Fuzzy Clustering with Pairwise Constraints for Knowledge-Driven Image Categorization

Nizar Grira, Michel Crucianu, Nozha Boujemaa
IMEDIA Research Group, INRIA Rocquencourt
Domaine de Voluceau, B.P. 105
78153 Le Chesnay Cedex, France
{Nizar.Grira, Michel.Crucianu, Nozha.Boujemaa}@inria.fr

To appear in IEE Proc. Vision, Image & Signal Processing.

Abstract

The identification of categories in image databases usually relies on clustering algorithms that only exploit the feature-based similarities between images. The addition of semantic information should help improving the results of the categorization process. *Pairwise constraints* between some images are easy to provide, even when the user has a very incomplete prior knowledge of the image categories he/she can expect to find in a database. A categorization approach relying on such semantic information is called *semi-supervised clustering*. We present here a new semi-supervised clustering algorithm, Pairwise-Constrained Competitive Agglomeration, based on a fuzzy cost function that takes pairwise constraints into account. Our evaluations show that with a rather low number of constraints this algorithm can significantly improve the categorization.

1 Introduction

Effective access to the content of an image collection requires a meaningful categorization of the images. To identify "natural" categories in a collection of images (or other data items), unsupervised clustering (or *cluster analysis*, see the surveys in [15], [16]) relies exclusively on measures of similarity

between the images. When compared to what a human would find natural on the same collection (note that a human may be unable to process very high-dimensional data and huge volumes of data), the results produced by clustering may be quite disappointing.

By letting the user provide some supervision to the system, one can expect to obtain more adequate results. Supervision may consist in class labels for a few data items (not necessarily from all the classes) or in pairwise constraints specifying whether two items should be in a same category or rather in different categories. Such pairwise constraints are indeed much easier to provide than class labels when the user has a very incomplete prior knowledge of the categories he/she can expect to find in the database. A categorization approach that takes into account such simple semantic information during the clustering process is called *semi-supervised clustering* and became a topic of significant interest (see [11], [20], [2], [6], [3]).

In the case of image collections, pairwise constraints can either be directly provided by users or obtained from the keyword annotations that are usually few and only available for some categories. As a simple example, *must-link* constraints can be defined between images that share many keywords and *cannot-link* constraints between annotated images that have no keyword in common. A deeper analysis of the semantic relations between keywords (synonymy, etc.) can also be performed when generating the constraints, but we do not address this issue here.

Existing semi-supervised clustering algorithms, such as Pairwise Constrained K-means (PCKmeans, [2]), rely on parameters that are difficult to set (such as the desired number of clusters) and require a high number of constraints to obtain significantly better results. The new semi-supervised clustering algorithm we put forward in the following, Pairwise Constrained Competitive Agglomeration (PCCA), brings improvements on both issues.

In the next section we review very briefly existing work on clustering and suggest a taxonomy of the semi-supervised clustering algorithms. In section 3 we remind the Competitive Agglomeration clustering algorithm (CA, [13]) and explain why we selected it as the starting point for our current work. PCCA, our new semi-supervised fuzzy clustering algorithm, is then described in section 4, where we also discuss the choice of its parameters. Section 5 presents an experimental evaluation of PCCA on a well-known benchmark and on an image categorization problem. In the conclusion we highlight the advantages of PCCA and we give directions for future work.

2 Related Research

The many existing clustering algorithms (see [15], [16]) can be grouped into two broad categories: partitional or hierarchical. The partitional algorithms aim at producing a partition of the data and are based on the optimization of specific objective functions. Since our main concern here is the categorization of a collection of images, we focus in the following on partitional algorithms.

Prototype-based partitional algorithms rely on the possibility to represent each cluster by a prototype and attempt to minimize a cost function that measures the dispersion of the clusters. In general, the prototypes are the cluster centroids, as in the popular k-means algorithm [19] or in its fuzzy evolution, Fuzzy C-Means (FCM, [5]). FCM has been constantly improved for more than twenty years by the use of the Mahalanobis distance [14], the definition of competitive agglomeration [13], [7] or the addition of a noise cluster [10], [18]. Due to their simplicity, computational efficiency (complexity of $O(CN)$, C being the number of prototypes and N the number of data items to cluster) and flexibility in using various metrics, prototype-based clustering algorithms are very popular. When compared to their crisp counterparts, fuzzy clustering algorithms are significantly more robust and can also model situations where clusters actually overlap.

In a probabilistic framework, mixture-resolving clustering algorithms assume that the data items in a cluster are drawn from one of several distributions and attempt to estimate the parameters of these distributions. A major step was the introduction of the expectation maximization (EM) algorithm in [12]. Recent developments concern the choice of the number of clusters, see [1] or [8]. Mixture-resolving methods usually have a higher computational complexity and make rather strong assumptions regarding the distribution of the data.

