27

Mining Image Data

Chabane Djeraba
IRIN, Nantes University, France

Gregory Fernandez
Wayne State University

INTRODUCTION

A key issue in data analysis methods either in a model design step or as part of the model projection in real-time operations is the grouping of data items into clusters. First, the clustering analysis is essentially an unsupervised process. Its objective is to group a given repository of unclassified data items into meaningful clusters. In a sense clusters are data driven and obtained solely from the data. Second, clustering may support an supervised process on the basis of classes of preclassified data items. The objective is to classify newly encountered yet unclassified data items. Typically, the given clusters (obtained by the unsupervised process

training) of data items are used to learn the descriptions of classes, which in turn are used to classify new data items. As a result, data items within a cluster are more similar than data items belonging to different clusters.

Any clustering method belongs to either partition or hierarchical clustering classes. A partition clustering method produces a single partition of the data. A partition method is better suited to data mining than a hierarchical clustering method because it is better at handling large data sets for which the construction of a hierarchy is computationally expensive, and it is insensitive to the order of input data items.

In spite of important efforts in partition clustering developments (Jain, Murty, & Flynn, 1999), a problem remains, and few partition clustering approaches deal with it seriously. It is the automatic discovery of the number of clusters. What is the desired number, known by k? Which is the best one? Is it often possible to have an idea of the desired number? When considering clustering in large repositories of data items such as images, is it realistic for the user, even if an expert, to specify the number of the desired clusters in an accurate way?

The goal of this chapter is to contribute answers to these questions by presenting a method that automatically discovers the number of clusters (k). Therefore, the method computes k automatically, rather than manually. We experimented with the method in image repositories. Two keys are necessary and sufficient to open the door of the k-discovery solution. The first one is the partition clustering method that supports multiple iterations according to multiple values of k. The second is the clustering confidence measure. It combines together an intercluster measure (global variance criterion ratio normalized; GVRCN) that considers the confidence of the whole clusters, and an intraclusters measure (local variance criterion ratio; LVCR) that considers the confidence of individual clusters. We will show that the combination of two levels of measures presents an interesting result. Furthermore, the confidence of results depend not only on the clustering algorithm and cluster confidence measures, but also on data item descriptors. As data item descriptors, we tried Wavelet CDF (2,2)-Cohen-Daubechies-Feauveau (2,2), and CDF (1,1)-Cohen-Daubechies-Feauveau (1, 1) (Cohen, Daubechies, & Feauveau, 1992), known, too, by Haar. The former presents better confidence of clustering results compared with the latter, because it supports less noise. The choice of wavelet descriptors, as in Sheikholeslami and Chatterjee Surojit (2000), is justified by the fact that wavelet descriptors contribute to the detection of clusters of arbitrary frontiers; they are insensitive to the noise; and using the multiresolution property of wavelet descriptors contributes to effectively identifying frontier clusters arbitrarily at different degrees of detail.

We will not present in this chapter the different aspects of indexing and retrieval linked to clustering or how to do semantic clustering. Our goal in this chapter is an experimental study of relationships between content descriptions based on wavelets, k-automatic discovery, and confidence measures of clusters.

We can summarize the scope of the chapter in the following points:

- An overview clustering of large repositories of data items
- Data items, to be clustered, are images
- The number of clusters, known by k, is automatically discovered
- The description of images is based on wavelet coefficients

Confidence measures of the clusters are composed of GVRCN and LVRC.

The chapter is composed of the following sections. The second section presents the related works. The third section highlights our method; more particularly, it presents the k-discovery and clustering methods. The fourth section presents the result experiments of the proposed method.

RELATED WORKS

During these last 30 years many techniques, including partition methods, have been proposed to represent data, to measure proximity (similarity), and to group data items. The most popular partition clustering methods are k-means (McQueen, 1967), k-medoids (Vinod, 1969), and their variants, such as dynamic clustering (Diday, 1973). K-means methods create a random clustering, compute the center of gravity of each cluster, and then assign each data item of the data set to the cluster with the nearest center of gravity. This process is active until obtaining a stable state of clusters. K-medoids are very similar to k-means, except that the centers of gravity are data items of the data set. This characteristic avoids empty clusters. The k-medoids process follows, generally, two steps. In the first step the process selects k data items randomly. In the second step each data item of the data set is assigned, on the basis of similarity, to a selected cluster, called medoid or gravity center. The two steps are repeated until the best clustering is reached.

The different variants of k-medoids methods have been applied in several domains concerned with data item exploration, decision-making, document retrieval, image segmentation, and data item classification. We focus this presentation on three recent variants of k-medoids methods (Kaufman & Rousseeuw, 1990), which may be considered as a representative sample of partition clustering methods.

The first variant of k-medoids, is the partition around medoids (PAM) method. This method tries to find the best medoids from the data set. This is done by evaluation of result swaping, when a medoid is swapped with a data item. The method is robust and returns the best clustering results. However, it is inefficient: Compute TC_{ih} requires $(\tilde{n}k)$ operations (one per nonmedoid item). There are $k(\tilde{n}k)$ pairs, k selected items—medoids—and $(\tilde{n}k)$ unselected items; for each pair of data item we compute one TC_{ih}. Therefore, the complexity for one iteration is $O(k(\tilde{n}k)^2)$. Moreover, we cannot estimate the number of iterations. Here, $n =$ the number of items; $k =$ the number of clusters; $x_i =$ the ith item; $C_J =$ the jth cluster; $C_{xi} =$ the cluster of x_i; $C_{ijh} =$ the cost of swapping roles of x_i (a medoid) and x_h (a normal item) for x_j. The total cost for swapping role of x_i and x_h: $TC_{ih} = \sum_j C_{ijh}$.

The second variant of k-medoids, is called clustering large applications (CLARA). The method is more suitable for large repositories of data items than the PAM method because it is computationally quicker, the cardinality of the sample is less voluminous than the data set, and its complexity is about $O(k(4\tilde{0}k)^2 + k(\tilde{n}k))$ for each iteration. The method is applied to a sample of a data repository rather than an entire data repository. Therefore, it is quicker than PAM. However, the disadvantage of the method is the random selection of the sample, so the confidence of clustering is not as good as for the PAM method.

The third variant of k-medoids, is a clustering large application based on randomized search (CLARANS). The method presents better performance than the previous ones. The complexity of the method is about $O(\tilde{n}k)$ by iteration. Experimental results (Raymond & Jiawei, 1985) showed that for the same confidence of clustering, the method outperforms PAM, and for the same run time, the method provides better clusters than CLARA. On the basis of these efficiency and effectiveness characteristics, we considered this partition method as a basic level of our solution to support multiple run executions discover the best values of k.

