cluster tree) is built by exploiting a greedy strategy. To this end, the tree is constructed top-down, by means of leaf-node splitting. At each step, the leaf with the largest SSQ is chosen, and it is split as to maximize ΔSSQ . Being SSQ a measure of a range skewness, we perform splits as long as ΔSSQ remains "significant". After the early splits that yield large SSQ reductions, the values of ΔSSQ become smaller and smaller, until after n splits both SSQ and ΔSSQ become 0 (since each point has become its own cluster). Thus, the average SSQ reduction per split is SSQ_0/n, and we will compare this value against the current ΔSSQ to decide when we should stop splitting, The rationale for this criterion is clearly illustrate by Fig. 1, where the typical ΔSSQ slope is displayed against the average SSQ: there is no gain in splitting beyond the turning point (marked with a solid circle) since the SSQ reduction is less than the average ΔSSQ and thus imputable to random distributions rather than cluster-like ones.

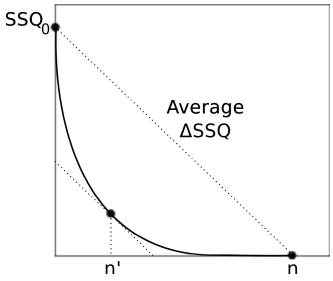

Fig. 1. Average SSQ and ΔSSQ example plots

The splitting process just described is tied to the grid partitioning and thus may cause a non-optimal splitting of some clusters. The successive phase overcomes this limitation since the merging is performed considering all the possible pairs of adjacent mini-clusters, and recombining those that offer best SSQ reduction. This agglomerative process offers significant advantages. One is that it merges clusters in different grid partitions, thus overcoming non optimal splits obtained in the first phase (see Fig. 3(b) and Fig. 3(c)). The second critical advantage is that the computational complexity of this bottom-up step is very low since the number of merging steps is related to the number clusters that is very low compared to usual dataset sizes. The final advantage is that this phase also halts automatically, producing an algorithm that does not require any seeding or other parameters from the user a really nice feature that is not shared by all clustering algorithms.

The Clustering Algorithm. Fig. 2 provides a more formal description of the CLUBS algorithm. We use the *initializeTree* to load the dataset into the root of the auxiliary tree structure BT exploited for partitioning. Once the tree structure has been initialized the *topdownsplitting* step starts. In particular, the root of BT is added to a priority queue whose ordering criterion is based on the SSQ values of clusters stored in the queue. The initial cluster assignment performed by *initializeClusters* is composed by the root r of BT and the initial SSQ is the one computed on r. The function *computeAverageDeltaSSQ* averages the actual SSQ for all the points in the cluster. The function *computeWeightedDeltaSSQ* is applied to the cluster C_s that is currently on top of the priority queue. The $weighted\Delta_{SSQ}$ is computed as the average gain of SSQ obtained by splitting C_s as explained above for Δ_{SSQ}, i.e. we pre-compute the marginal sums (S and Q) for a given splitting point (w.r.t the coordinates ordering) and reassigning the splitting point based on these partial sums. In order to improve the effectiveness of splits the value of Δ_{SSQ} is raised to a power of p, $p < 1$, thus obtaining $weighted\Delta_{SSQ}$ value. If $weighted\Delta_{SSQ}$ is greater than $avgDeltaSSQ$ computed by *computeAverageDeltaSSQ* then we proceed with the split, otherwise we do not. We use values of p that are less than 1, since for $p \geq 1$ we would end up splitting clusters where the gain does not exceed the Average Δ_{SSQ} associated with a random distribution. This would result in a large number of small clusters, where both intra-cluster and inter-cluster distances small. We instead seek values of p that reduce the former while magnifying the latter. We determined experimentally that the best value is $p = 0.8$ regardless the dataset feature thus the user is not required to set any parameter, due to space limitations we cannot report here the detailed discussion of the experiments being conducted.