Unsupervised clustering algorithms don't take into account prior knowledge unless it can be expressed directly in the metric (or, for mixture-resolving methods, in the distributions considered), so quite often the resulting categories do not reflect user expectations. Consequently, semi-supervised clustering—letting "knowledge" provide a limited form of supervision—has recently become a topic of interest. More specifically, to help unsupervised clustering a small amount of knowledge can be used, concerning either class labels for some items (not necessarily from all the classes) or pairwise constraints between data items; the constraints specify whether two data items should be in the same cluster or not.

Unlike traditional clustering, the semi-supervised approach to clustering has a short history and few methods were published until now. Two sources of information are usually available to a semi-supervised clustering method: the similarity between data items and some must-link or cannot-link pairwise constraints. Semi-supervised clustering implicitly assumes that these two sources of information do not contradict each other completely. These two sources of information are combined either by modifying the search for appropriate clusters or by adapting the similarity measure (see [4]).

- In *search-based* methods, the clustering algorithm itself is modified so that user-provided constraints can be used to bias the search for an appropriate partition. This can be done in several ways, such as by performing a transitive closure of the constraints and using them to initialize clusters [2], by including in the cost function a penalty for lack of compliance with the specified constraints [11], or by requiring constraints to be satisfied during cluster assignment in the clustering process [20].

- In *similarity-adapting* methods, an existing clustering algorithm using a similarity measure is employed; however, the similarity measure is first adapted in order to satisfy the constraints in the supervised data. Several similarity measures have been used for similarity-adapting semi-supervised clustering: the Jensen-Shannon divergence trained by gradient descent [9], the Euclidean distance modified by a shortest-path algorithm [17] or Mahalanobis distances adjusted by convex optimization [21]. Among the clustering algorithms using such adapted similarity measures we can mention hierarchical single-link [6] or complete-link [17] clustering and k-means [21], [6].

Similarity-adapting methods can potentially be applied to a wider range of situations, but they either need a significantly higher amount of supervision or rely on specific strong assumptions regarding the relation between the initial and the target similarity measures.

3 Competitive Agglomeration: a Short Reminder

Most early partitional algorithms assumed that the number of clusters was known prior to clustering; since this is rarely the case, techniques for finding an "appropriate" number of clusters had to be devised. For methods based on the minimization of a cost function, the problem is partly solved by including a *regularization* term in the cost function. This way, instead of having to specify arbitrarily a value for the desired number of clusters—with a strong impact on the outcome of the clustering—one must set a regularization parameter for which a relatively wide range of values allows to obtain good clustering results.

In the Competitive Agglomeration (CA) fuzzy partitional algorithm introduced in [13], regularization makes clusters compete for membership of data items and the number of clusters is progressively reduced until a minimum of the full cost function is reached. Let $\mathbf{x}_i$, $i \in \{1, .., N\}$ be the vectors representing the N data items to be clustered, $\mathbf{V}$ the matrix having as columns the prototypes μ_k, $k \in \{1, .., C\}$ of C clusters ($C \ll N$) and $\mathbf{U}$ the matrix of the membership degrees, with u_{ik} being the membership of $\mathbf{x}_i$ to the cluster k. Let $d(x_i, \mu_k)$ be the distance between the vector $\mathbf{x}_i$ and the cluster prototype μ_k. The cost function CA attempts to minimize is (see [13]):

$$\begin{aligned} \mathcal{J}(\mathbf{V}, \mathbf{U}) &= \sum_{k=1}^{C} \sum_{i=1}^{N} u_{ik}^2 \, d^2(\mathbf{x}_i, \mu_k) \\ &- \beta(t) \sum_{k=1}^{C} \left(\sum_{i=1}^{N} u_{ik} \right)^2 \end{aligned} \tag{1}$$

under the constraint

$$\sum_{k=1}^{C} u_{ik} = 1, \text{ for } i \in \{1, \ldots, N\} \tag{2}$$

The first term in (1) is the standard Fuzzy C-Means (FCM, [5]) cost function. The second term defines the competition that progressively reduces the number of clusters. The $\beta(t)$ factor sets a balance between the terms and its dependence on t (iteration number) will be explained later.

We selected CA as the basis for our semi-supervised clustering algorithm (presented in the following) because CA has a low computational complexity,

shows good robustness and does not require a prior specification of the desired number of clusters.

4 Pairwise Constrained Competitive Agglomeration

4.1 Principle of the Method

The cost function to be minimized by our semi-supervised clustering algorithm must take into account both the feature-based similarity between data points and knowledge of the pairwise constraints. Let $\mathcal{M}$ be the set of must-link constraints, $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}$ implying that $\mathbf{x}_i$ and $\mathbf{x}_j$ should be assigned to the same cluster, and $\mathcal{C}$ the set of cannot-link constraints, $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}$ implying that $\mathbf{x}_i$ and $\mathbf{x}_j$ should be assigned to different clusters. With the notations defined for CA, the objective function PCCA minimizes is:

$$\begin{aligned} \mathcal{J}(\mathbf{V}, \mathbf{U}) &= \sum_{k=1}^{C} \sum_{i=1}^{N} u_{ik}^2 \, d^2(\mathbf{x}_i, \mu_k) \\ &+ \alpha \Big(\sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}} \sum_{k=1}^{C} \sum_{l=1, l \neq k}^{C} u_{ik} u_{jl} \\ &+ \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}} \sum_{k=1}^{C} u_{ik} \, u_{jk} \Big) \\ &- \beta \sum_{k=1}^{C} \left(\sum_{i=1}^{N} u_{ik} \right)^2 \end{aligned} \tag{3}$$

with the same constraint (2).

