A new Interactive Semi-Supervised Clustering model for large image database indexing

LAI Hien Phuong[a,b], Muriel VISANI[a], Alain BOUCHER[a,b], Jean-Marc OGIER[a]

[a] *L3I, Université de La Rochelle, Avenue M. Crépeau, 17042 La Rochelle cedex 1, France*
[b] *IFI, Equipe MSI; IRD, UMI 209 UMMISCO*
Institut de la Francophonie pour l'Informatique, 42 Ta Quang Buu, Hanoi, Vietnam
Vietnam National University, Hanoi, Vietnam

Abstract

Indexing methods play a very important role in finding information in large image databases. They organize indexed images in order to facilitate, accelerate and improve the results for later retrieval. Alternatively, clustering may be used for structuring the feature space so as to organize the dataset into groups of similar objects without prior knowledge (unsupervised clustering) or with a limited amount of prior knowledge (semi-supervised clustering).

In this paper, we introduce a new interactive semi-supervised clustering model where prior information is integrated via pairwise constraints between images. The proposed method allows users to provide feedback in order to improve the clustering results according to their wishes. Different strategies for deducing pairwise constraints from user feedback were investigated. Our ex-

Email addresses: `Corresponding Author: L3I, Université de La Rochelle, Avenue M. Crépeau, 17042 La Rochelle cedex 1, France; Phone: +33 6 46 51 12 32; Fax: +33 5 46 45 82 42; Email: hien_phuong.lai@univ-lr.fr` (LAI Hien Phuong), `muriel.visani@univ-lr.fr` (Muriel VISANI), `alainboucher12@gmail.com` (Alain BOUCHER), `jean-marc.ogier@univ-lr.fr` (Jean-Marc OGIER)

periments on different image databases (Wang, PascalVoc2006, Caltech101) show that the proposed method outperforms semi-supervised HMRF-kmeans (Basu et al., 2004).

Keywords: Semi-supervised clustering, Interactive learning, Image indexing

1. Introduction

Content-Based Image Retrieval (CBIR) refers to the process which uses visual information (usually encoded using color, shape, texture feature vectors, etc.) to search for images in the database that correspond to the user's queries. Traditional CBIR systems generally rely on two phases. The first phase is to extract the feature vectors from all the images in the database and to organize them into an efficient index data structure. The second phase is to efficiently search in the indexed feature space to find the most similar images to the query image.

With the development of many large image databases, an exhaustive search is generally intractable. Feature space structuring methods (normally called indexing methods) are therefore necessary for facilitating and accelerating further retrieval. They can be classified into space partitioning methods and data partitioning methods.

Space partitioning methods (KD-tree (Bentley, 1975), KDB-tree (Robinson, 1981), LSD-tree (Henrich et al., 1989), Grid-File (Nievergelt et al., 1988)...) generally divide the feature space into cells (sometimes referred to as "buckets") of fairly similar cardinality (in terms of number of images per cell), without taking into account the distribution of the images in the feature space. Therefore, dissimilar points may be included in a same cell while similar points may end up in different cells. The resulting index is therefore not optimal for retrieval, as the user generally wants to retrieve similar images to the query image. Moreover, these methods are not designed to handle high dimensional data, while image feature vectors commonly count hundreds of elements.

Data partitioning methods (B-tree (Bayer and McCreight, 1972), R-trees (Guttman, 1984; Sellis et al., 1987; Beckmann et al., 1990), SS-tree (White and Jain, 1996), SR-tree (Katayama and Satoh, 1997), X-tree (Berchtold et al., 1996)...) also integrate information about image distribution in the feature space. However, the limitations on the cardinality of the space cells remain, causing the resulting index to be non-optimal for retrieval, especially in the case where groups of similar objects are unbalanced, *i.e.* composed of different numbers of images.

Our claim is that using clustering instead of traditional indexing to organize feature vectors, results in indexes better adapted to high dimensional and unbalanced data. Indeed, clustering aims to split a collection of data into groups (clusters) so that similar objects belong to the same group and dissimilar objects are in different groups, with no constraints on the cluster size. This makes the resulting index better optimized for retrieval. In fact, while in traditional indexing methods it might be difficult to fix the number of objects in each bucket (especially in the case of unbalanced data), clustering methods have no limitation on the cardinality of the clusters, objects can be grouped into clusters of very different sizes. Moreover, using clustering might simplify the relevance feedback task, as the user might interact with a small number of cluster prototypes rather than numerous single images.

Because feature vectors only capture low level information such as color, shape or texture, there is a semantic gap between high-level semantic concepts expressed by the user and these low-level features. The clustering results are therefore generally different from the intent of the user. Our work aims to involve users in the clustering phase so that they can interact with the system in order to improve the clustering results. The clustering methods should therefore produce a hierarchical cluster structure where the initial clusters may be easily merged or split. We are also interested in clustering methods which can be incrementally built in order to facilitate the insertion or deletion of new images by the user. It can be noted that incrementality is also very important in the context of huge image databases, when the whole dataset cannot be stored in the main memory. Another very important point is the computational complexity of the clustering algorithm, especially in an interactive online context where the user is involved.

In the case of large image database indexing, we may be interested in traditional clustering (unsupervised) (Jain et al., 1999; Xu and Wunsch, 2005) or semi-supervised clustering (Basu et al., 2002; Dubey et al., 2010; Wagstaff et al., 2001; Basu et al., 2004). While no information about ground truth is provided in the case of unsupervised clustering, a limited amount of knowledge is available in the case of semi-supervised clustering. The provided knowledge may consist of class labels (for some objects) or pairwise constraints (must-link or cannot-link) between objects.

In (Lai et al., 2012a), we proposed a survey of unsupervised clustering techniques and analyzed the advantages and disadvantages of different methods in a context of huge masses of data where incrementality and hierarchical structuring are needed. We also experimentally compared five methods (global k-means (Likas et al., 2003), AHC (Lance and Williams, 1967), R-tree (Guttman, 1984), SR-tree (Katayama and Satoh, 1997) and BIRCH (Zhang et al., 1996)) with different real image databases of increasing sizes (Wang, PascalVoc2006, Caltech101, Corel30k) (the number of images ranges from 1,000 to 30,000) to study the scalability of different approaches relative to the size of the database. In (Lai et al., 2012b), we presented an overview of semi-supervised clustering methods and proposed a preliminary experiment of an interactive semi-supervised clustering model using the HMRF-kmeans (Hidden Markov Random Fields kmeans) clustering (Basu et al., 2004) on the Wang image database in order to analyze the improvement in the clustering process when user feedback is provided.

There are three main parts to this paper. Firstly, we propose a new interactive semi-supervised clustering model using pairwise constraints. Secondly, we investigate different methods for deducing pairwise constraints from user feedback. Thirdly, we experimentally compare our proposed semi-supervised method with the widely known semi-supervised HMRF-kmeans method.

This paper is structured as follows. A short review of semi-supervised clustering methods is presented in Section 2. Our interactive semi-supervised clustering model is proposed in Section 3. Some experiments are presented in Section 4. Some conclusions and further works are provided in Section 5.

2. A short review of semi-supervised clustering methods

For unsupervised clustering only similarity information is used to organize objects; in the case of semi-supervised clustering a small amount of prior knowledge is available. Prior knowledge is either in the form of class labels (for some objects) or pairwise constraints between objects. Pairwise constraints specify whether two objects should be in the same cluster (must-link) or in different clusters (cannot-link). As the clusters produced by unsupervised clustering may not be the ones required by the user, this prior knowledge is needed to guide the clustering process for resulting clusters which are closer to the user's wishes. For instance, for clustering a database with thousands of animal images, an user may want to cluster by animal species or by background landscape types. An unsupervised clustering method may give, as a result, a cluster containing images of elephants with a grass background together with images of horses with a grass background and another cluster containing images of elephants with a sand background. These results are ideal when the user wants to cluster by background landscape types. But they are poor when the user wants to cluster by animal species. In this case, must-link constraints between images of elephants with a grass background and images of elephants with a sand background and cannot-link constraints between images of elephants with a grass background and images of horses with a grass background are needed to guide the clustering process. The objective of our work is to make the user interact with the system so as to define easily these constraints with only a few clicks. Note that the available knowledge is too poor to be used with supervised learning, as only a very limited ratio of the available images are considered by the user at each step. In general, semi-supervised clustering methods are used to maximize intra-cluster similarity, to minimize inter-cluster similarity and to keep a high consistency between partitioning and domain knowledge.

Semi-supervised clustering has been developed in the last decade and some methods have been published to date. They can be divided into semi-supervised clustering with labels, where partial information about object labels is given, and semi-supervised clustering with constraints, where a small amount of pairwise constraints between objects is given.

Some semi-supervised clustering methods using labeled objects have been put forward: seeded-kmeans (Basu et al., 2002), constrained-kmeans (Basu et al., 2002), etc. Seeded-kmeans and constrained-kmeans are based on the k-means algorithm. Prior knowledge for these two methods is a small subset of the input database, called seed set, containing user-specified labeled objects of k different clusters. Unlike k-means algorithm which randomly selects the initial cluster prototypes, these two methods use the labeled objects to initialize the cluster prototypes. Following this we repeat, until convergence, the re-assignment of each object in the dataset to the nearest prototype and the re-computation of the prototypes with the assigned objects. The seeded-kmeans assigns objects to the nearest prototype without considering the prior labels of the objects in the seed set. In contrast, the constrained-kmeans maintains the labeled examples in their initial clusters and assigns the other objects to the nearest prototype. An interactive cluster-level semi-supervised clustering was proposed in (Dubey et al., 2010) for document analysis. In this model, knowledge is progressively provided as assignment feedback and cluster description feedback after each interactive iteration. Using assignment feedback, the user moves an object from one cluster to another cluster. Using cluster description feedback, the user modifies the feature vector of any current cluster (*e.g.* increase the weighting of some important words). The algorithm learns from all the feedback to re-cluster the dataset in order to minimize average distance between points and their cluster centers while minimizing the violation of constraints corresponding to feedback.