The best variants of k-medoids methods, such as CLARANS, suffer of two shortcomings. The first is the automatic discovery of k. The second, which is shared by all clustering approaches, is the degeneration of the method performances when the data descriptors are high dimensional. The second disadvantage is not within the scope of the chapter; however, on basis of the strong presence and importance of the high dimensionality in multimedia descriptors, it is interesting to mention it. The high dimensionality of data descriptors does not simplify at all

the complexity of the clustering problem. The performance of lots of clustering algorithms decrease quickly with increasing dimension. The problem of clustering of high-dimensional data has been well investigated very recently, and clustering approaches such as X-tree (Berchtold, Keim, & Kriegel, 1996), Birch (Zhang, Ramakrishnan, & Linvy, 1996), Sting (Wand, Yang, & Muntz, 1997) and OptiGrid (Hinneburg & Keim, 1999) have been proposed to improve the efficiency and the effectiveness of the clustering. For example, OptiGrid (Hinneburg & Keim, 1999) finds clusters in high-dimension spaces with noise by projecting the data onto each axis and then partitioning the data using cutting planes at low-density points. The method shows how projections on subspaces of the input space improve the effectiveness of the clustering process. The method uses statistical measures to produce good projections and, hence, good clustering results.

In this chapter, we highlight an experimental solution to the second shortcoming: automatic discovery of k. In the state of the art, there are very few implemented and proven solutions. Furthermore, to our knowledge there are no solutions "k-automatic discovery" that have been tested in image data sets. Therefore, we limit our presentation to one of the least tested approaches of the state of the art. The approach discovers k automatically in the context of spatial data mining (Raymond & Jiawei, 1985). The first difference between the approach and our method is the domain of application. Our domain of application is the content-based indexing of large image databases; however, the domain of application of the state of the art approach is spatial data mining. This difference means that there are different objectives, different data item descriptors, and different confidence measures. We particularly emphasize the confidence measure. Our method is to propose a solution that considers confidence measures that are more accurate for content-based indexing of large image databases than the approach proposed for spatial data mining. Our solution is inspired, partly, by metrics discussed in Milligan and Cooper (1985) for hierarchical clustering, extended and tested for image databases.

The state of the art approach, called spatial data mining based on clustering algorithms (Raymond & Jiawei, 1994) tries to find k, where k is the most suitable number of clusters for the input data sets. The approach adopts the heuristics of computing the silhouette coefficients developed in Kaufman and Rousseeuw (1990) and Dubes (1987). The silhouette of an object Oj is a value varying between -1 and 1, which indicates how much Oj belongs to the cluster in which Oj is classified. The closer the value is to 1, the higher the degree Oj belongs to its cluster. The silhouette value of a cluster is the average silhouette of all data items in the cluster. Based on experiments, Kaufman and Rousseeuw (1990) proposes the following interpretation of the silhouette: $71\% \leq$ cluster silhouette ≤ 1 means it is a strong cluster; $51\% \leq$ cluster silhouette $\leq 70\%$ means it is a reasonable cluster, $26\% \leq$ cluster silhouette $\leq 50\%$ means it is a weak or artificial cluster; less than 25% means no cluster was found. The silhouette value for k is the mean silhouette values of the k clusters. If the value of k is too small, the reason may be that the clusters are grouped together incorrectly. If the k value is too large, then some clusters may be artificially split. The most suitable k is the one with the highest silhouette value. The experiments of Kaufman and Rousseeuw showed that in spatial data mining, using the highest silhouette coefficient might not lead to intuitive results. For example, some clusters may not have reasonable information (e.g., silhouette value $<50\%$). Therefore, they introduce the following heuristics:

1. Find the value k with the highest silhouette value.
2. When all the k clusters have silhouette values $>51\%$, than *final* $k = k$, and stop. Otherwise, remove the data items in those clusters with silhouette values below 50%, if the total number of data items removed so far is less than 25%.

3. The data items removed are considered noise. Go back to step 1 for a new data set without noise.
4. When in step 3, the amount of noise to be removed exceeds the threshold, then setting final k value is set to 1, indicating in effect that no clustering is reasonable.

The usefulness of these heuristics has been highlighted in specific applications of spatial data mining. However, it is not realistic to generalize the applications of heuristics to any application domain. In our experiments on image data sets, considering 50% of the silhouette value is not realistic. Many images are close together with silhouette values less than 20%. That is why the heuristics used in this approach are not applied with the same ratios. A more fundamental difference concerns silhouette computing itself. The approach presented in the state of the art considers the silhouette of the clustering equal to the average of cluster silhouettes. A cluster silhouette is the average of the data item silhouettes of the considered cluster. This manner of deducing the silhouette of clusters is similar to LVRC, as used in our approach. However, our experiments showed that the highest values of the silhouette do not mean the best clustering. What is missing is the silhouette variance between clusters. The lowest silhouette variance between clusters has been considered in our method; we called it the global variance ratio criterion (GVRC). Therefore, the best value of k corresponds to the biggest silhouette with the lowest silhouette variance. In the following section we present an example in which the highest value of the silhouette (for $k = 8$) does not mean the best clustering; however, the lowest value of the GVRC with the highest silhouette corresponds the best clustering ($k = 13$). The GVRC also has been tested and compared with several measures in Milligan and Cooper (1985). Our experiments may be considered as a form of validation of the Milligan and Cooper (1985) experiments in image repositories.

METHOD

How to Discover the Number of Clusters: k

A critical question in the partition clustering method is the number of clusters (k). In the majority of real-world experimental methods, the number of cluster k is manually estimated. The estimation needs a minimum knowledge of both data repositories and applications, and this requires the study of data. Bad values of k can lead to very bad clustering. Therefore, automatic computation of k is one of the most difficult problems in cluster analysis.

To deal with this problem, we consider a method based on the confidence measures of the clustering. To compute the confidence of the clustering, we consider together two levels of measures: the global level and the local level. GVRC, inspired of Calinski and Harabasz (1974), underlines the global level, and LVRC, inspired by Kaufman and Rousseeuw (1990). GVRC has never been used in partition clustering. It computes the confidence of the whole cluster, and LVRC computes the confidence of individual clusters, known also as the silhouette. Therefore, we combine the two levels of measures to improve the accuracy the computation of the clustering confidence.

Our choices of GVRC and LVRC confidence measures are based on the following arguments. Previous experiments in Milligan and Cooper (1985) in the context of hierarchical clustering, examined 30 measures to find the best number of clusters. They applied all of the measures to test data set and computed how many times an index gave the right k. GVRC presented the best results in all cases, and GVRCN normalized the GVRC measure (GVRCN value is between 0 and 1). A legitimate question may be: How is this

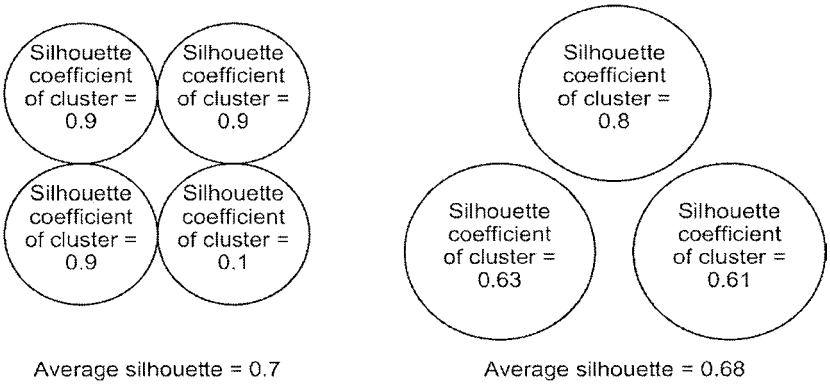

FIG. 27.1. Two clustering with $k = 4$ and $k = 3$.

measure (GVRC) good for partition methods? If GVRC is a good measure for hierarchical methods, is GVRC also good for partition methods? Are there any direct dependencies between data sets (in our case, image descriptors) and the confidence of clustering measures? What are the relationships between the initial medoids and the measures of clustering confidence?