The prototype of a cluster k ($k \in \{1, \ldots, C\}$) is given by

$$\mu_k = \frac{\sum_{i=1}^{N} u_{ik}^2 \, \mathbf{x}_i}{\sum_{i=1}^{N} u_{ik}^2} \tag{4}$$

and the cardinality of the cluster is defined as

$$N_k = \sum_{i=1}^{N} u_{ik} \tag{5}$$

The first term in (3) is the sum of the squared distances to the prototypes, weighted by the memberships and originates in the FCM cost function. This term attempts to reinforce the compactness of the clusters.

The second term is composed of

- The cost of violating the pairwise *must-link* constraints. The penalty corresponding to the presence of two such points in different clusters is weighted by the corresponding membership values.
- The cost of violating the pairwise *cannot-link* constraints. The penalty corresponding to the presence of two such points in a same cluster is weighted by the membership values.

This term is weighted by α, a factor that specifies the relative importance of the supervision and is discussed later.

The last term in (3), coming from the CA cost function, is the sum of the squares of the cardinalities of the clusters and controls the number of clusters.

When all the terms are combined and α, β have appropriate values, the final partition will minimize the sum of intra-cluster distances, while partitioning the data set into the smallest number of clusters such that the specified constraints are respected as well as possible. When the desired number of clusters is given and the membership degrees are crisp, this cost function can be simplified to obtain the one defined in [2] for the PCKmeans algorithm.

We show in the appendix that the equation for updating memberships is

$$u_{rs} = u_{rs}^{FCM} + u_{rs}^{Constraints} + u_{rs}^{Bias} \tag{6}$$

where

$$u_{rs}^{FCM} = \frac{\frac{1}{d^2(\mathbf{x}_r, \mu_s)}}{\sum_{k=1}^{C} \frac{1}{d^2(\mathbf{x}_r, \mu_k)}} \tag{7}$$

$$u_{rs}^{Constraints} = \frac{\alpha}{2d^2(\mathbf{x}_r, \mu_s)}(\overline{C_{v_r}} - C_{v_{rs}}) \tag{8}$$

$$u_{rs}^{Bias} = \frac{\beta}{d^2(\mathbf{x}_r, \mu_s)}(N_s - \overline{N_r}) \tag{9}$$

In (8), $C_{v_{rs}}$ and $\overline{C_{v_r}}$ are defined as

$$C_{v_{rs}} = \sum_{(\mathbf{x}_r,\mathbf{x}_j)\in\mathcal{M}} \sum_{l=1,l\neq s}^{C} u_{jl} + \sum_{(\mathbf{x}_r,\mathbf{x}_j)\in\mathcal{C}} u_{js} \tag{10}$$

$$\begin{aligned} \overline{C_{v_r}} &= \frac{1}{\sum_{k=1}^{C} \frac{1}{d^2(\mathbf{x}_r,\mu_k)}} \\ &\times \sum_{k=1}^{C} \frac{\Big(\sum_{(\mathbf{x}_r,\mathbf{x}_j)\in\mathcal{M}} \sum_{l=1,l\neq k}^{C} u_{jl}}{d^2(\mathbf{x}_r,\mu_k)} \\ &\quad \underline{+ \sum_{(\mathbf{x}_r,\mathbf{x}_j)\in\mathcal{C}} u_{jk} \Big)} \end{aligned}$$

and $\overline{N_r}$ in (9) is

$$\overline{N_r} = \frac{\sum_{k=1}^{C} \frac{N_k}{d^2(\mathbf{x}_r,\mu_k)}}{\sum_{k=1}^{C} \frac{1}{d^2(\mathbf{x}_r,\mu_k)}} \tag{11}$$

The first term in (6), u_{rs}^{FCM}, is the same as the membership term in FCM and only considers distances between data items and cluster prototypes. The second term, $u_{rs}^{Constraints}$, takes into account the available supervision: memberships are reinforced or deprecated according to the pairwise constraints available. The third term, u_{rs}^{Bias}, controls the competition that leads to a reduction of the cardinality of spurious clusters; as for CA, these clusters are discarded if their cardinality drops below a threshold (see section 4.2).