Among the semi-supervised clustering methods using pairwise constraints between objects, we can cite COP-kmeans (constrained-kmeans) (Wagstaff et al., 2001), HMRF-kmeans (Hidden Markov Random Fields Kmeans) (Basu et al., 2004), semi-supervised kernel-kmeans (Kulis et al., 2005), etc. The input data of these methods is data set X, a set of must-link constraints M and a set of cannot-link constraints C. In COP-kmeans, points are assigned to clusters without violating any constraint. A point x_i is assigned to its closest cluster μ_j unless a constraint is violated. If x_i cannot be placed in μ_j, we continue attempting to assign x_i to the next cluster in the sorted list of clusters by ascending order of distances with x_i until a suitable cluster is found. The clustering fails if no solution respecting the constraints is found. While the constraint violation is strictly prohibited in COP-kmeans, it is allowed with a violation cost (penalty) in HMRF-kmeans and in semi-supervised kernel-kmeans. The objective function to be minimized in the semi-supervised HMRF-kmeans is as follows:

$$J_{HMRF\_Kmeans} = \sum_{x_i \in X} D(x_i, \mu_{l_i}) + \sum_{(x_i,x_j) \in M, l_i \neq l_j} w_{ij} + \sum_{(x_i,x_j) \in C, l_i = l_j} \overline{w_{ij}} \quad (1)$$

where w_{ij} ($\overline{w_{ij}}$) is the penalty cost for violating a must-link (cannot-link) constraint between x_i and x_j, l_i refers to the cluster label of x_i, and $D(x_i, \mu_{l_i})$ measures the distance between x_i and its corresponding cluster center μ_{l_i}. The violation cost of a pairwise constraint may be either a constant or a function of the distance between the two points specified in the pairwise constraint as follows:

$$w_{ij} = wD(x_i, x_j) \tag{2}$$

$$\overline{w_{ij}} = \overline{w}(D_{max} - D(x_i, x_j)) \tag{3}$$

where w and $\overline{w}$ are constants specifying the cost for violating a must-link or a cannot-link constraint. D_{max} is the maximal distance between two points in the data set. We can see that, to ensure the most difficult constraints are respected, higher penalties are assigned to violations of must-link constraints between points which are distant and to violations of cannot-link constraints between points which are close. The term D_{max} in Equation (3) can make the cannot-link penalty term sensitive to extreme outliers, but all cannot-link constraints are treated in the same way, so even in the presence of extreme outliers, there would be no cannot-link constraint favored compared to the others. The objective function in Equation (1) is also sensitive to outliers. We can reduce this sensitivity by using an outlier filtering technique or by replacing the term D_{max} by the maximum distance between two clusters. HMRF-kmeans first initializes the k cluster centers based on user-specified constraints, as described in (Basu et al., 2004). After the initialization step, an iterative relocation approach similar to k-means is applied to minimize the objective function. The iterative algorithm represents the repetition of the assignment phase of each point to the cluster which minimizes its contribution to the objective function and the re-estimation phase of the cluster centers minimizing the objective function. The semi-supervised kernel-kmeans (Kulis et al., 2005) is similar to the HMRF-kmeans, but calculates the objective function in a transformed space instead of the original space using a kernel function mapping as follows:

$$J_{SS\_Kernel\_Kmeans} = \sum_{x_i \in X} \|\phi(x_i) - \overline{\phi_{l_i}}\|^2 - \sum_{(x_i,x_j) \in M, l_i = l_j} w_{ij} + \sum_{(x_i,x_j) \in C, l_i = l_j} \overline{w_{ij}} \quad (4)$$

where $\phi(x_i)$ is the kernel function mapping, $\overline{\phi_{l_i}}$ is the centroid of the cluster containing x_i and w_{ij} ($\overline{w_{ij}}$) is the penalty cost for violating a must-link (cannot-link) constraint between x_i and x_j. In the second term of Equation (4), instead of adding a penalty cost for a must-link violation if the two points are in different clusters, Kulis et al. (2005) give a reward for must-link constraint satisfaction if the two points are in the same cluster, by subtracting the corresponding penalty term from the objective function.

3. Proposed interactive semi-supervised clustering model

In this section, we present our proposed interactive semi-supervised clustering model. In our model, the initial clustering is carried out without any prior knowledge, using an unsupervised clustering method. In Lai et al. (2012a) we discussed the adequation between different unsupervised clustering methods and our applied context (involving user interactivity) as well as experimentally compared different unsupervised clustering methods (global k-means (Likas et al., 2003), AHC (Lance and Williams, 1967), R-tree (Guttman, 1984), SR-tree (Katayama and Satoh, 1997), BIRCH (Zhang et al., 1996)). Our conclusion was that BIRCH is the most suitable to our context. BIRCH is less sensitive to variations in its parameters. Moreover, it is incremental, it provides a hierarchical structure of clusters and it outperforms other methods in the context of a large database (best results and best computational time in our tests). Therefore, BIRCH is chosen for the initial unsupervised clustering in our model. After the initial clustering, the user views the clustering results and provides feedback to the system. The pairwise constraints (must-link, cannot-link) are deduced, based on user feedback; the system then re-organizes the clusters by considering the constraints. The re-clustering process is done using the proposed semi-supervised clustering described in Section 3.2. The interactive process (user provides feedback and system reorganizes the clusters) is repeated until the clustering result satisfies the user. The interactive semi-supervised clustering model contains the following steps:

1. Initial clustering using BIRCH unsupervised clustering.
2. *Repeat*:
   (a) Receive feedback from the user and deduce pairwise constraints.
   (b) Re-organize the clusters using the proposed semi-supervised clustering method.

   until the clustering result satisfies the user.

3.1. BIRCH unsupervised clustering

Let us briefly describe the BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) unsupervised clustering method (Zhang et al., 1996). The idea of BIRCH is to build a Clustering Feature Tree (CF-tree).

We define a CF-vector, summarizing information of a cluster including N vectors $(\overrightarrow{x_1}, ..., \overrightarrow{x_N})$, as a triplet $CF = (N, \overrightarrow{LS}, SS)$, where $\overrightarrow{LS}$ and SS are respectively the linear sum and the square sum of vectors ($\overrightarrow{LS} = \sum_{i=1}^{N} \overrightarrow{x_i}$; $SS = \sum_{i=1}^{N} \overrightarrow{x_i}^2$). From the CF-vectors, we can simply compute the centroid, the radius (average distance from points to the centroid) of a cluster and also the distance between two clusters (*e.g.* the Euclidean distance between their centroids). A CF-tree is a balanced tree having three parameters B, L and T:

- Each internal node contains, at most, B elements of the form $[CF_i, child_i]$ where $child_i$ is a pointer to its i$^{\text{th}}$ child node and CF_i is the CF-vector of this child.

- Each leaf node contains, at most, L entries of the form $[CF_i]$, it also contains two pointers, *prev* and *next*, to link leaf nodes.

- Each entry CF_i represents the information of a group of points which are close together. Each entry CF_i of a leaf node must have a radius lower than a threshold T (threshold condition).

The CF-tree is created by successively inserting points into the tree. A new point is preferably inserted in the closest CF_i of the closest leaf, if the threshold condition is not violated. If it is impossible, a new CF_j is created for the new point. The corresponding internal and leaf nodes must be split if necessary. After creating the CF-tree, we can use any clustering method (AHC, k-means, etc.) to cluster all leaf entries CF_i. In our work, we use k-means for clustering the leaf entries, as it is suitable to be used with our proposed semi-supervised clustering in the interactive phase.

3.2. Proposed semi-supervised clustering method

At each interactive iteration, our semi-supervised clustering method is applied after receiving feedback from the users for re-organizing the clusters according to their wishes. Our semi-supervised clustering method considers the set of all leaf entries $S_{CF} = (CF_1, ..., CF_m)$ of the CF-tree. Supervised information is provided as two sets of pairwise constraints between CF entries deduced from user feedback: must-links $M_{CF} = \{(CF_i, CF_j)\}$ and cannot-links $C_{CF} = \{(CF_i, CF_j)\}$. $(CF_i, CF_j) \in M_{CF}$ implies that CF_i, CF_j and therefore all points which are included in these two entries should belong to the same cluster, while $(CF_i, CF_j) \in C_{CF}$ implies that CF_i and CF_j should belong to different clusters. The objective function to be minimized is as follows:

$$\begin{aligned} J_{obj} &= \sum_{CF_i \in S_{CF}} D(CF_i, \mu_{l_i}) \\ &+ \sum_{(CF_i, CF_j) \in M_{CF}, l_i \neq l_j} w N_{CF_i} N_{CF_j} D(CF_i, CF_j) \\ &+ \sum_{(CF_i, CF_j) \in C_{CF}, l_i = l_j} \overline{w} N_{CF_i} N_{CF_j} (D_{max} - D(CF_i, CF_j)) \end{aligned} \tag{5}$$

where:

- The first term measures the distortion between each leaf entry CF_i and the corresponding cluster center μ_{l_i}, l_i refers to the cluster label of CF_i.
- The second and the third terms represent the penalty costs for respectively violating the must-link and cannot-link constraints between CF entries. w and $\overline{w}$ are constants specifying the violation cost of a must-link and a cannot-link between two points. As an entry CF_i represents the information of a group of N_{CF_i} points, a pairwise constraint between two entries CF_i and CF_j corresponds to $N_{CF_i} \times N_{CF_j}$ constraints between points of these two entries. The violation cost of

a pairwise constraint between two entries CF_i, CF_j is thus a function of their distance $D(CF_i, CF_j)$ and of the number of points included in these two entries. D_{max} is the maximum distance between two CF entries in the data set. Therefore, higher penalties are assigned to violations of must-link between entries that are distant and of cannot-link between entries which are close. As in HMRF-kmeans, the term D_{max} can make the cannot-link penalty term sensitive to extreme outliers, and could be replaced by the maximum distance between two clusters if the database contains extreme outliers.