It is too hard to answer all these questions at once without theory improvements and exhaustive real-world experiments. Therefore, our objective approach is not to answer all these important questions accurately, but to contribute to these answers by presenting the results of a real-world experiment. We extend a measure of clustering confidence by considering the best one presented in the literature (Milligan & Cooper, 1985). Then, we examined this measure in our repository data sets.

We combine the two levels of measures to make an accurate computation of clustering confidence. The combination of two levels of measures is an interesting facet of our method. It avoids a clustering state in which the max value of average the confidence of clusters does not mean the best confidence of clustering. For example, in Fig. 27.1, even if $k = 4$ the average confidence (0.7) of clustering is greater than the average confidence for $k = 3$ (0.61); however, for $k = 3$ the local confidence of clusters is better. All local confidence of clusters, when $k = 3$, are greater than 60%. And, for $k = 4$, three clusters have local confidence greater than 90% and one cluster less than 20%. If we consider $k = 4$, then we should ignore the fourth cluster, for which the local confidence is equal to 10%. It is considered as a noisy class. The data items of the fourth cluster are considered noise and then ignored, as in Raymond and Jiawei (1985). That is why we consider $k = 3$, because the variance is lowest, and the average confidence is among the highest, the local qualities are greater than 60%, and there are no noisy classes. On the basis of global and local measures, we consider $k = 3$ as the best number of clusters.

The method is improved by identifying the best cluster qualities with the lowest variance. In certain approaches, such as Raymond and Jiawei (1985), for different values of k, if the confidence of individual clusters is less than 50% and the cardinal of these clusters is less than 25%, then the data items are considered noise. So they are removed, and the approach recalculates the global and local qualities for the remaining data items. These data items are noise. Our experiments showed that fixing such a ratio (50 and 25%, etc.) is not suitable for evaluating cluster confidence. We often obtain good clustering, very similar to image references, with confidence values less than 20%.

GVRC is described by the following formula.

$$VRC(k) = \frac{\frac{trace(B)}{k-1}}{\frac{trace(W)}{n-k}}$$

where n is the cardinality of the data set, k is the number of clusters, $trace\ (B)/(k-1)$ is the dispersion (variance) between clusters, and $trace\ (W)/n-k$ is the dispersion within clusters. The expanded formula is:

$$GVRC(k) = \frac{n-k}{k-1} \frac{\left(\sum_{i=1}^{n} \|x_i - \bar{x}\|^2\right) - \left(\sum_{l=1}^{k}\left(\sum_{x_j \in C_l} \|x_j - \bar{x}_l\|^2\right)\right)}{\sum_{l=1}^{k}\left(\sum_{x_j \in C_l} \|x_j - \bar{x}_l\|^2\right)}$$

The best clustering coincides with the maximum of GVRC. To normalize the GVRC value ($0 \leq$ GVRCN ≤ 1), we compute GVRCN, where $GVRC\_max = GVRC(k')$, and $\forall k, 0 < k \leq k\_max, k \neq k', GVRC(k) < GVRC(k')$. K_max is the maximum of the clusters considered. $K\_max \leq$ the cardinal of the data items.

$$GVRCN(k) = \frac{GVRC(k)}{GVRC\_\max}$$

LVRC measures the confidence of *clusterj* (C_j). The formula is:

$$LVRC(c_j) = \frac{\sum_{i=1}^{cardinality(cj)} lvrc_{x_i}}{cardinality(c_j)}, \quad \text{with} \quad lvrc_{x_i} = \frac{b_{x_i} - a_{x_i}}{\max(a_{x_i}, b_{x_i})}$$

LVRC measures the probability of x_i to belong to the cluster C_{xi}, where a_{xi} is the average dissimilarity of object x_i to all other objects of the cluster C_{xi}, and b_{xi} is the average dissimilarity of object x_i to all objects of the closest cluster C'_{xi} (neighbor of object x_i). Note that the neighbor cluster is rather a second-best choice for object x_i. When cluster C_{xi} contains only one object x_i, the s_{xi} is set to zero ($LVRC_{xi} = 0$).

In our experiments we considered descriptor size: 2048 float, wavelet type: CDF (2, 2). To speed up the clustering process, we specified that $4 \leq k \leq 15$, because we know in advance that the best value of k turns around 10. After activating the clustering method that discovers the best value of k, we obtain the following results:

- $k = 8$, LVRC = 16%, 1− GVRCN = 55%, Medoids = {flag05, dawndusk00, belfry04, waterfall05, waterfall08, animals08, boat03, usa07}
- $k = 10$, LVRC = 3%, 1− GVRCN = 60%, Medoids = {flag07, dawndusk09, belfry04, waterfall05, waterfall08, animals08, montagne08, boat03, flower08, animals07}
- $k = 13$, LVRC = 10%, 1− GVRCN = 90%, Medoids = {flag05, dawndusk00, belfry04, animals07, waterfall05, animals06, flower08, boat01, montagne08, waterfall07, waterfall08, usa06, avion07}
- $k = 14$, LVRC = 9%, GVRCN = 85%, Medoids = {flag05, dawndusk09, belfry04, animals06, waterfall05, animals08, flower08, boat03, belfry02, waterfall09, waterfall08, usa07, belfry03, flower05}

If we consider $k = 13$, we notice that the LVCR of any cluster is greater than 8% (e.g., Table 27.1). However, for $k = 8$, certain LVCRs of clusters are greater than 10% and others

TABLE 27.1
$K = 2$ Clusters: (LVRC: 0.228815, GVRCN: 0.15)

	LVRC	*Medoid*	*Data Items*										
1	0.126265	usa05	usa01	usa02	usa04	Usa06	usa07	Usa08	belfry00	belfry01	belfry02	belfry03	
2	0.331365	flag05	flag00	flag01	flag02	Flag03	flag04	Flag06	flag07	flag08	flag09		

are less than 5%, so the variance of cluster qualities for $k = 8$ is greater than the variance of cluster qualities for $k = 13$. So GVRCN($k = 13$) < GVRCN($k = 8$). For $k = 10$, clustering confidence is weak (3%). So the best value of k discovered automatically does not mean the best value of k manually referenced. However, considering $k = 13$ with LVRC = 9%, we obtain the best value of recall and accuracy. That is why in our approach we consider the k values with the greatest values of LVCR and with the lowest values of the variances. Final confidence = (LVRC + (1 − GVRCN))/2. In our example, confidence($k = 13$) = (10% + 90%)/2 = 50%. Confidence ($k = 8$) = (16% + 55%)/2 = 35%. When we look carefully at the clustering result for $k = 13$, we notice that there are two modeoids "animals" (animals06 and animals08), two medoids "waterfall" (warterfall09, waterfall08), and two medoids "belfry" (belfry02, belfry03). So the clusters obtained automatically are more detailed than image references (clusters obtained manually). The resolutions considered to discriminate images are too high. From the user's point of view there are no contradictions. If we used LVRC exclusively, as in the state of the art approaches (Raymond & Jiawei, 1985), we would consider $k = 8$ as the best confidence value, which is not the best clustering result. In our example, "flowers" and "mountains" have not been discriminated, because there are no medoids that represent flowers and mountains.