When no more top-down splits are possible, the *topDownSplitting* ends and we begin the *bottomUpMerging*. In order to obtain more compact clusters, we select (by running *selectBestPair*) the pair of clusters that, if merged, yields the least SSQ increase (that is assigned to $minInc$ by function *computeSSQIncrease*). This merging step is repeated until $minInc$ becomes larger than $avgDeltaSSQ$. Fig. 3 shows the algorithm in action. After three steps, the initial samples in 3(a) are partitioned according to the grid shown in Fig. 3(b). The algorithm takes seven more splitting steps producing the partition of Fig. 3(c). The merging phase produces the final five clusters that a human will instinctively recognize at a glance Fig. 3(d).

Input:
A dataset DS of n points
Output:
A set of clusters C.
Vars:
An auxiliary binary tree BT;
An initial cluster assignment C'.
Method: CLUBS
1: $BT :=$ `initializeTree`(DS);
2: $C' :=$ `topDownSplitting`(BT);
3: $C :=$ `bottomUpMerging`(C');
4: **return** C;

Function `topDownSplitting`$(BT) : C'$;
Vars:
A priority queue PQ;
A boolean $finished$;
A double Δ_{SSQ};
A double $avgDeltaSSQ$;
Method:
1: $PQ :=$ `add`$(BT.root())$;
2: $C' =$ `initializeClusters`;
3: $finished = false$;
4: $avg\Delta_{SSQ} =$ `computeAverageDeltaSSQ()`;
5: **while** !$finished$ **do begin**
6: $C_s = PQ.get()$;
7: $weighted\Delta_{SSQ} =$ `computeWeightedDeltaSSQ`(C_s);
8: **if** $(weighted\Delta_{SSQ} > avg\Delta_{SSQ})$ **then**
9: $C' :=$ `update`(C');
10: **else** $finished = true$;
11: **end while**
12: **return** C';

Function `bottomUpMerging`$(C') : C$;
Vars:
A pair of cluster $Pair$;
A double $avgDeltaSSQ$;
A double $minInc$;
Method:
1: $C := C'$;
2: $Pair :=$ `selectBestPair`(C');
3: $minInc :=$ `computeSSQIncrease`$(Pair)$;
4: $avgDeltaSSQ =$ `computeAverageDeltaSSQ`$()$;
5: **while** $minInc < avgDeltaSSQ$ **do begin**
6: $C :=$ `merge`$(Pair)$;
7: $Pair :=$ `selectBestPair`(C);
8: $minInc :=$ `computeSSQIncrease`$(Pair)$;
9: **end while**;
13: **return** C;

Fig. 2. The CLUBS clustering algorithm

We point out that producing axis parallel cuts is not a limitation, we can still obtain, in our approach, non parallel cuts however this will not improve the performances of the algorithm. Furthermore, also grid based approaches are tied to parallel cuts since they allow more efficient computation without paying any accuracy loss.

In terms of computational complexity, we see that, in order to split, we have to compute the SSQ for each dimension and for each splitting point. Thus, each split has a complexity $O(n \cdot d \cdot l)$ and we perform s splits. The bottom-up step contributes to the overall complexity with a term $O(k^2)$ where k is the number of clusters, since for each cluster we have to consider all the possibly adjacent clusters for merging; but since $k << n$ we can disregard this term. Thus the complexity is as follows:

Proposition 1. *Algorithm CLUBS works in $O(n \cdot d \cdot l \cdot s)$ where n is the number of points, d is the number of dimensions, l is the number of splitting positions for each dimension and s is the number of splits.*

4 Experimental Evaluation

An extensive set of experiments was executed to evaluate the performance of CLUBS. In particular, we compared our method with BIRCH [14], K-means++ [2](we refer to it as KM++) and k*-means [4] (we refer to it as SMART) and OPTICS [1].

Our test suite encompasses a large number of widely used benchmarks over a wide spectrum of different characteristics. Due to space limitations we can present here a small subset of the results, so we choose the really interesting results we obtained on a severe test bench, i.e. microarray data. We used two publicly available dataset on Gene Expression Omnibus Database: a dataset provided by [8], *Dataset 1* hereafter, and a dataset provided by [5], *Dataset 2* hereafter.

As regards *Dataset 1*, authors examined 42 patients by using Affymetrix HU133A (Affymetrix, Santa Clara CA) microarrays. Patients were subdivided in three groups.