In our case, we use the most frequently used squared Euclidean distance as distortion measure. The distance between two entries $CF_i = (N_{CF_i}, \overrightarrow{LS}_{CF_i}, SS_{CF_i})$, $CF_j = (N_{CF_j}, \overrightarrow{LS}_{CF_j}, SS_{CF_j})$ is calculated as the distance between their means as follows:

$$D(CF_i, CF_j) = \sum_{p=1}^{d} \left(\frac{LS_{CF_i}(p)}{N_{CF_i}} - \frac{LS_{CF_j}(p)}{N_{CF_j}} \right)^2 \tag{6}$$

where d is the number of dimensions of the feature space.

The proposed semi-supervised clustering is as follows:

Input: Set of leaf entries $S_{CF} = \{CF_i\}_{i=1}^{m}$ which are clustered into K clusters with the corresponding centroids $\{\mu_h\}_{h=1}^{K}$,

set of must-link constraints $M_{CF} = \{(CF_i, CF_j)\}$

set of cannot-link constraints $C_{CF} = \{(CF_i, CF_j)\}$.

Output: New disjoint K clusters of S_{CF} such that the objective function in Equation (5) is locally minimized.

Method:

1. Set $t \leftarrow 0$

2. Repeat until convergence

   (a) Re-assignment step: Given $\{\mu_h^{(t)}\}_{h=1}^K$, re-assign cluster labels $\{l_i^{(t+1)}\}_{i=1}^m$ of entries $\{CF_i\}_{i=1}^m$ to minimize the objective function.

   (b) Re-estimation step: Given cluster labels $\{l_i^{(t+1)}\}_{i=1}^m$, re-calculate the cluster centroids $\{\mu_h^{(t+1)}\}_{h=1}^K$ to minimize the objective function.

   (c) $t \leftarrow t+1$.

In the re-assignment step, given the current cluster centers, each entry CF_i is re-assigned to the cluster μ_h which minimizes its contribution to the objective function as follows:

$$\begin{aligned} J_{obj}(CF_i, \mu_h) &= D(CF_i, \mu_h) \\ &+ \sum_{(CF_i, CF_j) \in M_{CF}, h \neq l_j} w N_{CF_i} N_{CF_j} D(CF_i, CF_j) \\ &+ \sum_{(CF_i, CF_j) \in C_{CF}, h = l_j} \overline{w} N_{CF_i} N_{CF_j} (D_{max} - D(CF_i, CF_j)) \end{aligned} \quad (7)$$

We can see that the optimal assignment of each CF entry also depends on the current assignment of the other CF entries due to the violation cost of pairwise constraints in the second and third terms of Equation 7. Therefore, after all entries are re-assigned, they are randomly re-ordered, and the re-assignment process is repeated until no CF entry changes its cluster label between two successive iterations.

In the re-estimation step, given the cluster labels $\{l_i^{(t+1)}\}_{i=1}^m$ of all CF entries, the cluster centers $\{\mu_h\}_{h=1}^K$ are re-calculated in order to minimize the objective function of the current assignment. For simple calculation, each cluster center is also represented in the form of a CF-vector. By using the squared Euclidean measure, the CF-vector of each cluster prototype μ_h is calculated based on CF entries which are assigned to this cluster as follows:

$$N_{\mu_h} = \sum_{l_i=h} N_{CF_i} \tag{8}$$

$$\overrightarrow{LS}_{\mu_h} = \sum_{l_i=h} \overrightarrow{LS}_{CF_i} \tag{9}$$

$$SS_{\mu_h} = \sum_{l_i=h} SS_{CF_i} \tag{10}$$

We can see that in each re-assignment step, each entry CF_i moves to a new cluster μ_h if its contribution to the objective function is decreased with this re-assignment. Therefore, the objective function J_{obj} is decreased or unchanged after the re-assignment step. And in each re-estimation step, the mean of the CF-vector of each cluster μ_h corresponds to the mean of the CF entries (and therefore the points) in this cluster, that minimizes the contribution of μ_h to the component $\sum_{CF_i \in S_{CF}} D(CF_i, \mu_{l_i})$ of J_{obj}. The penalty terms of J_{obj} are not functions of the centroid, thus they do not take part in cluster center re-estimation. Therefore, the objective function J_{obj} will decrease or remain the same in the re-estimation step. Since J_{obj} is bounded below and decreases after each re-assignment and re-estimation steps, the proposed semi-supervised clustering will converge to a (at least local) minimum in each interactive iteration.

After each interactive iteration, new constraints are given to the system. These new constraints might be in contradiction with some of the ones previously deduced by the system from the earlier user interactive iterations. For this reason and also for computational time matters, our system omits at each step some of the constraints deduced at earlier steps. Therefore, the objective function J_{obj} may be different between different interactive iterations. And the convergence of the interactive semi-supervised model is thus not guaranteed. But we can verify the convergence of the model, practically, by determining, at the end of all interactive iterations, the global objective function which considers all feedback given by the user in all interactive iterations and then by verifying if this global objective function has improved or not after different interactive steps. This is a part of our current work.

3.3. Interactive interface

In order to allow the user to view the clustering results and to provide feedback to the systems, we implement an interactive interface as shown in Figure 1.

The rectangle at the bottom right corner of Figure 1 is the principal plane representing all presented clusters by their prototype images. In our system, the maximum number of cluster prototypes presented to the user on the principal plane is fixed at 30. The prototype image of each cluster is the most representative image of that cluster chosen as follows. In our model, we use the internal measure Silhouette-Width (SW) (Rousseeuw, 1987) to estimate the quality of each image in a cluster. The higher the SW value of an image in a cluster, the more representative this image is for the cluster. The prototype image of a cluster is thus the image with the highest SW value in the cluster. Any other internal measure could be used instead. The position of the prototype image of each cluster in the principal plane represents the position of the corresponding cluster center. It means that, if two cluster centers are close (or distant) in the n-dimensional feature space, their prototype images are close (or distant) in the 2D principal plane. For representing the cluster centers which are n-dimensional vectors in 2D plane, we use Principal Component Analysis (PCA) (Pearson, 1901); the principal plane consists of the two principal axes associated with the highest eigenvalues. The importance of an axis is represented by its inertia (the sum of the squared elements of this axis (Abdi and Williams, 2010)) or by the percentage of its inertia in the total inertia of all axes. In general, if the two principal axes explain (cumulatively) greater or equal to 80% of the total inertia, the PCA approach could lead to a nice 2D-representation of the prototype images. In our case, the accumulated inertia explained by the two first principal axes is about 65% for the Wang and PascalVoc2006 databases and about 20% for the Caltech101 and Corel30k image databases. As only a maximum of 30 clusters (and therefore 30 prototype images) can be shown to the user in an interactive iteration, a not very nice 2D-representation of prototype images does not influence on the results as long as the user can distinguish between the prototype images and have a rough idea of the distances between the clusters. When there are some prototype images which overlap each other, a slight modification of the PCA components can help to separate these images.

By clicking on a prototype image in the principal plane, the user can view the corresponding cluster. In Figure 1, each cluster selected by the user is represented by a circle:

- The prototype image of this cluster is located at the center of the circle.
- The 10 most representative images (images with the highest SW values), which have not received feedback from the user in the previous

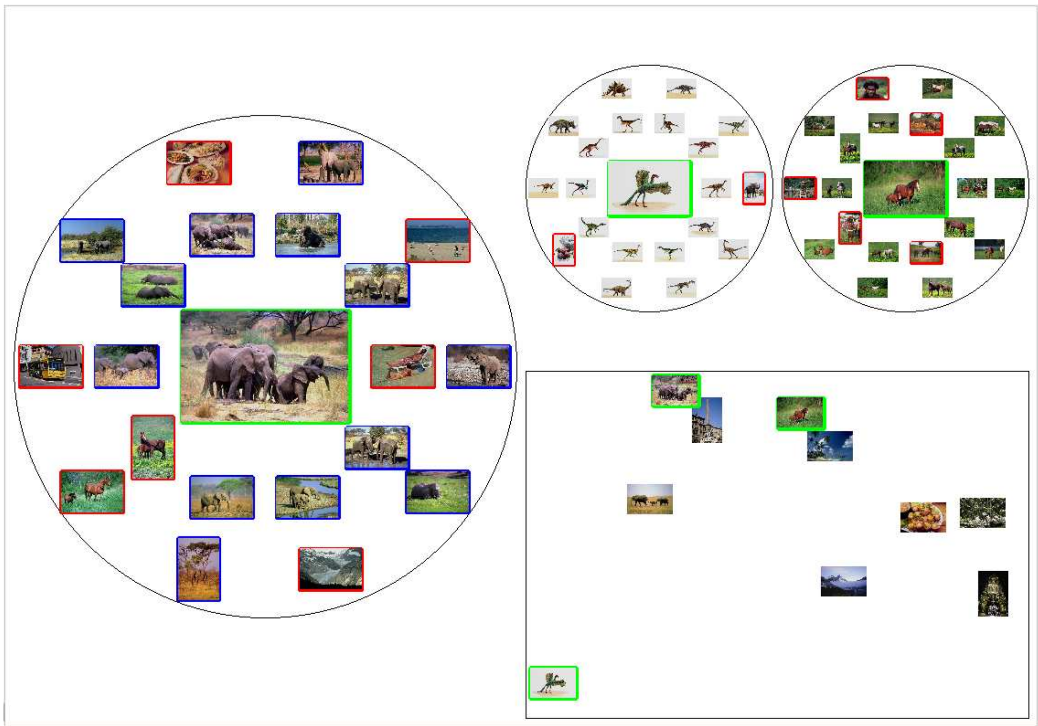


Figure 1: 2D interactive interface. The rectangle at the bottom right corner represents the principal plane consisting of the two first principal axes (obtained by PCA) of the prototype images of all clusters. Each circle represents the details of a particular cluster selected by the user.

iterations, are located in the first circle of images around the prototype image, near the center.