The β factor controls the competition between clusters and is defined at iteration t by:

$$\begin{aligned} \beta(t) &= \frac{\eta_0 \exp(-|t-t_0|/\tau)}{\sum_{j=1}^{C} \Big(\sum_{i=1}^{N} u_{ij} \Big)^2} \\ &\times \left[\sum_{j=1}^{C} \sum_{i=1}^{N} u_{ij}^2 \, d^2(\mathbf{x}_i,\mu_j) \right] \end{aligned} \tag{12}$$

Because of the exponential component of $\beta(t)$, the last term in (3) will dominate during the early iterations of the algorithm in order to eliminate spurious clusters, then the first two terms will become dominant and help

finding the best partition of the data. Wide ranges of values for η_0 and τ allow competition to operate correctly. The effects of competition can be seen very quickly, so the value of t_0 does not need to be high (we used $t_0 = 5$).

The choice of α is very important for PCCA because α is the weight given to the supervision. Since the number of available constraints is expected to be much lower than the total number of data items, to make sure that constraints have an impact on the clustering process the value of α should balance the first two terms of $\mathcal{J}$ in (3). Also, we consider the *normalized performance index* (NPI, the sum of the squared distances between items and prototypes divided by the sum of the squared memberships) to be a good quantifier of the need for supervision: the higher the value of the NPI, the more we need supervision. We then suggest the following expression for *alpha*:

$$\alpha = \frac{N}{M} \frac{\sum_{k=1}^{C} \sum_{i=1}^{N} u_{ik}^2 \, d^2(\mathbf{x}_i, \mu_k)}{\sum_{k=1}^{C} \sum_{i=1}^{N} u_{ik}^2} \tag{13}$$

M being the number of pairwise constraints available. This expression for α also guarantees that the second term in (3) is commensurate with the others.

The constraint-related terms in (3) can also be added to other fuzzy unsupervised clustering algorithms, such as the Adaptive Robust Competition (ARC, [18]), but the interaction between the semi-supervision and the of these other algorithms has to be studied.

4.2 Cluster Merging

In CA, as clustering proceeds, the clusters whose cardinality drops below a threshold are discarded [13]. This threshold reflects the minimum cardinality of the final clusters. However, this method for discarding the spurious clusters has two drawbacks:

- The outcome of the clustering process is very sensitive to this threshold, even if choosing a minimum cardinality for the clusters is often less arbitrary than setting a desired number of clusters.

- Since well-defined clusters may have very different cardinality numbers, for many data sets there may be no good compromise for the value of the threshold. With a low value, several clusters (each with its own prototype) can co-exist for a single large well-defined cluster. If the

threshold value is high, rather small but well-defined clusters can be incorrectly discarded.

We suggest a strategy for improving the agglomeration process in CA. First, we fix the minimum cardinality threshold to a small percentage of the number of items in the data set, such as to let even small clusters survive. Then we reduce the number of clusters by *merging* the clusters having the nearest prototypes among all possible pairs of clusters. This process is repeated until no more merging is possible. At every iteration we first compute all the distances between prototypes. If $d_{\max} = \max\{d(\mu_k, \mu_h) \mid 1 \leq k, h \leq C\}$, then we merge clusters k and h when

$$\frac{d(\mu_k, \mu_h)}{d_{\max}} < \text{proximity threshold} \tag{14}$$

The proximity threshold was fixed to 0.1 in all the experiments reported here, but can be seen as a *relative resolution* parameter whose value is set by the user according to his desired resolution.

4.3 PCCA Algorithm

The Pairwise-Constrained Competitive Agglomeration algorithm performs the minimization of the cost function 1 and includes the cluster merging method put forward in the previous section. The computational steps are summarized in Algorithm 1 below.

The pairwise constraints are provided before starting PCCA. Depending on the application context, the set of constraints can be available *a priori* or obtained from the user. For producing a set of constraints before starting PCKmeans, Basu et al. [3] proposed a constraint selection method relying on querying the user with pairs of items issued from a farthest-first traversal of the data. A similar method can be employed for PCCA, but we believe that a more attractive solution is to query the user *during* the clustering process, according to the partial results obtained.

5 Experimental Evaluation

We evaluated PCCA by comparing it to CA, the unsupervised clustering algorithm it is based upon, and to the PCKmeans [2] semi-supervised clus-

Algorithm 1 PCCA algorithm outline

fix the maximum number of clusters C;
randomly initialize cluster prototypes;
initialize memberships to $u_{ik} = 1/C$ for all items and clusters;
compute initial cardinality for every cluster;
repeat
  compute β using equation (12);
  compute α using equation (13);
  compute memberships u_{ik} using equation (6);
  compute cardinality for every cluster using equation (5);
  compute $d_{\max} = \max\{d(\mu_k, \mu_h) \mid 1 \leq k, h \leq C\}$;
  for $1 \leq k, h \leq C$ **do**
    if $\frac{d(\mu_k, \mu_h)}{d_{\max}} <$ proximity threshold **then**
      merge clusters k and h;
    end if
  end for
  update the prototypes using equation (4);
until prototypes stabilize