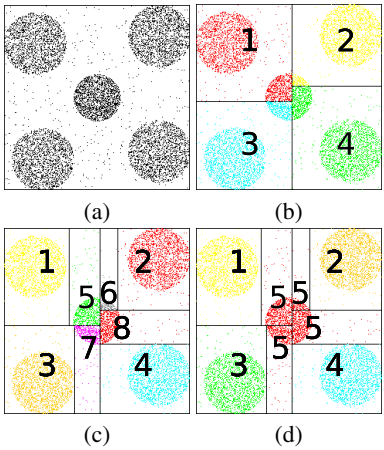

Fig. 3. Execution steps of CLUBS

Women at usual breast cancer risk undergoing mammoplasty reduction (RM) , women with breast cancer undergoing surgery for either an ER+ or ER- breast tumor (HN), and high-risk patients, consisting of women undergoing prophylactic mastectomy (PM). Dataset providers selected 98 differentially expressed genes in HN w.r.t. RM and they built a matrix of these genes for all three groups. The resulting dataset was analyzed by clustering in order to catch the difference among three groups.

Dataset 2 comprises samples extracted from human breast cancer cells analyzed using the Affymetrix U133A 2.0 gene chips (Affymetrix, Santa Clara, CA). Dataset provider considered 4 group of cells treated with 20 lh/ml of actein at 6 and 24 hours, and cells treated with 40 lg/ml of actein at 6 and 24 hours in order to elucidate the effect of actein. The initial preprocessing was performed using the GCRMA method. The statistical significance of differential expression with respect to the same reference value was calculated using the empirical Bayesian LIMMA (LI Model for MicroArrays).

We started our analysis considering these preprocessed datasets on which we used CLUBS and the other clustering algorithms for the sake of comparison. The obtained results are reported in Table 1 where values represent SSQ per dataset and milliseconds.

The results obtained are quite convincing both for the accuracy and the execution times where CLUBS offer best performances. In particular our clustering method correctly detected the number of clusters in the data (3 clusters for Dataset 1 and 4 clusters for Dataset 2). Indeed, CLUBS showed a nice feature when clustering Dataset 1: the HN group contains two subgroups ER+ and ER-, CLUBS during the splitting step identified these two subgroups that have been collapsed in a single cluster after the merging step. To asses, the validity of the approach we exploited several method-independent quality measure that are reported in the following.

Quality of Clustering Results. Here we will evaluate the quality of the results CLUBS produces and its reliability. The issue of finding method-independent measures for clustering results has been the source of much topical discussions, but over time sound measures have emerged that can be used reliably to compare the quality of the results produced by a wide range of clustering algorithms [3]. In particular the following three measures have sound theoretical and practical bases. The *Variance Ratio* measures the ratio between the average distance between points belonging to different clusters and the average distance between points within the same cluster [3]. The range of variance ratio is $[0, \infty)$ and larger values of variance ratio indicate better clustering quality. The *Relative Margin* reports the average of the *Relative Point Margin* defined as the ratio between the distance of a given point x to the center of the cluster it belongs to

Table 1. Accuracy and Time Performances for our test datasets

Algorithm	$Dataset1$		$Dataset2$	
	SSQ	time	SSQ	time
CLUBS	2.01E+8	2.513	1.77E+2	0.0784
OPTICS	3.55E+8	5.271	1.79E+2	0.2456
BIRCH	2.67E+8	9.124	1.78E+2	0.3522
KM++	4.31E+8	2.913	1.76E+2	0.1154
SMART	4.65E+8	3.025	1.81E+2	0.1243

and the distance between x and the closest cluster center different from the cluster x belongs to [3]. The range of relative margin is $[0, 1)$, and lower relative margin indicates a better clustering. The *Weakest Link* measure is defined as the maximal value of weakest link over all pairs of points belonging to the same cluster, divided by the shortest between-cluster distance [3]. The range of values of weakest link is $[0, \infty)$. Lower values of weakest link represent better clusterings. The results obtained for the above mentioned quality measures are given in Table 2(a): they show that CLUBS outperforms other methods significantly, producing values for Relative Margin & Weakest Link (resp. Variance Ratio) that are significantly lower (larger) than those other methods, i.e. clusters of much better quality. These results also confirm that CLUBS finds the exact number of clusters and the quality of the found cluster is overwhelming w.r.t the other methods.