- The 10 least representative images (images with the smallest SW values), which have not received feedback from the user in the previous iterations, are located in the second circle of images around the prototype image, close to the cluster border.

By showing, for each iteration, the images which have not received user feedback in previous iterations, we wish to obtain feedback for different images.

The user can specify positive feedback and negative feedback (images in Figure 1 with blue and red borders respectively) for each cluster. The user can also change the cluster assignment of a given image by dragging and dropping the image from the original cluster to the new cluster. When an image is changed from cluster A to cluster B, it is considered as negative feedback for cluster A and positive feedback for cluster B. Therefore, after each interactive iteration, the process returns a positive image list and a negative image list for each cluster with which the user has interacted.

3.4. Pairwise constraint deduction

In each interactive iteration, user feedback is in the form of positive and negative images, while the supervised input information of the proposed semi-supervised clustering method are pairwise constraints between CF entries. Therefore, we have to deduce the pairwise constraints between CF entries from the user feedback.

At each interactive iteration and for each interacted cluster, all positive images should be in this cluster while negative images should move to another cluster. We consider that each image in the positive set is linked to each image in the negative set by a cannot-link, while all images in the positive set are linked by must-links. If we assume that all feedback is coherent between different interactive iterations, we try to group images, which should be in the same cluster according to the user feedback of all interactive iterations, in a group called *neighborhood*. We define:

- $Np = \{Np_i\}$ is the neighborhood list, each neighborhood $Np_i = \{x_j\}$ including a list of images which should be in a same cluster.
- $CannotNp = \{cannotNp_i\}$, each element $cannotNp_i = \{n_j\}$ including labels of the neighborhoods which should not be in the same cluster as Np_i. Two neighborhoods Np_i and Np_j are called cannot-link neighborhoods if there is at least one cannot-link between a point of Np_i and a point of Np_j.

After receiving the list of feedback in the current iteration, the lists Np and $CannotNp$ are updated as follows:

1. Update based on positive feedback: For each cluster μ_h which receives interaction from the user:
   (a) Initialize $n_h \leftarrow -1$, n_h indicates the neighborhood including positive images of the cluster μ_h.
   (b) If all positive images of μ_h are not included in any neighborhood $\rightarrow$ create a new neighborhood for these positive images and assign n_h as the index of this neighborhood.

(c) If some positive images of μ_h are already included in one or multiple neighborhoods $\rightarrow$ merge these neighborhoods (in the case of multiple neighborhoods) into one single neighborhood, insert the other positive images which are not included in any neighborhood to this neighborhood and update n_h as the index of this neighborhood. Also update the set *CannotNp* to signify that neighborhoods that had cannot-link with one of the neighborhoods which has merged, now have cannot-link with the new neighborhood.

2. Update based on negative feedback: For each negative image x_j of each cluster μ_h which receives interaction from the user:

   (a) If x_j is not included in any neighborhood $\rightarrow$ create a new neighborhood for x_j.

   (b) If x_j is already included in the neighborhood Np_{n_j}, and Np_{n_h} is the neighborhood corresponding to the positive images of the cluster μ_h, update the corresponding $cannotNp_{n_j}$ and $cannotNp_{n_h}$ to signify that Np_{n_j} and Np_{n_h} have cannot-link.

As we assume that the user feedback is coherent among different interactive iterations, all images in a same neighborhood should be in a same cluster and images of cannot-link neighborhoods should be in different clusters. There may be cannot-link images belonging to the same CF_i. There may also be simultaneous must-link and cannot-link between images of CF_i and images of CF_j. In such cases, these CF entries should be split into purer CF entries. To do so, we define a *seed* of an entry CF_i as a subset of images of CF_i so that the images of this *seed* are included in a same neighborhood.

Therefore, an entry CF_i may contain some *seeds* corresponding to different neighborhoods and other images which are not included in any other neighborhood. Cannot-link may or may not exist between seeds of a CF entry. With each CF entry that should be split, we present the user with each pair of seeds, which do not have cannot-link between them, to demand more information (for each seed, the image which is closest to the center of the seed is presented):

- If the user indicates that there is must-link between these two seeds, these seeds and also their corresponding neighborhoods are merged.
- If the user indicates that there is cannot-link between these two seeds, update the corresponding $cannotCF$ lists specifying that their two corresponding neighborhoods have cannot-link between them.

An entry CF_i is split as follows: if CF_i has p seeds, it should be split into p different CF entries; each new CF entry contains all points of a seed; every other point of CF_i which is not included in any seed is assigned to the CF entry corresponding to the closest seed. By splitting the necessary CF entries into purer CF entries, we can eliminate the case where cannot-link exists between images of a same CF or where must-link and cannot-link exist simultaneously between images of two different CF entries. Subsequently, pairwise constraints between CF entries can be deduced based on pairwise constraints between images as follows: if there is must-link (or respectively cannot-link) between two images of two CF entries, a must-link (or respectively cannot-link) is created between these two CF entries.

Concerning pairwise constraints between images, a simple and complete way to deduce them is to create must-link between each pair of images of a same neighborhood, and to create, for each pair of cannot-link neighborhoods (Np_i, Np_j), cannot-link between each image of Np_i and each image of Np_j. By deducing pairwise constraints between images in this way, the number of constraints between images can be very high, and therefore the number of constraints between CF entries could also be very high. The processing time of the semi-supervised clustering in the next phase could thus be very high due to the high number of constraints. There are different strategies for deducing pairwise constraints between images that could reduce the number of constraints and also the processing time. One of them is presented in Figure 2 and others are described and tested in Section 4. In Figure 2, must-links are created between positive images of each cluster while cannot-link are created between positive and negative images of each cluster (note the displacement feedback corresponding to a negative image of the source cluster and a positive image of the destination cluster).

4. Experiments

In this section, we present some experimental results of our interactive semi-supervised clustering model. We also, experimentally, compare our semi-supervised clustering model with the semi-supervised HMRF-kmeans. When using the semi-supervised HMRF-kmeans in the re-clustering phase, the initial unsupervised clustering is k-means.

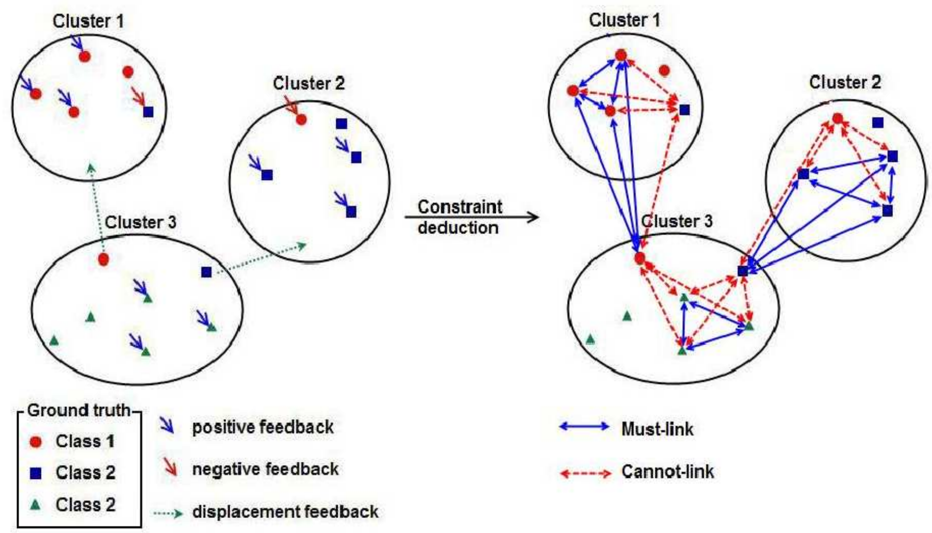


Figure 2: Example of pairwise constraint deduction between images from the user feedback.

4.1. Experimental protocol

In order to analyze the performance of our interactive semi-supervised clustering model, we use different image databases (Wang[1] (1000 images divided into 10 classes), PascalVoc2006[2] (5304 images divided into 10 classes), Caltech101[3] (9143 images divided into 101 classes)). Note that in our experiments we use the same number of clusters as the number of classes in the ground truth. As presented in Section 3.3, the cluster prototype images are shown to the user on the principal plane; users can choose to view and interact with any cluster in which they are interested. For databases which

[1] `http://wang.ist.psu.edu/docs/related/`

[2] `http://pascallin.ecs.soton.ac.uk/challenges/VOC/`

[3] `http://www.vision.caltech.edu/ImageDatasets/Caltech101/`

have a small number of classes, such as Wang and PascalVoc2006, all prototype images can be shown on the principal plane. For databases which have a large number of classes, such as Caltech101, only a part of the prototype images can be shown for visualization. In our system, the maximum number of cluster prototypes shown to the user in each iteration is fixed at 30. We use two simple strategies for choosing clusters to be shown for each iteration: 30 clusters chosen randomly or iteratively chosen pairs of closest clusters until there are 30 clusters.