tering algorithm. It is important to note that unsupervised clustering, semi-supervised clustering and supervised classification rely on different assumptions concerning the data, so benchmarks designed for unsupervised clustering or for supervised classification cannot be directly used for the evaluation of semi-supervised clustering. Standard benchmarks for unsupervised algorithms prove to be too simple for semi-supervised clustering, while benchmarks for supervised classification may be too difficult. Since the semi-supervised approach has a short history, few specific benchmarks exist. We selected for our evaluation the well-known IRIS benchmark (also used in [2]) containing 3 classes of 50 instances each and a ground truth image database containing 4 classes of 100 images each. A sample taken from the image database is shown in Figure 2. The classes are rather diverse and many images belonging to different classes are quite similar. In both experiments, random pairs of data items are selected and corresponding constraints are provided from the ground truth: depending on whether the two items belong to the same class or not, a must-link or a cannot-link constraint is generated.

The image features we used are the Laplacian weighted histogram, the probability weighted histogram, the Hough histogram, the Fourier histogram

and a classical color histogram obtained in HSV color space. The dimension of the joint feature vector (originally above 600) was reduced of about 5 times using linear principal component analysis.

Since the shape of the clusters is usually not spherical, we employ the Mahalanobis distance rather than the classical Euclidean distance.

Figures 3 and 4 present the dependence between the percentage of well-categorized data items and the number of pairwise constraints considered, for each of the two data sets. The graphs for CA and for K-means (both ignoring the constraints) are shown for reference.

The correct number of classes was directly provided to K-means and PCK-means. CA and PCCA were initialized with a significantly larger number of classes and found the appropriate number (i.e. the one in the ground truth) by themselves.

For the fuzzy algorithms (CA and PCCA) every data item is assigned to the cluster to which its membership value is the highest. For every number of constraints, 100 experiments were performed with different random selections of the pairs of data items for which constraints are provided (from the ground truth). This resulted in error bars for PCCA and for PCKmeans.

The significant difference between the graphs for the unsupervised clustering and the semi-supervised clustering algorithms clearly shows that, by providing a simple form of semantic information (the pairwise constraints) the quality of the resulting categories can be significantly improved. It can also be seen that the number of pairwise constraints required for reaching such an improvement is relatively low with respect to the number of items in the data sets.

With a similar number of constraints, PCCA performs significantly better than PCKmeans by making a better use of the available constraints. The signed constraint terms in (10) let the fuzzy clustering process directly take into account the pairwise constraints.

The error bars given for PCCA and PCKmeans in Fig. 3 and 4 show that, with a given number of constraints, there is a significant variance in the quality of the final clustering results. Performance clearly depends not only on the number of constraints, but also on the *specific* constraints employed. It is then useful to study criteria for finding the constraints that are potentially the most informative for the clustering algorithm.

6 Conclusion

We have shown that the provision of a limited amount of simple semantic information—pairwise constraints—brings the results obtained for the categorization of the images in a database closer to user's expectations. We put forward a new semi-supervised clustering algorithm, Pairwise-Constrained Competitive Agglomeration, based on a fuzzy cost function that directly takes pairwise constrains into account.

Experimental evaluation on the Iris data set and on a ground truth image database shows that PCCA performs considerably better than Competitive Agglomeration, the unsupervised algorithm PCCA is based upon, and than PCKMeans, an existing semi-supervised clustering algorithm. By making better use of the constraints, PCCA allows the number of constraints to remain sufficiently low for this semi-supervised approach to be an interesting alternative in the categorization of image databases. Also, the computational complexity of PCCA is linear in the number of data vectors and in the number of clusters, making this algorithm suitable for real-world clustering applications.

We have found experimentally that performance also depends on the specific constraints selected. In an attempt to diminish the number of constraints required in a scenario where constraints are provided interactively by the user, we currently explore *active* mechanisms for the selection of pairs of candidate items. We are also working towards a further reduction in the computational complexity in order to be able to categorize fast very large image databases.

7 Acknowledgements

The authors are very grateful to the anonymous reviewers for their valuable suggestions. Part of this work is supported by the European Union Network of Excellence "MUSCLE" (http://www.muscle-noe.org/).

References

[1] J. D. Banfield and A. E. Raftery. Model-based gaussian and non-gaussian clustering. *Biometrics*, 49:803–821, 1993.

[2] S. Basu, A. Banerjee, and R. J. Mooney. Semi-supervised clustering by seeding. In *19th International Conference on Machine Learning (ICML'02)*, pages 19–26, 2002.

[3] S. Basu, A. Banerjee, and R. J. Mooney. Active semi-supervision for pairwise constrained clustering. In *SIAM International Conference on Data Mining (SDM'04)*, pages 251–258, 2004.

[4] S. Basu, M. Bilenko, and R. J. Mooney. Comparing and unifying search-based and similarity-based approaches to semi-supervised clustering. In *Proceedings of the ICML-2003 Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, pages 42–49, Washington, DC, August 2004.

[5] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, 1981.

[6] M. Bilenko and R. J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *International Conference on Knowledge Discovery and Data Mining*, pages 39–48, Washington, DC, 2003.

