PERGAMON

Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Pattern Recognition 37 (2004) 567–581

PATTERN
RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Unsupervised learning of prototypes and attribute weights☆

Hichem Frigui*, Olfa Nasraoui

*Department of Electrical and Computer Engineering, The University of Memphis, 206 Engineering Science Bldg.,
Memphis, TN 38152, USA*

Received 25 February 2003; accepted 4 August 2003

Abstract

In this paper, we introduce new algorithms that perform clustering and feature weighting simultaneously and in an unsupervised manner. The proposed algorithms are computationally and implementationally simple, and learn a different set of feature weights for each identified cluster. The cluster dependent feature weights offer two advantages. First, they guide the clustering process to partition the data set into more meaningful clusters. Second, they can be used in the subsequent steps of a learning system to improve its learning behavior. An extension of the algorithm to deal with an unknown number of clusters is also proposed. The extension is based on competitive agglomeration, whereby the number of clusters is over-specified, and adjacent clusters are allowed to compete for data points in a manner that causes clusters which lose in the competition to gradually become depleted and vanish. We illustrate the performance of the proposed approach by using it to segment color images, and to build a nearest prototype classifier.

Keywords: Feature weighting; Fuzzy clustering; Competitive agglomeration; Image segmentation; Nearest prototype classifier

1. Introduction

The problem of selecting or weighting the best subset of attributes constitutes an important part of the design of good learning algorithms for real word tasks. The generalization performance of these algorithms can be significantly degraded if irrelevant attributes are used. As a result, several methods have been proposed for feature selection and weighting [1–4]. In feature selection, the task's dimensionality is reduced by completely eliminating irrelevant features. This amounts to assigning binary relevance weights to the features (1 for relevant, and 0 for irrelevant). Feature weighting is an extension of the selection process where the features are assigned continuous weights which can be regarded as degrees of relevance. Continuous weighting provides a richer feature relevance representation. Therefore, it tends to outperform feature selection from an accuracy point of view in tasks where some features are useful but less important than others. Most feature weighting (and selection) methods assume that feature relevance is invariant over the task's domain, and hence learn a single set of weights for the entire data set. This assumption can impose unnecessary and pernicious constraints on the learning task when the data set is made of different categories or classes. If the data is already labeled (supervised learning), then it is possible to learn a different set of weights for each class. For instance, in [5], context-sensitive feature selection is introduced as a mechanism to improve lazy supervised learning techniques. However, this technique requires labeled data, and relates the relevance of a given feature in terms of the feature values of other features.

On the other hand, if the data set is unlabeled (unsupervised learning), then no methods exist for assigning different relevance weights for the different categories of a data set prior to clustering. The classical approach in this case is to

☆ This material is based upon work supported by the National Science Foundation Career Awards No. IIS-0133415 to H. Frigui and No IIS-0133948 to O. Nasraoui.

* Corresponding author. Tel.: +1-901-624-6928; fax: +1-901-624-5469.

E-mail address: hfrigui@memphis.edu (H. Frigui).

0031-3203/$30.00 © 2003 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2003.08.002

determine a single subset of features or feature weights for the entire unlabeled data set prior to clustering. However, by ignoring the existence of different sub-structures in the data set, which require different subsets of feature weights, the performance of any clustering procedure can be severely degraded. Hence, it is clear that the clustering and feature selection/weighting steps are coupled, and applying these steps in sequence can degrade the performance of the learning system. In fact, even if the data set is labeled according to several known classes, it is preferable to model each complex (non-convex) class by several simple sub-classes or clusters, and to use a different set of feature weights for each cluster.