The external measures compare the clustering results with the ground truth, thus they are compatible for estimating the quality of the interactive clustering involving user interaction. As different external measures analyze the clustering results in a similar way (see Lai et al. (2012a)), we use, in this paper, the external measure V-measure (Rosenberg and Hirschberg, 2007). The greater the V-measure values are, the better the results (compared to the ground-truth).

Concerning feature descriptors, we implement the local descriptor rgSIFT (van de Sande et al., 2008), an extension for color image of the SIFT descriptor (Lowe, 2004), that today is widely used for its high performance. The SIFT descriptor detects interest points from an image and describes the local neighborhood around each interest point by a 128-dimensional histogram of local gradient directions of image intensities. The rgSIFT descriptor of each interest point is computed as the concatenation of the SIFT descriptors calculated for the r and g components of the normalized RGB color space (van de Sande et al., 2008) and the SIFT descriptor in the intensity channel, resulting in a 3×128-dimensional vector. The "Bag of words" (Sivic and Zisserman,

2003) approach is chosen to group local features of each image into a single vector. It consists in two steps. Firstly, K-means clustering is used to group local features of all images in the database according to a number $dictSize$ of clusters. We then generate a dictionary containing $dictSize$ visual words which are the centroids of these clusters. The feature vector of each image is a $dictSize$ dimension histogram representing the frequency of occurrence of the visual words in the dictionary, by replacing each local descriptor of the image by the nearest visual word. Our experiments in (Lai et al., 2012a) show that local descriptors are better than global descriptors regarding the external measures and the value $dictSize = 200$ is a good trade-off between the size of the feature vector and the performance. Therefore, in our experiments, we use the rgSIFT descriptor together with a visual word dictionary of size 200.

In order to undertake the interactive tests automatically, we implement a software agent, later referred to as "user agent" that simulates the behavior of the human user when interacting with the system (assuming that the agent knows all the ground truth containing the class label for each image). At each interactive iteration, clustering results are returned to the user agent by the system; the agent simulates the behavior of the user giving feedback to the system. For simulating the user behavior, we suggest some rules:

- At each interactive iteration, the user agent interacts with a fixed number of c clusters.
- The user agent uses two strategies for choosing clusters: randomly chosen c clusters, or iteratively chosen pairs of closest clusters until there are c clusters.

- The user agent determines the image class (in the ground truth) corresponding to each cluster by the most represented class among the 21 presented images of the cluster. The number of images of this class in the cluster must be greater than a threshold $MinImages$. If this is not the case, this cluster can be considered as a noise cluster. In our experiments, $MinImages = 5$ for databases having a small number of classes (Wang, PascalVoc2006), and $MinImages = 2$ for databases having a large number of classes (Caltech101).

- When several clusters (among chosen clusters) correspond to a same class, the cluster in which the images of this class are the most numerous (among the 21 shown images of the cluster) is chosen as the principal cluster of this class. The classes of the other clusters are redefined as usual, but neutralize the images from this class.

- In each chosen cluster, all images, where the result of the algorithm corresponds to the ground truth, are labeled as positive samples of this cluster, while the others are negative samples of this cluster. All negative samples are moved to the cluster (among chosen clusters) corresponding to their class in the ground truth.

As presented in Section 3.4, we have to deduce pairwise constraints between images based on user feedback in each iteration and also on the neighborhood information. User feedback is in the form of positive and negative images of each cluster (the image which is displaced from one cluster to another cluster is considered as a negative image of the source cluster and a positive image of the destination cluster). The neighborhood information is in the form of the lists $Np = \{Np_i\}$ and $CannotNp = \{cannotNp_i\}$, where each neighborhood Np_i contains images which should be in a same cluster and $cannotNp_i$ identifies the list of neighborhoods having cannot-link with Np_i. Neighborhood information is deduced from user feedback during all interactive iterations, as presented in Section 3.4. Pairwise constraints between images will be used directly for the semi-supervised HMRF-Kmeans, while they have to be deduced into pairwise constraints between CF entries (see Section 3.4) to be used by our proposed semi-supervised clustering. We divide pairwise constraints between images into two kinds: *user constraints* and *deduced constraints*. *User constraints* are created directly, based on user feedback in each iteration, while *deduced constraints* are created by deduction rules. For instance, in the first iteration, the user marks x_1, x_2 as positive images and x_3 as a negative image of cluster μ_i; while in the second iteration, he marks x_1 and x_4 as positive images of cluster μ_j. The created user constraints are: must-link between positive images in the first iteration (x_1, x_2), must-link between positive images in the second iteration (x_1, x_4), and cannot-links between positive and negative images in the first iteration (x_1, x_3), (x_2, x_3). As there are must-links (x_1, x_2), (x_1, x_4), there is also a deduced must-link (x_2, x_4). In addition deduced cannot-link (x_3, x_4) is created, based on the must-link (x_1, x_4) and the cannot-link (x_1, x_3). We can see that deduced constraints can be created based on neighborhood information. In our experiments, we use different strategies for deducing pairwise constraints between images. These strategies are detailed in Table 1.

Table 1: Different strategies for deducing pairwise constraints between images based on user feedback and on neighborhood information.

N°	Take into account	Details
1	• ***All*** user constraints of ***all*** interactive iterations. • ***All*** deduced constraints of ***all*** interactive iterations.	All constraints are created based on the neighborhood information: • Must-link between each pair of images of each neighborhood. • Cannot-link between each image of each neighborhood $Np_i \in Np$ and each image of each neighborhood having cannot-link with Np_i (listed in $cannotNp_i$).
2	• ***All*** user constraints of ***all*** interactive iterations. • ***None*** of deduced constraints.	In each iteration, all possible user constraints are created: • Must-link between each pair of positive images of each cluster. • Cannot-link between each pair of a positive image and a negative image of a same cluster.
3	• ***All*** user constraints of ***all*** interactive iterations. • ***All*** deduced constraints in the ***current*** iteration (deduced constraints in the previous iterations are eliminated).	• In each iteration, all possible user constraints are created as in Strategy 2. • Deduced constraints in the current iteration are created while updating the neighborhoods as follows: – If there is a must-link (or cannot-link) (x_i, x_j), $x_j \in Np_m$, deduced must-links (or cannot-links) (x_i, x_l), $\forall x_l \in Np_m$ are created. – If there is a must-link (or cannot-link) (x_i, x_j), $x_i \in Np_m$, $x_j \in Np_n$, deduced must-links (or cannot-links) (x_k, x_l), $\forall x_k \in Np_m$, $\forall x_l \in Np_n$ are created.

Continued on next page

Table 1 – continued from previous page

N°	Take into account	Details
4	• User constraints between images and cluster centers of ***all*** interactive iterations. • Deduced constraints between images and cluster centers in the ***current*** iteration (deduced constraints in the previous iterations are eliminated).	In each iteration, the positive image having the best internal measure (SW) value among all positive images of each cluster is the center of this cluster. • Must-link/cannot-link user constraints are created in each iteration between each positive/negative image and the corresponding cluster center. • Deduced constraints in the current iteration are created while updating the neighborhoods as follows: – If x_i and x_j must be in the same (or different) clusters (based on user feedback), $x_j \in Np_m$, deduced must-links (or cannot-links) are created between x_i and each center image of Np_m. – If x_i and x_j must be in the same (or different) clusters (based on user feedback), $x_i \in Np_m$, $x_j \in Np_n$, deduced must-links (or cannot-links) are created between x_i and each center image of Np_n and between x_j and each center image of Np_m.
5	• User constraints (must-links between the most distant images and cannot-links between the closest images) of ***all*** iterations. • Deduced constraints (must-links between the most distant images and cannot-links between the closest images) of ***all*** iterations.	• User constraints are created for each cluster in each iteration as follows: must-links are successively created between two positive images (at least one of them is not selected by any must-link) that have the longest distance until all positive images of the cluster are connected by these must-links; cannot-links are created between each negative image and the nearest positive image of the cluster. • Deduced constraints are created in each iteration as follows: must-links for each neighborhood are successively created between two images that have the longest distance until all images of this neighborhood are connected by these must-links; cannot-links are deduced, for each pair of cannot-link neighborhoods (Np_i,Np_j), between each image of Np_i and the nearest image of Np_j and between each image of Np_j and the nearest image of Np_i.
6	Same idea as in strategy 5, but the size of the neighborhoods is considered while creating deduced cannot-links.	User constraints and deduced must-link constraints are created as in Strategy 5. For each pair of cannot-link neighborhoods, deduced cannot-links are only created between each image of the neighborhood that has the least number of images and the nearest image of the neighborhood that has the most images.