Additional Quality Measures. SSQ is a natural and widely used norm of similarity, but a devil's advocate can point out that other clustering algorithms might not measure their effectiveness in terms of SSQ or even the compactness of each cluster around its centroid. Thus, in this section we will measure the quality of the clusters produced by CLUBS using very different criteria inspired by the nearest subclass classifiers that were previously used in a similar role in [12] and [7].

A first relevant evaluation measure in this approach is the error rate of a k-Nearest Neighbor classifier defined by the clustering results. This value provide relevant information about the ability of the clustering method under evaluation to minimize the errors due to incorrect assignment of points to the proper cluster. Indeed, this information is crucial for biological data analysis. Thus, for each point, we can check whether the dominant class of the k closer elements allows to correctly predict the actual class of membership (there is no relationship between the value of k used here and that of k-means). Thus, the total number of points correctly classified measures the effectiveness

Table 2. Clustering Quality Measures Evaluation

(a)

Dataset 1	#Clusters	Variance Ratio	Relative Margin	Weakest Link
M-CLUBS	3	75.41	0.098	0.817
OPTICS	5	56.18	0.135	2.045
BIRCH	6	63.42	0.176	1.934
KM++	3	65.44	0.157	4.152
SMART	3	64.77	0.198	4.789
Dataset 2	#Clusters	Variance Ratio	Relative Margin	Weakest Link
M-CLUBS	4	81.33	0.066	0.713
OPTICS	4	67.18	0.153	1.876
BIRCH	4	70.41	0.182	1.943
KM++	4	68.67	0.201	3.412
SMART	4	69.97	0.225	3.725

(b)

Dataset 1			
method/index	ε	$e_{k=10}$	$q_{k=10}$
CLUBS	0.0661	0.0984	0.9998
OPTICS	0.1253	0.1976	0.8934
BIRCH	0.1154	0.2010	0.9756
KM++	0.1002	0.1974	0.9803
SMART	0.1086	0.2101	0.9057
Dataset 2			
method/index	ε	$e_{k=10}$	$q_{k=10}$
CLUBS	0.0054	0.0352	0.9999
OPTICS	0.0432	0.1312	0.9875
BIRCH	0.0165	0.0953	0.9923
KM++	0.0487	0.1657	0.9764
SMART	0.0568	0.1789	0.9734

of the clustering at hand. Formally, the error $e_k(D)$ of a k-NN classifier exploiting a the distance matrix among every pair of points. D can be defined as

$$e_k(D) = \frac{1}{N}\sum_{i=1}^{N}\gamma_k(i)$$

where N is the total number of points, and $\gamma_k(i)$ is 0 if the predicted class of the i-th point (x_i) coincides with its actual class, and 1 otherwise. Low values of the $e_k(D)$ index denote high-quality clusters.

Following [7], we can go deeper in our evaluation by measuring the average number of elements, in a range of k elements (we recall again that we use the expected cluster size value), having the same class as the point under consideration. Practically, we define q_k as the average percentage of points in the k-neighborhood of a generic point belonging to the same class of that point. Formally:

$$q_k(D) = \frac{1}{N}\sum_{i=1}^{N}\frac{|\mathcal{N}_k(i) \cap Cl(i)|}{min(k, n_i)}$$

where $Cl(i)$ represents the actual class associated with the i-th point in the dataset, $n_i = |Cl(i)|$, and $\mathcal{N}_k(i)$ is the set of k points having the lowest distances from x_i, according to the distance used at hand. This value will provide a really interesting information, in fact it will measure the *purity* of the clusters since it take into account the number of points wrongly assigned to a cluster. In principle, a Nearest Neighbor classifier exhibits a good performance when q_k is high. Furthermore, q_k provides a measure of the stability of a Nearest-Neighbor: high values of q_k make a k-NN classifier less sensitive to increasing values k of neighbors considered. The sensitivity of the clustering can also be measured by considering, for a given group of points x, y, z, the probability that x and y belong to the same class and z belongs to a different class, but z is more similar to x than y is. We denote this probability by $\varepsilon(D)$, estimated as:

$$\varepsilon(D) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{(n_i-1)(N-n_i)}\sum_{Cl(j)=Cl(i),j\neq i}\ \sum_{Cl(k)\neq Cl(i)}\delta_D(i,j,k)\right)$$

where δ_D is 1 if $D(i,j) < D(i,k)$, and 0 otherwise. This value gives information about the ambiguity in cluster assignments. Here too, low values of $\epsilon(D)$ denote a good performance of the clustering under consideration.