[7] N. Boujemaa. On competitive unsupervized clustering. In *Proceedings of the International Conference on Pattern Recognition (ICPR'2000)*, volume 1, pages 631–634, Barcelona, Spain, September 2000.

[8] G. Celeux and G. Govaert. Gaussian parsimonious clustering models. *Pattern Recognition*, 28(5):781–793, 1995.

[9] D. Cohn, R. Caruana, and A. McCallum. Semi-supervised clustering with user feedback. Technical report, Cornell University, United States, 2003.

[10] R. N. Dave. Characterization and detection of noise in clustering. *Pattern Recognition Letters*, 12:657–664, 1991.

[11] A. Demiriz, K. Bennett, and M. Embrechts. Semi-supervised clustering using genetic algorithms. In C. H. D. et al., editor, *Intelligent Engineering Systems Through Artificial Neural Networks 9*, pages 809–814. ASME Press, 1999.

[12] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[13] H. Frigui and R. Krishnapuram. Clustering by competitive agglomeration. *Pattern Recognition*, 30(7):1109–1119, 1997.

[14] E. E. Gustafson and W. C. Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *IEEE CDC*, pages 761–766, San Diego, California, 1979.

[15] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[16] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[17] D. Klein, S. Kamvar, and C. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *Proceedings of the The Nineteenth International Conference on Machine Learning*, pages 63–70, Sydney, Australia, 2002.

[18] B. Le Saux and N. Boujemaa. Unsupervised robust clustering for image database categorization. In *Proceedings of the IEEE-IAPR International Conference on Pattern Recognition (ICPR'2002)*, pages 357–362, August 2002.

[19] J. McQueen. Some methods for classifcation and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathmatical Statistics and Probabilty*, pages 281–297, 1967.

[20] K. Wagstaff and C. Cardie. Clustering with instance-level constraints. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1103–1110, 2000.

[21] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 505–512. MIT Press, Cambridge, MA, 2003.

8 Appendix

To minimize (3) with respect to $\mathbf{V}$ and $\mathbf{U}$ under the constraints (2), we introduce the Lagrange multipliers λ_i, $i \in \{1, .., N\}$ and have

$$\begin{aligned}
\mathcal{J}_\Lambda(\mathbf{V}, \mathbf{U}) &= \sum_{k=1}^{C} \sum_{i=1}^{N} u_{ik}^2 \, d^2(\mathbf{x}_i, \mu_k) \\
&+ \alpha \Bigg(\sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}} \sum_{k=1}^{C} \sum_{l=1, l \neq k}^{C} u_{ik} \, u_{jl} \\
&+ \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}} \sum_{k=1}^{C} u_{ik} \, u_{jk} \Bigg) \\
&- \beta \sum_{k=1}^{C} \left(\sum_{i=1}^{N} u_{ik} \right)^2 \\
&- \sum_{i=1}^{N} \lambda_i \left(\sum_{k=1}^{C} u_{ik} - 1 \right)
\end{aligned} \tag{15}$$

The cluster prototypes and the memberships that produce extreme values for $\mathcal{J}(\mathbf{V}, \mathbf{U})$ under the constraints (2) must satisfy

$$\frac{\partial \mathcal{J}_\Lambda}{\partial \mu_{sj}}(\mathbf{V}, \mathbf{U}) = 0 \quad \text{and} \quad \frac{\partial \mathcal{J}_\Lambda}{\partial u_{rs}}(\mathbf{V}, \mathbf{U}) = 0 \tag{16}$$

for $s \in \{1, \ldots, C\}$, $r \in \{1, \ldots, N\}$, where μ_{sj} is the j-th component of cluster prototype μ_s. When computing the partial derivatives we ignore the dependencies through α and β. Then, the first condition directly produces the expression (4) for updating the prototypes. The second condition in (16) becomes

$$\begin{aligned}
\frac{\partial \mathcal{J}_\Lambda}{\partial u_{rs}} &= 2 u_{rs} d^2(\mathbf{x}_r, \mu_s) \\
&- 2\beta \sum_{i=1}^{N} u_{is} - \lambda_r \\
&+ \alpha \Big(\sum_{(\mathbf{x}_r, \mathbf{x}_j) \in \mathcal{M}} \sum_{l=1, l \neq s}^{C} u_{jl} \\
&+ \sum_{(\mathbf{x}_r, \mathbf{x}_j) \in \mathcal{C}} u_{js} \Big) \\
&= 0
\end{aligned} \tag{17}$$

for $s \in \{1, \ldots, C\},\ r \in \{1, \ldots, N\}$, to be solved together with (2).