K-Automatic Discovery Algorithm

The algorithm is run several times with different k values, and the best configuration of k obtained from all runs is used as the output clustering.

We consider the sequence variable *sorted_clustering* initialized to an empty sequence (<>). *sorted_clustering* contains a sorted, on the basis of *confidencei,* sequence of elements in the form of <$confidence_i$,k_i>, where k_i is the cluster number at i iteration, and $confidence_i$ is the GVRC and all local variance ratio criterion associated to k_i clusters. *sorted_clustering* = <, ... , <$confidence_{i-1}$,k_{i-1}>, <$confidence_i$,k_i>>, where $confidence_{i=}$<*GVRC(k), <LVRC*(c_1), *LVRC*(c_2), ... , *LVRC*(c_{ki})>>.

The algorithm follows five steps:

- The first step initializes the maximum number of k authorized (*max_k*), by default *max_k* = $n/2$. n is the length of the data item repositories.
- The second step applies the clustering algorithm at each iteration k_i (for $k = 1$ to *max_k*), computes *GVRC*(k), and for each k value computes *LVRC*(c_j), for $j = 1, \ldots, k$. $confidence_i$ = <*GVRC(k), <LVRC*(c_1), *LVRC*(c_2), ... , *LVRC*(c_k)>>
- The third step normalizes the GVRC by computing the GVRCN. *GVRC_max* = *max_GVRC*(k) where *GVRC_max* corresponds to the best clustering
- The fourth step considers only k clustering for which *GVRCN*(k) = *GVRC*(k)/ *GVRC_max* is the minimum value and *LVRC*(k) is the maximum. So,

$$\frac{\sum_{i=1}^{k} LVRC(C_i)}{k} + (1 - GVRCN(k))$$

is the maximum, and $\forall j,\ j \in [1,k],\ LVRC(c_k) \geq 1\%$. The results are sorted in *correct_sorted_clustering*. If the final *correct_sorted_clustering* is not empty, therefore, we have at least one solution and the algorithm is stopped. If not, the current step is followed by the fifth step.

- The fifth step looks for false or weak clusters (LVRC $< 1\%$). All false or weak data items of these false or weak clusters are moved to a specific cluster called "noisy cluster." If the cardinality of the noisy cluster is less than 10%, then we compute the k-discovery without considering the false or weak data items. However, if the cardinal of the noisy cluster is greater than 10%, then we consider that there is too much noise and the data item features of the initial repositories should be reconsidered before again applying the algorithm. We deduce that the data item descriptors are not suited to clustering analysis.

```
K-discovery()
{
Set max_k //By default max_k = n/2
sorted_clustering ← < > /* empty sequence */
sorted_clustering contains a sequence of sorted qualities of
 clustering and associated k.
sorted_clustering = <..., <confidence_{i-1},k_{i-1}>,<confidence_i,k_i>>,
 where confidence_{i-1} < confidence_{i-1.} <confidence_i, k_i>= <GVRC(k_i),
 k_i>
// The clustering confidence is often the best when the value
 is high.
for k=1 to max_k do
 Apply chosen clustering algorithm.
current ← GVRC(k),<LVRC(c_1), LVRC(c_2),...., LVRC(c_k)>
/* GVRC(k) = confidence of actual clustering */
sorted_clustering ← insert <k, current> in best_clustering, by
 considering sorted sequence of sorted_clustering.
 end for
GVRC_max ← max_confidence (sorted_clustering)/*GVRC_max =
 best_ clustering */
correct_sorted_clustering ← < >
for k = 1 to max_k do
```

$$\textbf{if } VRCN(k) = \frac{VRC(k)}{VRC\_\max} \geq 1\% \quad \text{and} \quad \forall j,\ j \in [1,k], LRC(c_k) \geq 1$$

```
 then /* C_k is a correct cluster */
  correct_sorted_clustering ← insert <k, confidence_k> in
  correct_sorted_clustering>
 end if
end for
if correct_sorted_clustering = <>
then
{
for i=1 to cardinal(correct_sorted_clustering) do
{
moving false or weak data items of C_ij for which LRC(c_ikj) < 1% to
 Noisy_cluster
if  cardinal(Noisy_cluster) < 10%  then  k-discovery() without
```

```
 considering the noisy data items in Noisy Class
}
Return correct_sorted_clustering.
}
```

Clustering Algorithm

The clustering algorithm is a variant of k-medoids, inspired by Ray and Jiawei (1994). The particularity of the algorithm is the replacement of sampling by heuristics. Sampling consists of finding better clustering by changing one medoid. But finding the best pair (medoid, item) to swap is very costly ($O(k(\bar{n}k)^2)$). That is why heuristics have been introduced in [Ray 94] to improve the confidence of the swap (medoid, data item). To speed up the choice of a pair (medoid, data item), the algorithm sets a maximum number of pairs to test (*num_pairs*), then choose randomly a pair and compares the dissimilarity (the comparison is done by evaluating TC_{th}). If this dissimilarity is greater than the actual dissimilarity, the algorithm continues choosing pairs until the number of pairs chosen reaches the fixed maximum. The medoids found are very dependant on the k first medoids selected. So the approach selects k other item and restarts *num_tries* times (*num_tries* is fixed by user). The best clustering is kept after the *num_tries* tries.

```
Clustering()
{
Initialize num_tries and num_pairs
min_cost ← big_number
for k=1 to num_tries do
  current ← k randomly selected items in the entire data set.
  l ← 1
  repeat
    xi ← a randomly selected item in current
    xh ← a randomly selected item in {entire data set current}.
    if TCih <0 then
      current ← currentxi+xh
    else
      j ← j+1
    end if
  until j ≤ num_pairs
  if min_cost < cost(current) then
    best ← current.
  end if
end for
Return best.
}
```

EXPERIMENTAL RESULTS

We conducted experiments in large image data sets to analyze the performance of the k-automatic discovery method. However, before presenting the performance analysis, we outline the data sets and the metrics used to evaluate the performance.