To illustrate the need for different sets of feature weights for different clusters, we consider the unsupervised learning problem of segmenting the color image shown in Fig. 1(a). In this problem, the pixels of the image must be categorized into meaningful clusters corresponding to different homogeneous regions of the image without any prior knowledge about its contents. In order to be categorized, the pixels are first mapped to feature vectors as explained in Section 5.1. Fig. 1(b) and (c) show the segmentation results with different feature subsets using the fuzzy C-means (FCM) algorithm [6], when the number of clusters is fixed to 4. The FCM, like other clustering algorithms, has no provision for cluster dependent feature weights. Any given feature must either be used (completely relevant) or ignored (irrelevant) for all clusters. Fig. 1(b) shows the segmentation results when the x and y position features are not used during clustering. It can be seen that some clusters suffer from a fragmentation phenomenon. That is, instead of being clean and compact, these clusters are scattered around the image. Fig. 1(c) shows the results when the x and y position features are used. As expected, the clusters obtained are more compact. However, except for the tiger cluster, the remaining clusters do not correspond to homogeneous regions in the original image. This is because, for example for the dirt region, the y-position feature is relevant while the x-position feature is irrelevant since this region forms a narrow strip that extends over the entire width of the image. For the grass region, both the x and y position features are irrelevant because grass can be found in almost all areas of the image. We conclude that in this case, the x and y position features are both useful, and thus relevant for delineating clusters corresponding to the compact regions of the image such as the tiger. However, that is not the case for non-compact regions such as the background. The remaining features cannot be easily visualized as in the case of position. However, their relevance is also expected to vary across the different regions of the image because of the inherent differences in their color and texture properties. This simple example shows how feature relevance can vary widely within the domain of a data set.

In light of our discussion, it is only natural to conclude that ideally, clustering and feature selection should be performed simultaneously in an unsupervised learning problem. In this paper, we propose a new approach, called simultaneous clustering and attribute discrimination (SCAD), that performs clustering and feature weighting *simultaneously*. When used in conjunction with a supervised or unsupervised learning system, SCAD offers several advantages. First, it uses continuous feature weighting, hence providing a much richer feature relevance representation than feature selection. Secondly, SCAD learns the feature relevance representation of each cluster independently and in an unsupervised manner. Moreover, SCAD can adapt to the variations that exist within a data set by categorizing it into distinct clusters, which are allowed to overlap because of the use of fuzzy membership degrees. Based on the SCAD approach, we will present two new clustering algorithms, called SCAD1 and SCAD2, respectively. These two algorithms achieve the same goal, however, they minimize different objective functions. Finally, because the number of categories in a data set is not always known a priori, we present an extension of SCAD that can determine the number of clusters using competitive agglomeration [7].

The rest of the paper is organized as follows. In Section 2, we review existing approaches to feature selection/weighting, and prototype-based clustering algorithms. In Section 3, we present our new algorithms: SCAD1 and SCAD2. In Section 4, we extend SCAD2 to the case where the number of clusters is unknown. In Section 5, we apply SCAD2 to segmenting color images in an unsupervised manner. In Section 6, we present a second application that uses SCAD2 to build a nearest prototype classifier in a supervised learning environment. Finally, Section 7 contains the summary conclusions.

2. Background

2.1. Feature selection and feature weighting

Feature selection and weighting techniques generally rely on a criterion function and a search strategy. The criterion function is used to decide whether a feature subset is better than another, while the search strategy determines feature subset candidates. Depending on the criterion function used, there are two types of feature selection/weighting methods: the *wrapper* and *filter* approaches [8]. The wrapper approach relies on feedback from the performance algorithm, such as classifier accuracy, to learn feature weights or to decide whether a particular feature subset is superior to another. The filter approach optimizes a classifier independent criterion function. The wrapper approach tends to perform better, however it can cause over-fitting [9]. Moreover, it should only be applied in combination with classifiers of low complexity to limit its computational cost.

Feature selection methods have exploited several search strategies. The most rudimentary strategy, exhaustive search, considers $2^n - 1$ (where n is the maximum number of features) possible feature subsets, and is impractical for large n.

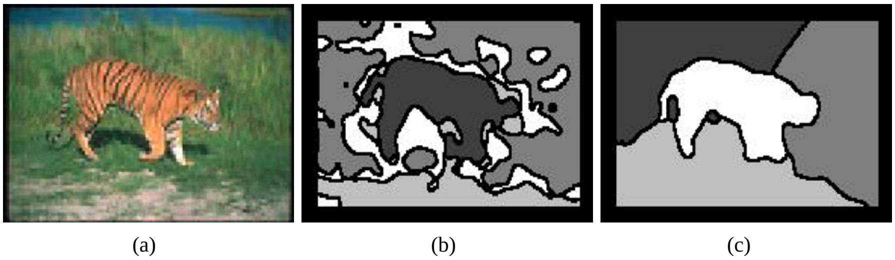


Fig. 1. (a) Original color image, (b) results of the FCM with no position features, (c) results of the FCM with position features.