4.2. Experimental results

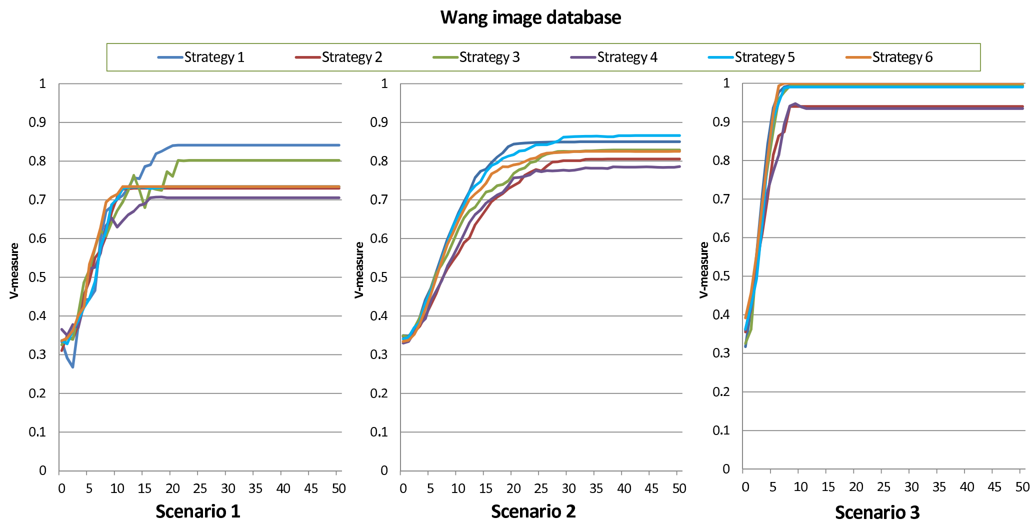
4.2.1. Analysis of different strategies for deducing pairwise constraints between images

The first set of experiments aims at evaluating the performance of our interactive semi-supervised clustering model using different strategies for deducing pairwise constraints between images. Note that constraints between CF entries should be deduced from constraints between images, before being used in the re-clustering phase. We use the Wang and the PascalVoc2006 image databases for these experiments. For these two databases, we propose three test scenarios (note that c specifies the number of clusters which are chosen for interacting in each iteration):

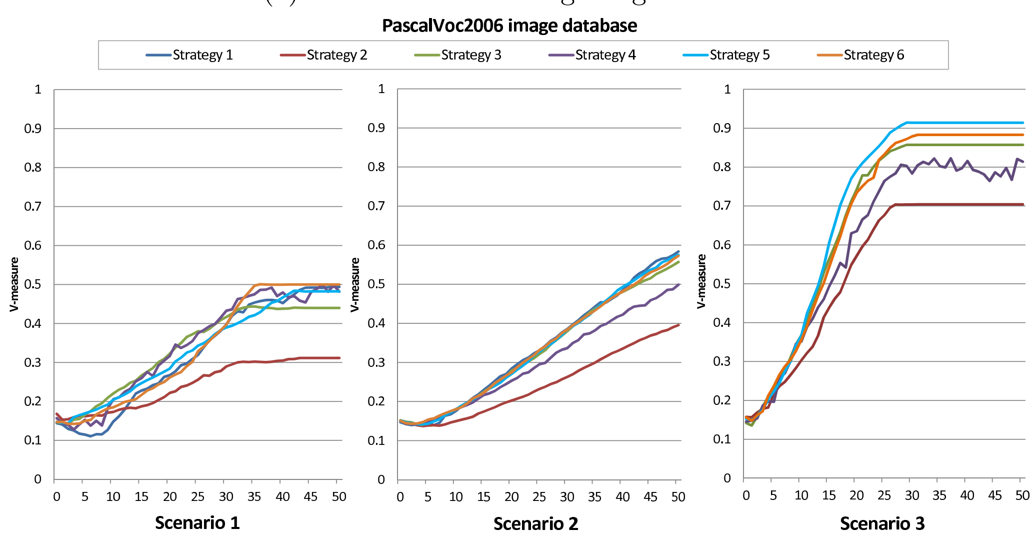
- Scenario 1: $c = 5$ closest clusters are chosen.
- Scenario 2: $c = 5$ clusters are randomly chosen.
- Scenario 3: $c = 10$, all cluster are chosen (Wang and PascalVoc2006 both have 10 clusters).

Note that our experiments are carried out automatically, *i.e.* the feedback is given by a software agent simulating the behaviors of the human user when interacting with the system. In fact, the human user can give feedback by clicking for specifying the positive and/or negative images of each cluster or by dragging and dropping the image from a cluster to another cluster. For each cluster selected by the user, only 21 images of this cluster are displayed (see Figure 1). Therefore, for interacting with 5 clusters (scenarios 1, 2) or 10 clusters (scenario 3), the user has to realize respectively a maximum of 105 or 210 mouse clicks in each interactive iteration. These upper bounds do not depend on neither the size of the database nor the pairwise constraint deduction strategy, and in practice the number of clicks that the user has to provide is far lower. However, the number of deduced constraints may be much greater than the user's clicks (and this number depends on the database size and on the pairwise constraint deduction strategy). When applying the interactive semi-supervised clustering model in the indexing phase, the user is generally required to provide as much feedback as possible for having a good indexing structure which could lead to better results in the further retrieval phase. Therefore, in the case of the indexing phase, the proposed number of clicks seems tractable.

Figures 3a and 3b show, respectively, the results during 50 interactive iterations of our proposed interactive semi-supervised clustering model on the Wang and PascalVoc2006 image databases, with the three proposed scenarios. The results are shown according to 6 strategies for deducing pairwise constraints presented in Table 1. The vertical axis specifies the V-measure values, while the horizontal axis specifies the number of iterations. Note that with each selected cluster, the user agent gives all possible feedback. Therefore, for each scenario, the numbers of user feedback are equivalent between different iterations and between different strategies. As in scenario 2, clusters are randomly chosen, we realize this scenario 10 times for each database. The curves of the scenario 2 shown in Figures 3a and 3b represent the mean values of the V-measure over these 10 executions at each iteration. The average standard deviation of each strategy after 50 iterations is presented in Table 2. The corresponding execution time for these experiments is presented in Table 3 (note that for the scenario 2, the average execution



(a) Results on the Wang image database



(b) Results on the PascalVoc2006 image database

Figure 3: Results of our proposed interactive semi-supervised clustering model during 50 interactive iterations on the Wang and PascalVoc2006 image databases, using 6 strategies for deducing pairwise constraints. The horizontal axis specifies the number of iterations.

Table 2: Average standard deviation of 10 executions of the scenario 2 after 50 interactive iterations corresponding to the experiments of our proposed interactive semi-supervised clustering model shown in Figures 3a and 3b

	Average standard deviation	
	Wang database	PascalVoc2006 database
Strategy 1	0.033	0.022
Strategy 2	0.044	0.017
Strategy 3	0.045	0.025
Strategy 4	0.047	0.022
Strategy 5	0.036	0.024
Strategy 6	0.044	0.026

times of 10 executions are shown). The experiments are executed using a normal PC with 2GB of RAM.

We can see that the clustering results progress, in general, after each interactive iteration, in which the system re-clusters the dataset by considering the constraints deduced from accumulated user feedback. In most cases, the clustering results converge after only a few iterations. This may be due to the fact that no new knowledge is provided. Moreover, we can easily see that the clustering results are better and converge more quickly when the number of chosen clusters (and therefore the number of constraints) in each interactive iteration is higher (scenario 3 gives better results and converges more quickly than scenarios 1 and 2). In addition, for both image databases, scenario 2, in which clusters are randomly chosen for interacting, gives better results than scenario 1, in which the closest clusters are chosen. When selecting the

Table 3: Processing time after 50 interactive iterations of the experiments of our proposed interactive semi-supervised clustering model shown in Figures 3a and 3b.

	Wang database		
	Scenario 1	Scenario 2	Scenario 3
Strategy 1	1h58'	2h24'	1h41'
Strategy 2	9'	12'	10'
Strategy 3	31'	19'	47'
Strategy 4	8'	9'	8'
Strategy 5	8'	9'	9'
Strategy 6	6'	8'	8'
	PascalVoc2006 database		
	Scenario 1	Scenario 2	Scenario 3
Strategy 1	16d12h	14d11h	
Strategy 2	2h55'	4h02'	5h6'
Strategy 3	3h23'	6h39'	6h22'
Strategy 4	1h9'	1h33'	2h17'
Strategy 5	3h33'	4h42'	3h10'
Strategy 6	1h3'	1h21'	2h

closest clusters there may be only several clusters that always receive user feedback; thus the constraint information is less than when all the clusters could receive user feedback when we randomly select the clusters.

As regards different strategies for deducing pairwise constraints, we can see that for each database, the average standard deviations over 10 executions of the scenario 2 are similar for all scenarios. Therefore, we can compare different strategies based on the mean values shown on Figures 3a and 3b. We can see that:

- Strategy 1 shows, in general, very good performance but the processing time is huge because it uses all possible user constraints and deduced constraints created during all iterations.

- Strategy 2, the only strategy uniquely using user constraints, generally gives the worst results; thus deduced constraints are needed for better performance. Its processing time is also high due to the large number of user constraints.

- Strategy 3 shows good or very good performance but some oscillations exist between different iterations because, when overlooking previously deduced constraints, some important constraints may be omitted. Its processing time is high.

- Strategy 4 gives better results than strategy 2, but the results are unstable because this strategy also overlooks previously deduced constraints. It has good execution time while reducing the number of constraints.

- Strategy 5 generally gives good or very good results by keeping important constraints (must-links between the most distant images and

Table 4: Processing time after 50 interactive iterations corresponding to the experiments presented in Figures 4a and 4b of the proposed semi-supervised clustering and of the semi-supervised HMRF-kmeans. Strategy 6 in Table 1 for deducing pairwise constraints is used.