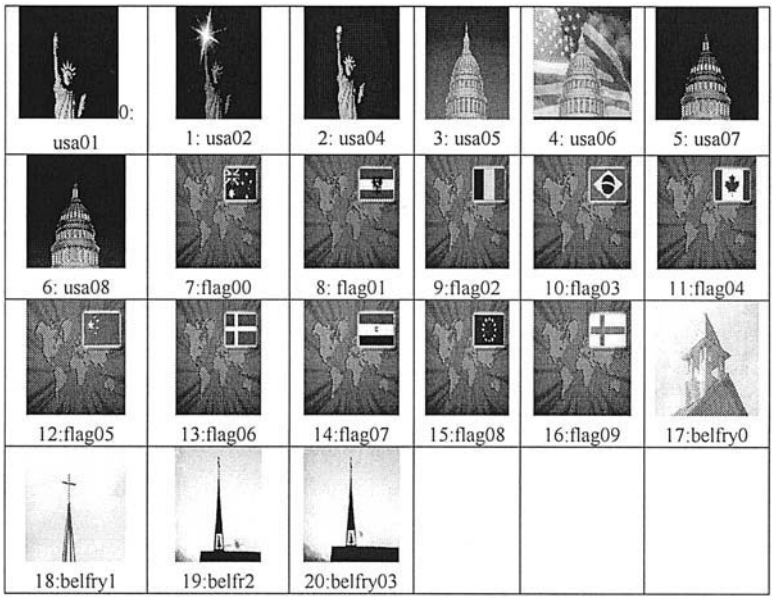

FIG. 27.2. Sample of a data set composed of 21 images.

Data Sets

We conducted 40 experiments on large image data sets covering a range of categories including panorama, scenery, flowers, and so on. The data sets came from collections compiled by IMSI Soft Company, which specializes in image libraries. These image data sets vary from 21 and 100,000 images of different resolutions. For example, the data set composed of 21 images is presented in Fig. 27.2. All images are cataloged into broad categories and each image carries an associated description. In this case the manual partition of image data sets into relevant clusters was feasible. The preclassification of images in semantic categories such as panorama and scenery helped our manual partition process. For all images we predetermined by hand a set of relevant clusters that constitutes a partition of image data sets. However, we may obtain several manual partitions that depend on our interpretation of image content. For example, in the sample of images in Fig. 27.3, manual clustering returns two "semantically correct" partitions. The first one (Table 27.2) is composed of two clusters (image references). It contains, respectively, U.S. symbols and flags. The second one (Table 27.3) is composed of five clusters (image references). For example, Cluster 1 (line 1 of Table 27.3) is composed of flag images = {flag00, flag01, flag02, Flag03, Flag04, flag05, flag06, flag07, Flag08, flag09}. It contains, respectively, flags, Statues of Liberty, Whites Houses, big belfries, and thin belfries. In Table 27.3 we consider essentially the visual proximity between images that is, cluster 1 = flag set, cluster 2 = Statue of Liberty set, cluster 3 = White House set, cluster 4 = big belfry set, and cluster 5 = thin belfry set. Of course, we can group all the belfries together, but because a person must cluster this data set without considering the semantic content of the data, when we consider just the visual proximity of data, the algorithm is not sure to group all belfries together. We can see that Belfry02 and belfry03 are very similar, and near belfry01. The flags usa01 usa02 and usa4 are also very similar. That is why they belong to the same clusters.

To summarize our approach, we can say that two clusterings are highlighted: manual and automatic clustering. Manual clustering generates several "correct" partitions. The clusters

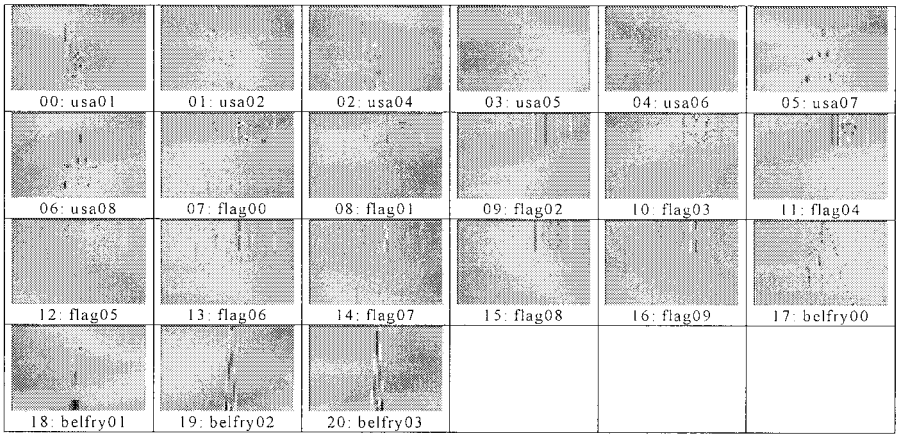

FIG. 27.3. Wavelet coefficients of experiment 4.

TABLE 27.2
First Manual Clustering of the Data Set Composed of 21 Images

Cluster 1	usa01	usa02	usa04	usa05	usa06	usa07	usa08	belfry00	belfry01	belfry02	belfry03
Cluster 2	flag00	flag01	flag02	flag03	flag04	flag05	flag06	flag07	flag08	flag09	

TABLE 27.3
Second Manual Clustering of the Data Set Composed of 21 Images

Cluster 1	flag00	flag01	flag02	flag03	flag04	flag05	flag06	flag07	flag08	flag09
Cluster 2	usa01	usa02	usa04							
Cluster 3	usa05	usa06	usa07	usa08						
Cluster 4	belfry00									
Cluster 5	belfry01	belfry02	belfry03							

obtained manually (image references) are compared with clusters obtained automatically on the basis of the following evaluation method.

Data Item Representation

Data items are represented by wavelet descriptors, and their proximity measures are based on Euclidean distance (for simplification). Image descriptors for indexing may be based on color histograms, anglogram, Fourier coefficients (frequency representation), and so on. The advantages of wavelet descriptors, compared with the previous descriptors are the representation of images with different resolutions, keeping in touch with the spatial and frequency information of the images.

The legitimate question is: how is the content of images represented by wavelet descriptors? Images are naturally represented by a two-dimensional spatial signal: abscissa (x) and ordinate (y). Although wavelet analysis could be generalized into a two-dimensional spatial signal, the computation involved would be time-consuming. Therefore, we consider the image signal as a one-dimensional linear signal. To maximize information details in the descriptors, we use four series of detail coefficients by scanning images in four different directions.

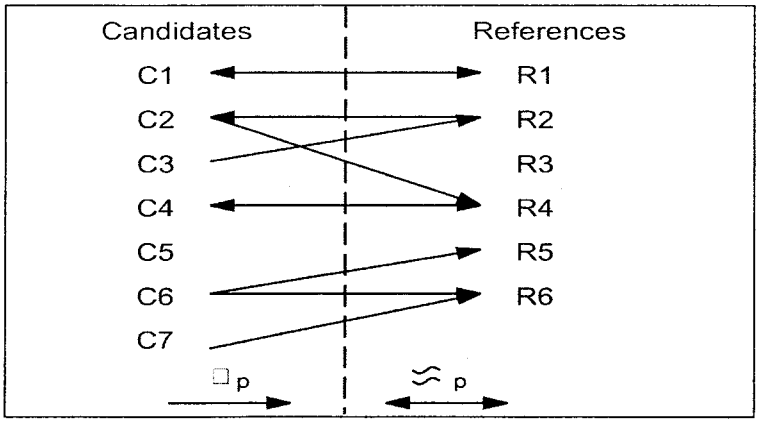

FIG. 27.4. Different cases of cluster matching.

With these directions we can extract horizontal, vertical, and diagonal remarkable frequency changes (i.e., horizontal, vertical, and diagonal contours). In addition, these different directions make the description of the image more accurate.