As a result, other optimized search strategies such as forward selection and backward selection [10] can be used. These approaches are computationally feasible. Unfortunately, they are only appropriate for binary weighting and are prone to yielding sub-optimal solutions [8]. Other strategies include random mutation hill climbing [11], and parallel search [12].

Feature selection/weighting can either be applied to the entire data set to obtain a single set of features or weights, or to each class independently to obtain a different set of features or weights for each class. It is also possible to use the entire data set and the class labels to learn a different set of weights for each class [13,11]. In general, class dependent feature selection/weighting is superior to the approach yielding a single set of features/weights. Unfortunately, class dependent feature selection/weighting cannot be applied when the data set is unlabeled.

2.2. Prototype-based clustering

Let $\mathbf{X} = \{\mathbf{x}_j \mid j = 1, \ldots, N\}$ be a set of N feature vectors in an n-dimensional feature space. Let $\mathbf{B} = (\beta_1, \ldots, \beta_c)$ represent a C-tuple of prototypes each of which characterizes one of the C clusters. Each β_i consists of a set of parameters. Let u_{ij} represent the grade of membership of feature point $\mathbf{x}_j$ in cluster β_i. The $C \times N$ matrix $\mathbf{U} = [u_{ij}]$ is called a constrained fuzzy C-partition matrix if it satisfies the following conditions [6]:

$$u_{ij} \in [0,1] \quad \forall i; \qquad 0 < \sum_{j=1}^{N} u_{ij} < N \quad \forall i,j;$$

$$\sum_{i=1}^{C} u_{ij} = 1 \quad \forall j. \tag{1}$$

The problem of fuzzily partitioning the feature vectors into C clusters can be formulated as the minimization of an objective function $J(\mathbf{B}, \mathbf{U}; \mathbf{X})$ of the form

$$J(\mathbf{B}, \mathbf{U}; \mathbf{X}) = \sum_{i=1}^{C} \sum_{j=1}^{N} (u_{ij})^m d_{ij}^2, \tag{2}$$

subject to the constraint in (1). In the above equation, $m \in [1, \infty)$ is a weighting exponent called the fuzzifier, and d_{ij}^2 represents the distance from a feature point $\mathbf{x}_j$ to the prototype β_i. Minimization of (2) with respect to $\mathbf{U}$ subject to (1) gives us [6]

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} (d_{ij}^2 / d_{kj}^2)^{1/(m-1)}}. \tag{3}$$

Minimization of (2) with respect to $\mathbf{B}$ varies according to the choice of the prototypes and the distance measure. For example, in the fuzzy C-means (FCM) algorithm, the Euclidean distance is used, and each of the prototypes is described by the cluster center $\mathbf{c}_i$, which may be updated in each iteration using [6]

$$\mathbf{c}_i = \frac{\sum_{j=1}^{N} u_{ij}^m \mathbf{x}_j}{\sum_{j=1}^{N} u_{ij}^m}. \tag{4}$$

2.3. Competitive agglomeration

The objective function in (2), which is essentially the sum of (fuzzy) intra-cluster distances, has a monotonic tendency with respect to the number of clusters, C, and has the minimum value of zero when $C = N$. Therefore, it is not useful for the automatic determination of the “optimum” number of clusters, and C has to be specified a priori. The Competitive Agglomeration (CA) algorithm [14,7] overcomes this drawback by adding a second regularization term to prevent over-fitting the data set with too many prototypes. The CA algorithm starts by partitioning the data set into a large number of small clusters. As the algorithm progresses, adjacent clusters compete for data points, and clusters that lose in the competition gradually vanish. The CA algorithm minimizes the following objective function

$$J_A(\mathbf{B}, \mathbf{U}; \mathbf{X}) = \sum_{i=1}^{C} \sum_{j=1}^{N} (u_{ij})^2 d_{ij}^2 - \alpha \sum_{i=1}^{C} \left[\sum_{j=1}^{N} u_{ij} \right]^2 \tag{5}$$

subject to the constraints in (1). It should be noted that the number of clusters C in (5) is dynamically updated in the CA. The first term in (5) controls the shape and size of the clusters and encourages partitions with many clusters, while the second term penalizes solutions with a large number of clusters and encourages the agglomeration of clusters. When both terms are combined and α is chosen properly, the final partition will minimize the sum of intra-cluster distances, while partitioning the data into the smallest possible number of clusters. It can be shown [7] that the membership update equation for (5) is given by

$$u_{ij} = \frac{1/d_{ij}^2}{\sum_{k=1}^{C} 1/d_{kj}^2} + \frac{\alpha}{d_{ij}^2}(N_i - \bar{N}_j), \tag{6}$$