	Wang database		
	Scenario 1	Scenario 2	Scenario 3
Proposed semi-supervised clustering	6'	8'	8'
HMRF-kmeans	7'	11'	10'
	PascalVoc2006 database		
	Scenario 1	Scenario 2	Scenario 3
Proposed semi-supervised clustering	1h3'	1h21'	2h'
HMRF-kmeans	2h16'	3h10'	2h49'

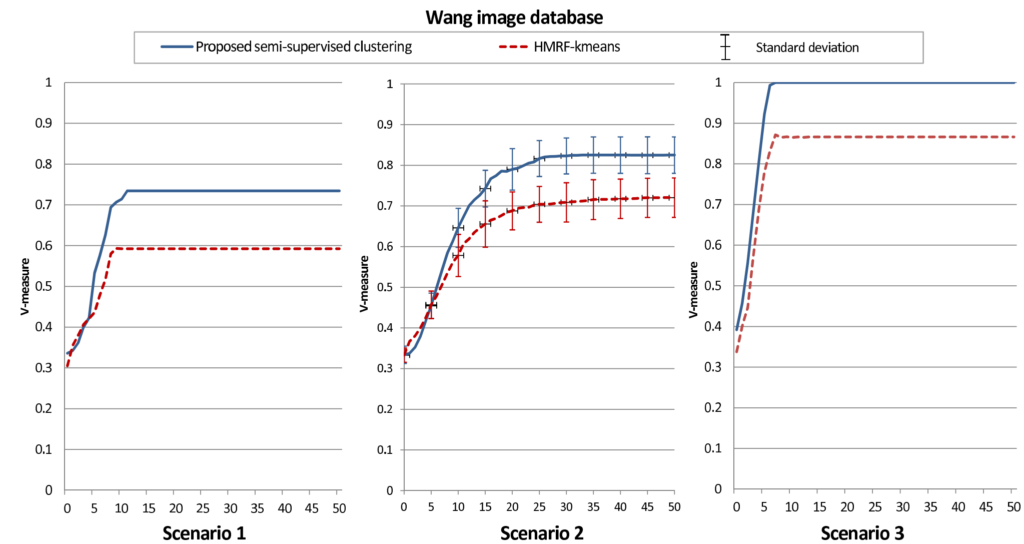
cannot-links between the closest images), but its processing time is still high.

- Strategy 6, by reducing the deduced cannot-link constraints from strategy 5, gives in general very good results in low execution time.

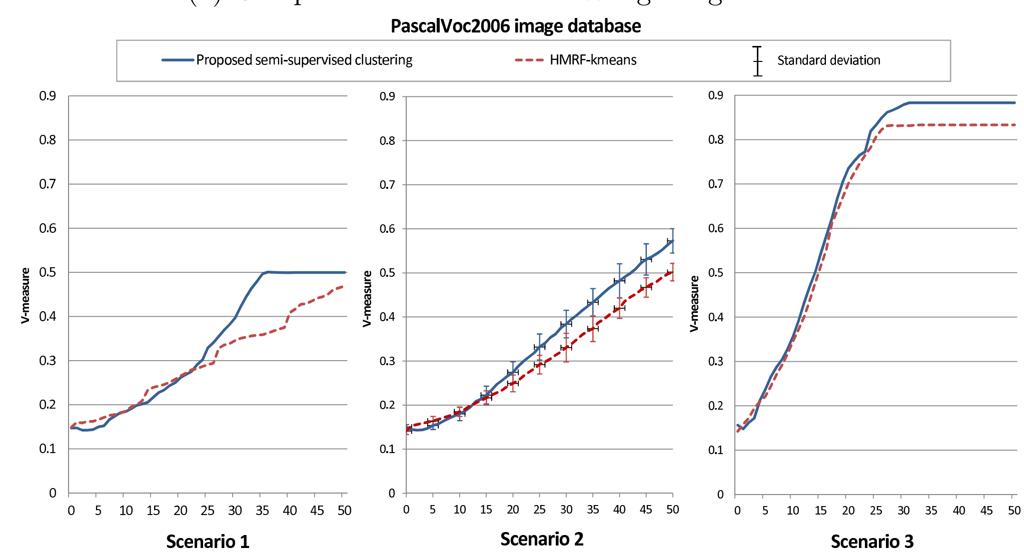
We can conclude, from this analysis, that strategy 6 shows the best trade-off between performance and processing time. This strategy will be used in further experiments.

4.2.2. Comparison of the proposed semi-supervised clustering model and the semi-supervised HMRF-kmeans

Figures 4a and 4b represent, respectively, the clustering results for 50 interactive iterations on the Wang and the PascalVoc2006 image databases



(a) Comparison results on the Wang image database



(b) Comparison results on the PascalVoc2006 image database

Figure 4: Comparison of the proposed semi-supervised clustering and the semi-supervised HMRF-kmeans with 50 interactive iterations using Strategy 6 in Table 1 for deducing the pairwise constraints between images. The horizontal axis represents the number of iterations.

when using our proposed semi-supervised clustering and the semi-supervised HMRF-kmeans in the re-clustering phase. The three scenarios described in Section 4.2.1 and strategy 6, for deducing pairwise constraints between images, are used. Note that the results of scenario 2 represent the mean values and also the standard deviations over 10 executions at each iteration. The corresponding processing time is presented in Table 4. We can see that in all scenarios, our proposed method gives better results, in lower processing time than the HMRF-kmeans. While the pairwise constraints between images are directly used by the HMRF-kmeans, they are deduced in pairwise constraints between CF entries for being used by our proposed semi-supervised clustering. A CF entry groups a list of similar images, thus many pairwise constraints between images can be represented by only one pairwise constraints between CF entries. Therefore, with a same set of user feedback, the number of pairwise constraints between images is generally greater than the number of the pairwise constraints between CF entries. Thus the processing time of the HMRF-kmeans is much higher than the processing time of our proposed method. Moreover, when a pairwise constraint (CF_i, CF_j) is deduced from the pairwise constraint of the corresponding images $(x_k, x_l), x_k \in CF_i, x_l \in CF_j$, the constraint (CF_i, CF_j) forces the grouping or separating of not only the two images x_i and x_j but also the other images included in CF_i and CF_j. And therefore, the clustering results given by our proposed method are better than the ones given by the HMRF-kmeans. Moreover, similar to the experiments presented in Section 4.2.1, the scenario 2 in which the clusters are randomly chosen for interacting gives better results than the scenario 1 in which the closest clusters are chosen. In the

Table 5: Processing time after 50 interactive iterations corresponding to the experiments on the Caltech101 image database in Figure 5 for the proposed semi-supervised clustering and for the semi-supervised HMRF-kmeans. Strategy 6 in Table 1 for deducing pairwise constraints is used.

	Proposed semi-supervised clustering	HMRF-kmeans
Scenario 4	13h26’	48h33’
Scenario 5	8h4’	33h45’
Scenario 6	33h34’	157h26’
Scenario 7	50h12’	101h11’

following experiments on the Caltech101 image database, we present only the clustering results when the clusters are randomly chosen.

As the Caltech101 database has a large number of classes (101 classes), we do not show all clusters to the user on the principal plane but only a small number of clusters (we fix the maximum number of cluster that could be shown on the principal plane to 30). There are two strategies for choosing clusters to be shown on the principal plane: either clusters are randomly chosen or the closest clusters are chosen. The user agent randomly chooses, among shown clusters, c clusters for interacting. We use 4 scenarios for the experiments on the Caltech101 image database:

- Scenario 4: the closest clusters are chosen to be shown to the user, c=5 clusters are chosen by the user agent for interacting.
- Scenario 5: clusters are randomly chosen to be shown to the user, c=5 clusters are chosen by the user agent for interacting.