To speed up the clustering, we reduce the size of the descriptor vector. We choose a vector length equal to 2^n, with $n = 9, 10$. To reduce the feature vector to a smaller size, we apply the wavelet transform many times. In addition, we consider two reductions.

In the first reduction we loop on the lifting scheme; the input signal is the output signal of the previous iteration. When the signal is small enough (512, 1024, 2048), we store the wavelet coefficients as a feature vector.

In the second reduction the detail coefficients not saved in Fig. 27.4 contain information. How can we evaluate the lost information? In fact, in this approach we calculate all the detail coefficients and compute the average of each vector. Then, we reduce the vector with the higher average value. The reduction is the same as for the first method.

The higher the prediction degree of a wavelet, the more significant are the details of descriptors. Therefore, we experimented with Wavelet CDF (1,1): Haar and Wavelet CDF (2,2): Cohen-Daubechies-Feauveau (2,2). For Wavelet CDF (1,1): Haar, detail coefficients (or wavelet coefficients): $d_{j1,l} = s_{j,2l+1} - s_{j,2l}$; coarser signal: $s_{j1,l} = (s_{j,2l} + s_{j,2l+1})/2$. For Wavelet CDF (2,2): Cohen-Daubechies-Feauveau (2,2), detail coefficients (or wavelet coefficients): $d_{j1,l} = s_{j,2l+1} - (s_{j,2l} + s_{j,2l+2})/2$, coarser signal: $s_{j1,l} = s_{j,2l} + (d_{j1,l1} + d_{j1,l})/4$. We choose to reduce the size of the feature vector to 512, 1024, and 2048.

Evaluation Method

The clustering is evaluated by comparing automatic and manual clustering (image references). Manual clustering is expected. The evaluation method, inspired by Koschke and Eisenbarth (2000), considers the following input parameters: initial data set: I; reference clusters $R_i, \cup R_i = I$, R_i is obtained manually; candidate clusters C_i that are obtained automatically by the method presented in the previous section, $\cup C_i = I$. It also considers the following output measures: $overlap(C, R) = [(|C \cap R|)/(|C \cup R|)]$; affinity relationship $\approx_p$, $X \approx_p Y$ if and only if $overlap(X, Y) \geq p$; partial subset relationship $\subseteq_p$, $X \subseteq_p Y$ if and only if $[(|X \cap Y|)/(|X|)] \geq p$. P measures the degree of overlapping between two sets. A significant value of P means significant overlapping between two sets. And conversely a weak value of p means a weak overlapping between two sets.

The evaluation method compares the reference clusters with candidate clusters. So, for each couple of clusters (R_i, C_j), we obtain good matches: $R_i \approx_p C_j$; this matching is denoted $1 \sim 1$. Or, acceptable matches: $R_i \subseteq_p C_j$ or $C_j \subseteq_p R_i$ and not $R_i \approx_p C_j$. If $R_i \subseteq_p C_j$ then the candidate cluster C_j is too detailed. This case is denoted $n \sim 1$. Conversely, if $C_j \subseteq_p R_i$, then the candidate cluster has too few details. This case is denoted $1 \sim n$. There may also be false positive matches. Candidate clusters that neither match a reference nor are matched by any reference are called *false positives.* Inversely, they are *true negatives* when they are not even partially detected.

In Fig. 27.1 we have:

- Good matches $= \{(C_1, R_1), (C_4, R_4)\}, -\{n \sim 1\} = \{(R_2, C_2)\}, -\{1 \sim n\} = \{(C_3, R_2), (C_2, R_4), (C_6, R_5), (C_6, R_6), (C_7, R_6)\}$
- False positive $= \{C5\}$
- True negative $= \{R3\}$

The ideal evaluation scheme is composed exclusively of good matches. In this case, $\forall$ a cluster C, $\exists$ a reference R for which C $\approx_p$ R, and $\forall$ a reference R, $\exists$ a cluster C for which C $\approx_p$ R, with $p = 1.0$. However, a more realistic evaluation scheme considers a vector composed of the number of false positives, true negatives, the average accuracies of $1 \approx_p 1$, $1 \approx_p n$, $n \approx_p 1$ matches, and *overall recall rate,* with $p = 70\%$.

For two clusters, the accuracy between A and B, denoted $accuracy(A, B) = overlap(A, B)$, and for two sets of clusters: $accuracy(\{A_1, A_2, \ldots, A_a\}, \{B_1, B_2, \ldots, B_b\}) = overlap(\cup_{i=1}^{a} A_i, \cup_{i=1}^{b} B_i)$. So, for a class M of matches (M considers clusters in which we have: $1 \approx_p 1$, $1 \approx_p n$ or $n \approx_p 1$):

$$accuracy(M) = \frac{\sum_{(a,b)\in M} accuracy\,(a, b)}{card\,(M)}$$

Overall recall rate:

$$recall = \frac{\sum_{(a,b)\in GOOD} accuracy(a, b) + \sum_{(a,b)\in OK} accuracy\,(a, b)}{card\,(GOOD) + card\,(OK) + card\,(true\ \ negative)}$$

where *card* is the cardinality of a set. In other words, it corresponds to the number of data items in the set.

Results and Analysis

We considered 40 experiments with wavelet CDF (1, 1), wavelet CDF (2, 2), monochrome, RGB colors with descriptor size equal to 512 and 2048 elements, and data sets of respectively 21, 100, 1,000, 10,000 and 100,000 images. We obtained the results shown in Table 27.1.

In the column color, RGB means "red, green, blue," so these are color images. N/B means "black and white" images. To illustrate the meaning of an experiment, we detail experiment 4 (see Table 27.4).

The descriptor size is 2048, wavelet type is CDF (2, 2) (Cohen et al., 1992), color information is RGB, and the cardinality of the data set is equal to 21 images. In Fig. 27.2 we present a visual representation of wavelet image descriptors of the data set presented in the Fig. 27.1. We can see using these visual images how multiresolution representation based on wavelets

TABLE 27.4
Experiments

Experiment	*Size of Descriptor*	*Wavelet Type*	*Color*	*Data Sets*
1	2048	CDF(1,1)	N/B	21 images
2	2048	CDF(1,1)	RGB	21 images
3	2048	CDF(2,2)	N/B	21 images
4	2048	CDF(2,2)	RGB	21 images
5	512	CDF(1,1)	N/B	21 images
6	512	CDF(1,1)	RGB	21 images
7	512	CDF(2,2)	N/B	21 images
8	512	CDF(2,2)	RGB	21 images
9	2048	CDF(1,1)	N/B	100 images
10	2048	CDF(1,1)	RGB	100 images
11	2048	CDF(2,2)	N/B	100 images
12	2048	CDF(2,2)	RGB	100 images
13	512	CDF(1,1)	N/B	100 images
14	512	CDF(1,1)	RGB	100 images
15	512	CDF(2,2)	N/B	100 images
16	512	CDF(2,2)	RGB	100 images
17	2048	CDF(1,1)	N/B	1000 images
18	2048	CDF(1,1)	RGB	1000 images
19	2048	CDF(2,2)	N/B	1000 images
20	2048	CDF(2,2)	RGB	1000 images
21	512	CDF(1,1)	N/B	1000 images
22	512	CDF(1,1)	RGB	1000 images
23	512	CDF(2,2)	N/B	1000 images
24	512	CDF(2,2)	RGB	1000 images
25	2048	CDF(1,1)	N/B	10000 images
26	2048	CDF(1,1)	RGB	10000 images
27	2048	CDF(2,2)	N/B	10000 images
28	2048	CDF(2,2)	RGB	10000 images
29	512	CDF(1,1)	N/B	10000 images
30	512	CDF(1,1)	RGB	10000 images
31	512	CDF(2,2)	N/B	10000 images
32	512	CDF(2,2)	RGB	10000 images
33	2048	CDF(1,1)	N/B	100000 images
34	2048	CDF(1,1)	RGB	100000 images
35	2048	CDF(2,2)	N/B	100000 images
36	2048	CDF(2,2)	RGB	100000 images
37	512	CDF(1,1)	N/B	100000 images
38	512	CDF(1,1)	RGB	100000 images
39	512	CDF(2,2)	N/B	100000 images
40	512	CDF(2,2)	RGB	100000 images