To obtain the updating equation for the memberships, we assume that the cardinality of the clusters ($N_s = \sum_{i=1}^{N} u_{is}$ for cluster s, $s \in \{1, \ldots, C\}$) does not change significantly from one iteration to the next, so we can use the values obtained in the previous iteration. With this assumption, (17) becomes

$$\begin{aligned} u_{rs} &= \frac{2\beta N_s + \lambda_r}{2d^2(\mathbf{x}_r, \mu_s)} \\ &- \alpha \frac{\sum_{(\mathbf{x}_r, \mathbf{x}_j) \in \mathcal{M}} \sum_{l=1, l \neq s}^{C} u_{jl} + \sum_{(\mathbf{x}_r, \mathbf{x}_j) \in \mathcal{C}} u_{js}}{2d^2(\mathbf{x}_r, \mu_s)} \end{aligned} \tag{18}$$

From (18) and (2) we obtain

$$\begin{aligned} & \sum_{k=1}^{C} \frac{2\beta N_k + \lambda_r}{2d^2(\mathbf{x}_r, \mu_k)} \\ - & \alpha \sum_{k=1}^{C} \frac{\sum_{(\mathbf{x}_r, \mathbf{x}_j) \in \mathcal{M}} \sum_{l=1, l \neq k}^{C} u_{jl}}{2d^2(\mathbf{x}_r, \mu_k)} \\ & \underline{+ \sum_{(\mathbf{x}_r, \mathbf{x}_j) \in \mathcal{C}} u_{jk}} \\ = & \ 1 \end{aligned} \tag{19}$$

As a consequence,

$$\begin{aligned} \lambda_r &= \frac{1}{\sum_{k=1}^{C} \frac{1}{d^2(\mathbf{x}_r, \mu_k)}} \left[1 - \beta \sum_{k=1}^{C} \frac{N_k}{2d^2(\mathbf{x}_r, \mu_k)} \right. \\ &+ \alpha \sum_{k=1}^{C} \frac{\sum_{(\mathbf{x}_r, \mathbf{x}_j) \in \mathcal{M}} \sum_{l=1, l \neq k}^{C} u_{jl}}{2d^2(\mathbf{x}_r, \mu_k)} \\ & \left. \underline{+ \sum_{(\mathbf{x}_r, \mathbf{x}_j) \in \mathcal{C}} u_{jk}} \right] \end{aligned} \tag{20}$$

Substituting (20) in (17), we obtain the final update equation for the membership of the data item $\mathbf{x}_r$ to the cluster μ_s:

$$u_{rs} = \frac{\frac{1}{d^2(\mathbf{x}_r, \mu_s)}}{\sum_{k=1}^{C} \frac{1}{d^2(\mathbf{x}_r, \mu_k)}}$$

$$
\begin{aligned}
&+ \quad \frac{\beta}{d^2(\mathbf{x}_r,\mu_s)}\left[N_s - \frac{\sum_{k=1}^{C}\frac{N_k}{d^2(\mathbf{x}_r,\mu_k)}}{\sum_{k=1}^{C}\frac{1}{d^2(\mathbf{x}_r,\mu_k)}}\right] \\
&+ \quad \frac{\alpha}{2d^2(\mathbf{x}_r,\mu_s)\sum_{k=1}^{C}\frac{1}{d^2(\mathbf{x}_r,\mu_k)}} \\
&\times \quad \sum_{k=1}^{C}\frac{\Big(\sum_{(\mathbf{x}_r,\mathbf{x}_j)\in\mathcal{M}}\sum_{l=1,l\neq k}^{C}u_{jl}}{d^2(\mathbf{x}_r,\mu_k)} \\
&\qquad \underline{+\sum_{(\mathbf{x}_r,\mathbf{x}_j)\in\mathcal{C}}u_{jk}\Big)} \\
&- \quad \frac{\alpha}{2d^2(\mathbf{x}_r,\mu_s)}\Bigg(\sum_{(\mathbf{x}_r,\mathbf{x}_j)\in\mathcal{M}}\sum_{l=1,l\neq s}^{C}u_{jl} \\
&+ \quad \sum_{(\mathbf{x}_r,\mathbf{x}_j)\in\mathcal{C}}u_{js}\Bigg)
\end{aligned}
$$

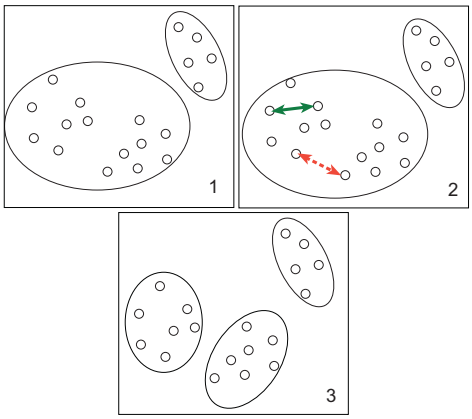

Figure 1: An illustration of semi-supervised clustering using pairwise constraints. The clustering process takes into account the *must-link* (continuous line) and *cannot-link* (dashed line) constraints provided.