The results in Table 2(b) show that CLUBS produces better results than the other algorithms. Table 2(b) shows that CLUBS offers the best performance on all indices and in particular the really high values of q_k (it is practically 1 since it detects exactly the number of clusters for each dataset and the point assignment to cluster is correct) allow to asses that the clusters are well defined, and CLUBS outperforms both BIRCH and OPTICS. In measuring e_k and q_k, we used neighborhoods of size 10 (this value is the actual cluster size available by datasets provider). The overall structure of the clusters and the points distribution for Dataset 1 (results in Table 2(b)) produced superior performance for CLUBS on every index, with particularly low values of ε. This result is confirmed also for Dataset 2 and suggests that CLUBS exhibits the highest effectiveness compared to the other approaches even when SSQ is not the chosen metric.

5 Conclusion

The naturalness of the hierarchical approach for clustering objects is widely recognized, and also supported by psychological studies of children's cognitive behaviors[1]. CLUBS is providing the analytical and algorithmic advances that have turned this intuitive approach into a data mining method of superior accuracy, robustness and speed. The speed achieved by our approach is largely due to CLUBS' ability of exploiting the analytical properties of its quadratic distance functions to simplify the computation. We conjecture that similar benefits might be at hand for situations where the samples are in data streams or in secondary store. These situations were not studied in this paper, but represent a promising topic for future research.

References

1. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: Optics: Ordering points to identify the clustering structure. SIGMOD Record 28(2), 49–60 (1999)
2. Arthur, D., Vassilvitskii, S.: k-means++: the advantages of careful seeding. In: SODA, pp. 1027–1035 (2007)
3. Ben-David, S., Ackerman, M.: Measures of clustering quality: A working set of axioms for clustering. In: NIPS, pp. 121–128 (2008)
4. Cheung, Y.M.: k*-means: A new generalized k-means clustering algorithm. Pattern Recognition Letters 24(15), 2883–2893 (2003)
5. Einbond, L.S., Su, T., Wu, H., Friedman, R., Wang, X., Ramirez, A., Kronenberg, F., Weinstein, I.B.: The growth inhibitory effect of actein on human breast cancer cells is associated with activation of stress response pathways. I. J. of Cancer 121(9), 2073–2083 (2007)
6. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD (1996)
7. Flesca, S., Manco, G., Masciari, E., Pontieri, L., Pugliese, A.: Fast detection of xml structural similarity. TKDE 17(2), 160–175 (2005)
8. Graham, K., De Las Morenas, A., Tripathi, A., King, C., Kavanah, M., Mendez, J., Stone, M., Slama, J., Miller, M., Antoine, G., Willers, H., Sebastiani, P., Rosenberg, C.L.: Gene expression in histologically normal epithelium from breast cancer patients and from cancer-free prophylactic mastectomy patients shares a similar profile. Br. J. Cancer 102(8), 1284–1293 (2010)
9. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann (2000)
10. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: 5-th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
11. Ostrovsky, R., Rabani, Y., Schulman, L.J., Swamy, C.: The effectiveness of lloyd-type methods for the k-means problem. In: FOCS (2006)
12. Veenman, C.J., Reinders, M.J.T.: The nearest subclass classifier: A compromise between the nearest mean and nearest neighbor classifier. IEEE Trans. Pattern Anal. Mach. Intell. 27(9), 1417–1429 (2005)
13. Wang, W., Yang, J., Muntz, R.R.: Sting: A statistical information grid approach to spatial data mining. In: VLDB, pp. 186–195 (1997)
14. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: An efficient data clustering method for very large databases. In: SIGMOD, pp. 103–114 (1996)

[1] Jodie M. Plumert, Flexibility in Children's Use of Spatial and Categorical Organizational Strategies in Recall Developmental Psychology 1994. Vol. 30. No. 5. 738-747.