discriminates the outline of the image. The wavelet descriptor is a suitable signature of images, particularly when the images support variations of textures and colors.

As mentioned previously, the data set may be manually classified in two or five clusters. So we may have two or five image references. Then we activate the clustering method, which is the focus of this chapter, to find the best values of k. To limit the computing time, we introduce the born [2, 8]. This born means that $8 \geq k \geq 2$. So, k have to be between 2 and 8. Tables 27.5–27.10 show the LVRC and GLVRC values for each cluster.

The best value of k is 2 with LVRC $= 22.88\%$, GVRCN $= 15\%$ and with 2 (good+ acceptable) matches with image references. More generally, we notice that for all experiments the best value of k is 2 (see Tables 27.11 and 27.12).

TABLE 27.5
$K = 3$ Clusters: (LVRC: 0.160459, GVRCN: 0.35)

Silhouette Coefficient	*Medoid*	*Data Items*								
0.163170	usa05	Usa01	usa02	usa04	usa06	usa07	usa08	belfry00	belfry01	belfry02
0.318206	flag05	flag00	Flag01	flag02	Flag03	flag04	flag06	flag07	flag08	flag09
0.000000	Belfry03									

TABLE 27.6
Four Clusters: (LVRC: 0.100644, GVRCN = 0.38)

	Silhouette Coefficient	*Medoid*	*Data Items*								
1	0.162024	usa05	usa01	usa02	usa04	Usa06	usa07	usa08	belfry00	belfry01	belfry02
2	0.209901	flag05	flag00	flag02	flag03	flag07					
3	0.000000	belfry03									
4	0.030651	flag08	flag01	flag04	flag06	flag09					

TABLE 27.7
Five Clusters: (LVRC: 0.087042, GVRCN = 0.05)

	Silhouette Coefficient	*Medoid*	*Data Items*							
1	0.194658	usa05	usa01	Usa02	usa04	Usa06	usa07	usa08	belfry00	belfry01
2	0.209901	flag05	flag00	Flag02	flag03	Flag07				
3	0.000000	belfry03								
4	0.030651	flag08	flag01	Flag04	flag06	Flag09				
5	0.000000	belfry02								

TABLE 27.8
Six Clusters: (LVRC: 0.079843, GVRCN = 0.43)

	Silhouette Coefficient	*Medoid*	*Data Items*						
1	0.194658	usa05	usa01	usa02	usa04	Usa06	usa07	usa08	belfry01
2	0.209901	flag05	flag00	flag02	flag03	flag07			
3	0.000000	belfry03							
4	0.030651	flag08	flag01	flag04	flag06	flag09			
5	0.000000	Belfry02							
6	0.000000	Belfry00							

TABLE 27.9
Seven Clusters: (LVRC: 0.112124, GVRCN = 0.51)

	Silhouette Coefficient	*Medoid*	*Data Items*				
1	−0.008364	usa05	usa01	usa02	usa04	Usa06	belfry01
2	0.209901	flag05	flag00	flag02	flag03	flag07	
3	0.000000	belfry03					
4	0.030651	flag08	flag01	flag04	flag06	flag09	
5	0.000000	belfry02					
6	0.000000	belfry00					
7	0.552677	usa8	usa7				

TABLE 27.10
Eight Clusters: (LVRC: 0.099835, GVRCN = 0.7)

	Silhouette Coefficient	*Medoid*	*Data Items*			
1	0.021699	usa05	usa01	usa02	usa04	usa06
2	0.209901	flag05	flag00	flag02	flag03	flag07
3	0.000000	belfry03				
4	0.030651	flag08	flag01	flag04	flag06	flag09
5	0.000000	belfry02				
6	0.000000	belfry00				
7	0.536432	usa8	usa7			
8	0.000000	belfry01				

TABLE 27.11
Matches with Two References

2		1,000000	0	1	0.181818	0	0	0.727273
1		1,000000	0	1	0.55	0	0	0.7
2		1,000000	0	1	0.181818	0	0	0.727273
2		1,000000	0	0		0	0	1.0
2		1,000000	0	1	0.181818	0	0	0.727273
2	0.908333		0	1	0.181818	0	0	0.666162
2	1.0		0	0		0	0	1.0
2	1.0		0	0		0	0	1.0

TABLE 27.12
Matches with Five References

1	1	1	0.888889	1	0.666667	0	0	0.851852
1	1	1	0.571429	3	0.422222	0	0	0.567619
1	1	1	0.777778	1	0.5	0	0	0.759259
1	1	1	1	0		0	0	1
1	1	1	0.888889	1	0.666667	0	0	0.851852
1	0.9	1	0.8	1	0.666667	0	0	0.788889
1	1	1	1	0	0		0	1
1	1	1	1	0	0		0	1

We compare here the experiments 1–8 on the basis of overall recall rate with two image references and five image references respectively. So experiments 4, 7, and 8 return cluster results very close to those of manual clustering. The overcall rate is near 1. The first conclusion is that the CDF (2,2) wavelet returns better results than CDF (1,1). This is due to the fact that CDF (2,2) attenuates the noise of images. Only real and hard changes in the color of the pictures are noticed. For example, a CDF (2,2) wavelet is not perturbed by a sky color gradation. When comparing experiments on the basis of good matches and acceptable matches to see which automatic clustering best matches our manual clustering, we obtain the results presented in the Fig. 27.5. All experiments return two (acceptable+good) matches. In this case there is no $1 \sim n$ matche (because there is no clustering less detailed than a clustering with two clusters).

Experiments 4, 7, and 8 all give the same results: two clusters. But if they return results comparable with our five-clusters clustering, that is because the manual clusters are totally included in another cluster. So our manual clustering is just more precise. The evaluation method looks for misplaced objects. That is why experiments 4, 7, and 8 return good results.