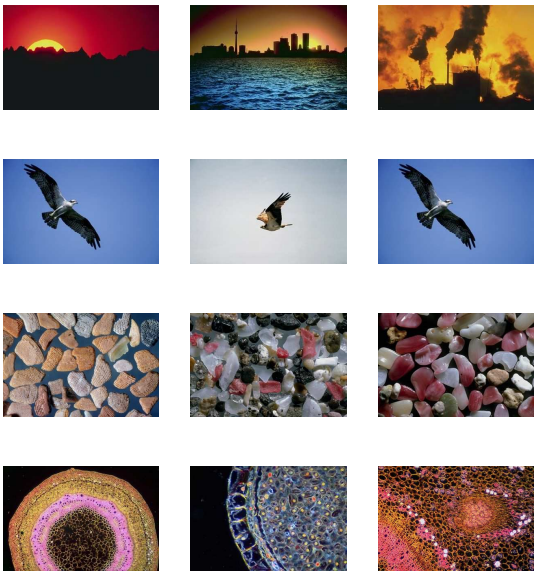

Figure 2: Every line shows a sample of images from a different class of the image database.

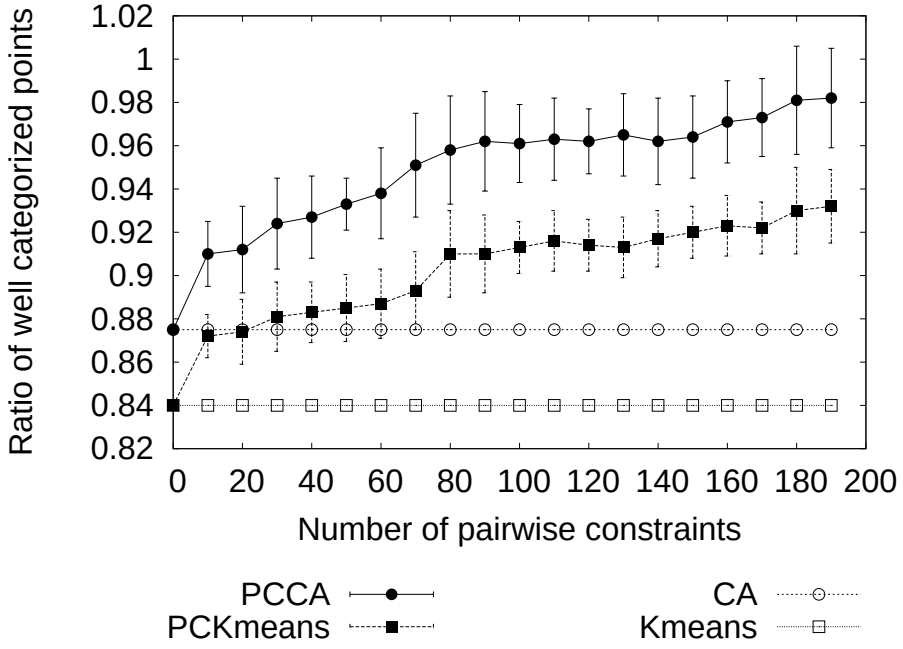

Figure 3: Results obtained by the different clustering algorithms on the Iris benchmark

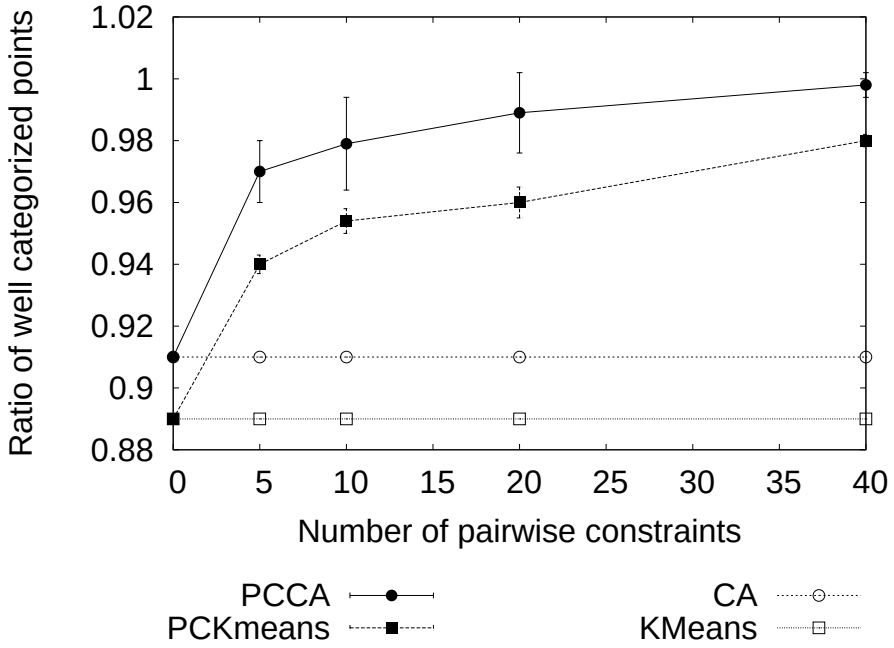

Figure 4: Results obtained by the different clustering algorithms on the ground truth image database