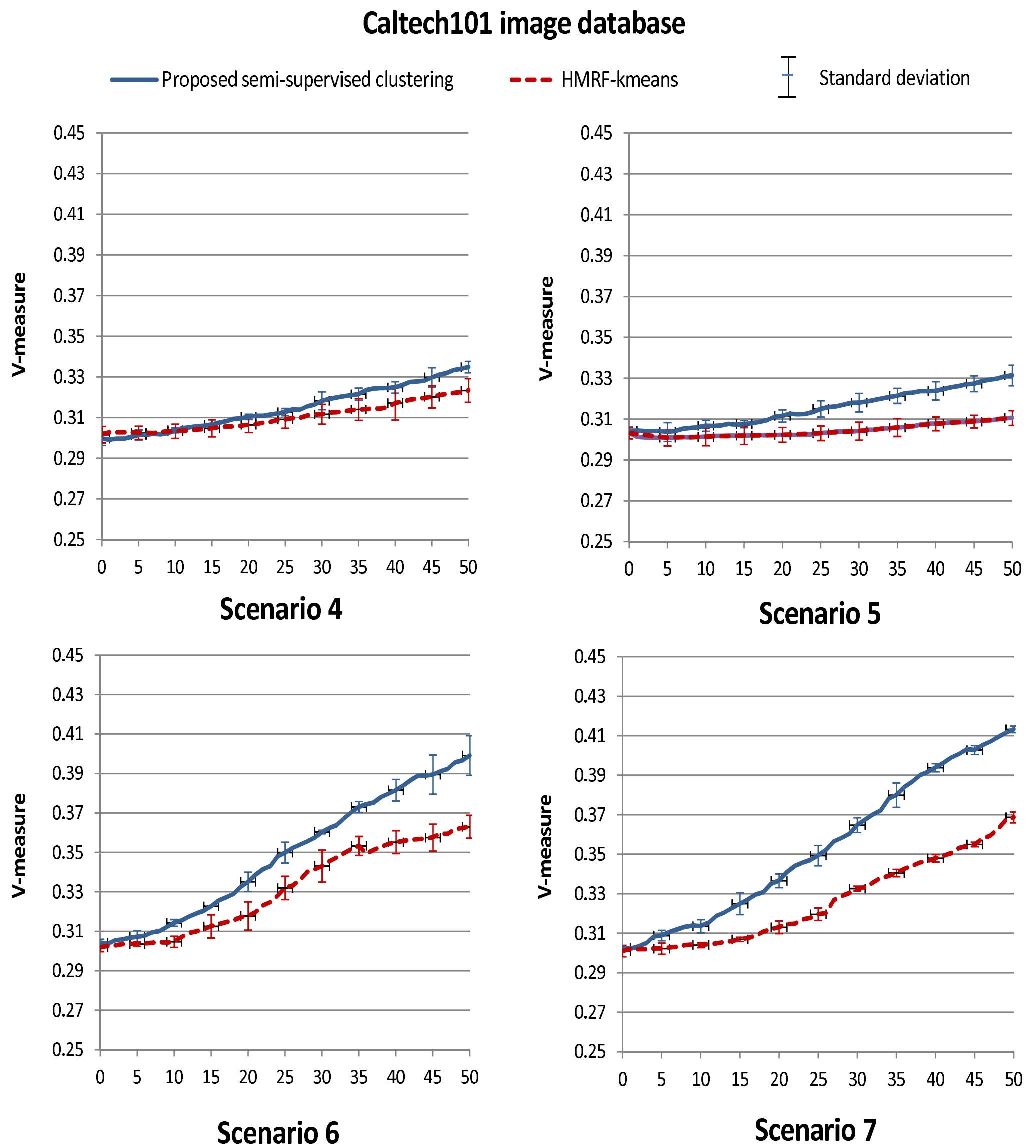


Figure 5: Comparison of the proposed semi-supervised clustering and the semi-supervised HMRF-kmeans on the Caltech101 image database for 50 interactive iterations. The strategy 6 in Table 1 for deducing pairwise constraints are used. The horizontal axis represents the number of iterations.

- Scenario 6: the closest clusters are chosen to be shown to the user, c=10 clusters are chosen by the user agent for interacting.
- Scenario 7: clusters are randomly chosen to be shown to the user, c=10 clusters are chosen by the user agent for interacting.

Figure 5 compares our proposed semi-supervised clustering and the HMRF-kmeans during 50 interactive iterations on the Caltech101 image database. The corresponding processing time is presented in Table 5. As in all these four scenarios, the clusters are randomly chosen for interacting, we realize each scenario 5 times and present in Figure 5 the mean values and also the standard deviations over 5 executions. The results shows that our proposed semi-supervised clustering outperforms the HMRF-kmeans in all four scenarios. Moreover, the clustering results are also better when the number of feedback for each iteration is high (scenarios 6 and 7 give better results than scenarios 4 and 5).

5. Conclusion

A new interactive semi-supervised clustering model for indexing image databases is presented in this article. After receiving user feedback for each interactive iteration, the proposed semi-supervised clustering re-organizes the dataset by considering the pairwise constraints between CF entries deduced from the user feedback. We present an interactive interface allowing the user to view, and to provide feedback. Experimental analysis, using a software user agent for simulating human user behavior, shows that our model improves the clustering results at each interactive iteration. Note that our experimental scenarios are realistic, they can be realized by a real user as the number of clicks required is tractable. The experiments on different image databases (Wang, PascalVoc2006, Caltech101), presented in this paper, also show that our semi-supervised clustering outperforms the semi-supervised HMRF-kmeans (Basu et al., 2004) in both performance and processing time.

Moreover, we propose and compare, experimentally, different strategies for deducing pairwise constraints from the user feedback accumulated from all interactive iterations. The experimental results show that strategy 6 in Table 1, which keeps only the most important constraints (must-links between the most distant images and cannot-links between the closest images), provides the best trade-off between the performance and the processing time. Strategy 6 is therefore the most suitable, in our context involving the user in the indexing phase by clustering.

Our future work aims to verify our proposed semi-supervised clustering model with larger image databases such as Corel30k, MIRFLICKR, to prove experimentally the convergence of our algorithm, and to look for different strategies for deducing the pairwise constraints or for representing the clustering results that could improve the performance of our model in the context of huge image databases.

References

Abdi, H., Williams, L. J., 2010. Principal component analysis.

Basu, S., Banerjee, A., Mooney, R. J., 2002. Semi-supervised clustering by seeding. In: Proceedings of the Nineteenth International Conference on

Machine Learning. ICML '02. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 27–34.

Basu, S., Bilenko, M., Mooney, R. J., 2004. A probabilistic framework for semi-supervised clustering. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. KDD '04. ACM, New York, NY, USA, pp. 59–68.

Bayer, R., McCreight, E. M., 1972. Organization and maintenance of large ordered indexes. Acta Informatica 1, 173–189.

Beckmann, N., Kriegel, H.-P., Schneider, R., Seeger, B., 1990. The r*-tree: an efficient and robust access method for points and rectangles. In: Proceedings of the 1990 ACM SIGMOD international conference on Management of data. SIGMOD '90. ACM, New York, NY, USA, pp. 322–331.

Bentley, J. L., Sep. 1975. Multidimensional binary search trees used for associative searching. Commun. ACM 18 (9), 509–517.

Berchtold, S., Keim, D. A., Kriegel, H.-P., 1996. The x-tree: An index structure for high-dimensional data. In: Proceedings of the 22th International Conference on Very Large Data Bases. VLDB '96. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 28–39.

Dubey, A., Bhattacharya, I., Godbole, S., 2010. A cluster-level semi-supervision model for interactive clustering. In: Proceedings of the 2010 European conference on Machine learning and knowledge discovery in databases: Part I. ECML PKDD'10. Springer-Verlag, Berlin, Heidelberg, pp. 409–424.

Guttman, A., 1984. R-trees: A dynamic index structure for spatial searching. In: International Conference on Management of Data. ACM, pp. 47–57.

Henrich, A., Six, H. W., Widmayer, P., 1989. The lsd tree: spatial access to multidimensional and non-point objects. In: Proceedings of the 15th international conference on Very large data bases. VLDB '89. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 45–53.

Jain, A. K., Murty, M. N., Flynn, P. J., 1999. Data clustering: a review. ACM Comput. Surv. 31 (3), 264–323.

Katayama, N., Satoh, S., 1997. The sr-tree: an index structure for high-dimensional nearest neighbor queries. In: Proceedings of the 1997 ACM SIGMOD international conference on Management of data. SIGMOD '97. ACM, New York, NY, USA, pp. 369–380.

Kulis, B., Basu, S., Dhillon, I., Mooney, R., 2005. Semi-supervised graph clustering: a kernel approach. In: In ICML 05: Proceedings of the 22nd international conference on Machine learning. ACM Press, pp. 457–464.

Lai, H. P., Visani, M., Boucher, A., Ogier, J.-M., 2012a. An experimental comparison of clustering methods for content-based indexing of large image databases. Pattern Analysis and Applications 15, 345–366.

Lai, H. P., Visani, M., Boucher, A., Ogier, J.-M., 2012b. Unsupervised and semi-supervised clustering for large image database indexing and retrieval. In: IEEE International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF). pp. 1–6.

Lance, G. N., Williams, W. T., 1967. A general theory of classificatory sorting strategies ii. clustering systems. The computer journal 10 (3), 271–277.

Likas, A., Vlassis, N., Verbeek, J. J., Feb. 2003. The global k-means clustering algorithm. Pattern Recognition 36 (2), 451–461.

Lowe, D. G., Nov. 2004. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vision 60 (2), 91–110.
URL http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94

Nievergelt, J., Hinterberger, H., Sevcik, K. C., 1988. Readings in database systems. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, Ch. The grid file: an adaptable, symmetric multikey file structure, pp. 582–598.

Pearson, K., 1901. On lines and planes of closest fit to systems of points in space. Philosophycal Magazine 6 (2), 559–572.

Robinson, J. T., 1981. The k-d-b-tree: a search structure for large multidimensional dynamic indexes. In: Proceedings of the 1981 ACM SIGMOD international conference on Management of data. SIGMOD '81. ACM, New York, NY, USA, pp. 10–18.

Rosenberg, A., Hirschberg, J., 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). pp. 410–420.

Rousseeuw, P., Nov. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. 20 (1), 53–65.

Sellis, T. K., Roussopoulos, N., Faloutsos, C., 1987. The r+-tree: A dynamic index for multi-dimensional objects. In: Proceedings of the 13th International Conference on Very Large Data Bases. VLDB '87. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 507–518.

Sivic, J., Zisserman, A., 2003. Video google: a text retrieval approach to object matching in videos. In: Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on. pp. 1470–1477.

van de Sande, K. E. A., Gevers, T., Snoek, C. G. M., 2008. Evaluation of color descriptors for object and scene recognition. In: IEEE Conference on Computer Vision and Pattern Recognition.

Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S., 2001. Constrained k-means clustering with background knowledge. In: Proceedings of the Eighteenth International Conference on Machine Learning. ICML '01. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 577–584.

White, D. A., Jain, R., 1996. Similarity indexing with the ss-tree. In: Proceedings of the Twelfth International Conference on Data Engineering. ICDE '96. IEEE Computer Society, Washington, DC, USA, pp. 516–523.

Xu, R., Wunsch, D., I., 2005. Survey of clustering algorithms. Neural Networks, IEEE Transactions on 16 (3), 645–678.

Zhang, T., Ramakrishnan, R., Livny, M., 1996. Birch: an efficient data clustering method for very large databases. In: Proceedings of the 1996 ACM SIGMOD international conference on Management of data. SIGMOD '96. ACM, New York, NY, USA, pp. 103–114.