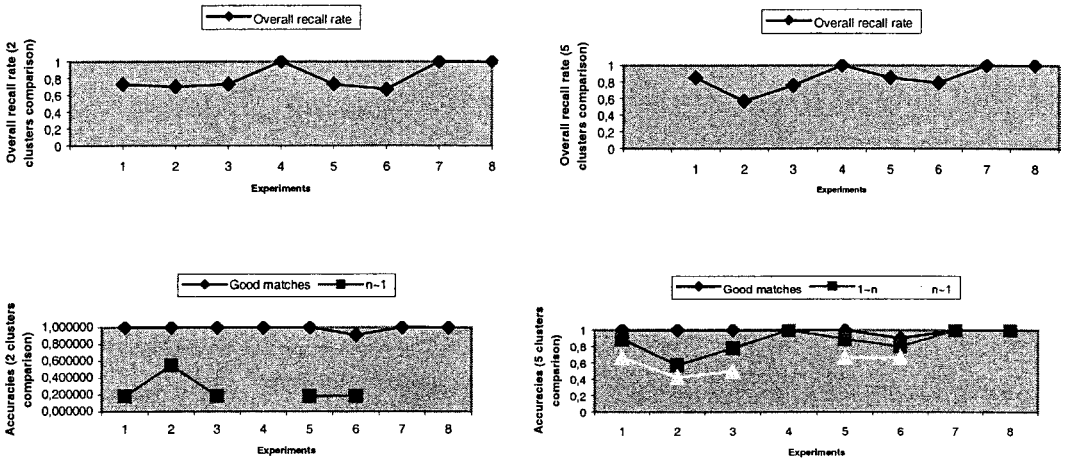

FIG. 27.5. Experiments 1–8.

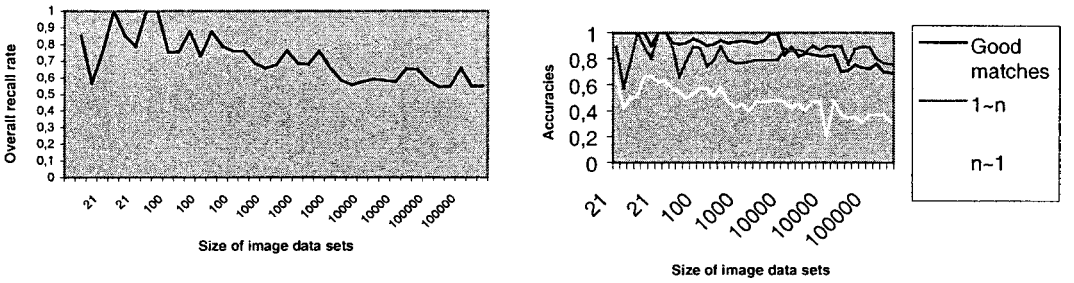

FIG. 27.6. Experiments 1–40.

Figure 27.6 illustrates the evolution of accuracies and overall recall rate for the 40 experiments. We notice the light degradation of accuracies and overall recall rate for very large image data sets. However, the measure $1 \sim n$ presents stable performance. This may be explained by the fact that the clusters obtained automatically are included in image references.

SUMMARY

This chapter focuses on k-discovery in the context of a partition method in voluminous data items. Partitions organize data item repositories (usually represented by vectors of values) into clusters based on similarity. More particularly, this chapter presents a method that automatically discovers the number of clusters (k). We tested the method on image repositories. Our strategy is based on the confidence measures composed of intercluster confidence (GVRCN) that considers the confidence of the whole cluster, and intracluster confidences (LVRC) that considers the confidence of individual cluster. We showed that the two levels of confidences present interesting results. Results of experiments showed that the value of k obtained automatically is generally the better one, if we consider the confidence measure computed on the basis of GVRCN and LVRC. So the value of k corresponds to clustering results that have in the

first step, the best GVRCN, and in the second step the best LVRC. Many current approaches consider only LVRC known by silhouette. In these cases the best values of k did not mean the best clusters, when compared with image references (manual clusters).

Many efforts should be focused to speed up the k-discovery process for very large data items. Our experiments showed that when considering very large data items, the algorithm of k-discovery remains time-consuming.

The experiments showed, too, that the type of wavelet coefficients considered—wavelet CFD (2,2)—have more influence on the final results than the levels of resolution (512 or 2048 wavelet coefficients) or color/gray parameters. So the clustering process in which image descriptors are based (wavelet CFD [2,2]) returns results that are better than wavelet CFD (1,1), independently of level or resolutions or color/gray parameters.

The overall experiment results showed that the confidence measures of clusters are close to image references (clusters obtained manually). In these cases is it sufficient to ensure that wavelet descriptors and the k-discovery method are well suitable to cluster semantically images? This question is not easy to answer. However, we can say that wavelet descriptors are suitable to discriminate images by considering levels of resolutions; however, it is not clear at all whether there is a causal link between levels of resolutions and semantic content of clusters.

REFERENCES

Berchtold, S., Keim, D. A., & Kriegel, H.-P. (1996). The X-tree: An index structure for high dimensional data, *Proceedings of the 22nd International Conference on Very Large Data Bases* (pp. 28–39.) San Francisco: Morgan Kaufmann.

Calinski, T., & Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics, 3,* 1–27.

Cohen, A., Daubechies, I., & Feauveau, J. (1992). Bi-orthogonal bases of compactly supported wavelets. *Communications on Pure Applied Mathematics, 45,* 485–560.

Diday, E. (1973). The dynamic cluster method on non-hierarchical clustering. *Journal of Information Science, 2,* 61–88.

Dubes, R. C. (1987). How many clusters are best?—An experiment. *Pattern Recognition Journal, 20,* 645–663.

Hinneburg, A., & Keim, D. (1999). Optimal grid-clustering: Towards breaking the curse of dimensionality. In *Proceedings of the 25th International Conference on Very Large Data Bases* (pp. 506–517). San Francisco: Morgan Kaufmann.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys, 31,* 264–323.

Kaufman, L., & Rousseeuw, P. J. (Eds.). (1990). *Finding groups in data: An introduction to cluster analysis.* New York: J. Wiley.

Koschke, R., & Eisenbarth, T. (2000). A framework for experimental evaluation of clustering techniques. In *International Workshop on Program Comprehension (IWPC '2000).*

McQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Fifth Berkeley Symposium on Mathematical Statistics and Probability* (pp. 281–297). Berkeley, CA: University of California.

Milligan, G. W., & Cooper, M. C. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika, 50,* 159–179.

Raymond, T. N., & Jiawei, H. (1994). *Efficient and effective clustering methods for spatial data mining* (Tech. Rep. TR-94-13). University of British Columbia, Vancouver, B.C., Canada.

Sheikholeslami, G., & Chatterjee Surojit, Z. A. (2000). WaveCluster: A wavelet-based clustering approach for spatial data in very large databases. *VLDB Journal, 8,* 289–304.

Vinod, H. D. (1969). Integer programming and the theory of grouping. *Journal of the American Statistical Association, 64,* 506–519.

Wand, W., Yang, J., & Muntz, R. (1997). STING: A statistical information grid approach to spatial data mining. In *Proceedings of the 23rd Int. Conf. on Very Large Data Bases,* Santa Clara, CA: Morgan Kaufmann.

Zhang, T., Ramakrishnan, R., & Linvy, M. (1996). BIRCH: An efficient data clustering method for very large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (pp. 103–114). New York: ACM Press.