In (6), $N_i = \sum_{j=1}^{N} u_{ij}$, is the cardinality of cluster i, and $\bar{N}_j = (\sum_{k=1}^{C} N_k/d_{kj}^2)/(\sum_{k=1}^{C} 1/d_{kj}^2)$ is a weighted average of the cardinalities of all clusters. The first term in (6) is the membership term in the FCM algorithm [6] (see Eq. (3)) which takes into account only the relative distances of the feature point to all clusters. The second term is a signed bias term which allows good clusters to agglomerate and spurious clusters to disintegrate.

The value of α needs to be initially small to encourage the formation of small clusters. Then, it should be increased gradually to promote agglomeration. After a few iterations, when the number of clusters becomes close to the "optimum", the value of α should again decay slowly to allow the algorithm to converge. In [7], α is updated in every iteration (t) using

$$\alpha(t) = \eta(t)\frac{\sum_{i=1}^{C}\sum_{j=1}^{N}(u_{ij}^{(t-1)})^2(d_{ij}^2)^{(t-1)}}{\sum_{i=1}^{C}[\sum_{j=1}^{N} u_{ij}^{(t-1)}]^2}, \tag{7}$$

where

$$\eta(t) = \begin{cases} \eta_0 e^{-|t_0 - t|/\tau} & \text{if } t > 0, \\ 0 & \text{if } t = 0. \end{cases} \tag{8}$$

In (8), η_0 is the initial value, τ is a time constant, and t_0 is the iteration number at which η starts to decrease. The superscript $(t-1)$ is used on u_{ij} and d_{ij}^2 to denote their values in the previous iteration, $(t-1)$.

3. Simultaneous clustering and attribute discrimination

The proposed approach to perform clustering and attribute discrimination is designed to search for the optimal prototype parameters, $\mathbf{B}$, and the optimal set of feature weights, $\mathbf{V}$, simultaneously. Each cluster i is allowed to have its own set of feature weights $\mathbf{V}_i = [v_{i1}, \dots, v_{in}]$. We present two versions of the SCAD algorithm: SCAD1 tries to balance between the two terms of a compound objective function in order to determine the optimal attribute relevance weights, while SCAD2 minimizes a single term criterion.

3.1. Simultaneous clustering and attribute discrimination—version 1 (SCAD1)

The SCAD1 algorithm minimizes the following objective function:

$$J_1(\mathbf{C}, \mathbf{U}, \mathbf{V}; \mathbf{X}) = \sum_{i=1}^{C}\sum_{j=1}^{N} u_{ij}^m \sum_{k=1}^{n} v_{ik} d_{ijk}^2 + \sum_{i=1}^{C}\delta_i \sum_{k=1}^{n} v_{ik}^2, \tag{9}$$

subject to the constraint on fuzzy memberships, u_{ij} in (1), and the following constraint:

$$v_{ik} \in [0,1] \ \forall i, k; \quad \text{and} \quad \sum_{k=1}^{n} v_{ik} = 1, \ \forall i. \tag{10}$$

In (9), v_{ik} represents the relevance weight of feature k in cluster i, and d_{ijk} is given by

$$d_{ijk} = |x_{jk} - c_{ik}|, \tag{11}$$

where x_{jk} is the kth feature value of data point $\mathbf{x}_j$, and c_{ik} is the kth component of the ith cluster center vector. In other words, d_{ijk} is the projection of the displacement vector between feature point $\mathbf{x}_j$ and the ith class center $(\mathbf{c}_i)$ along the kth dimension.

The objective function in (9) has two components. The first component, which is similar to the FCM objective function [6], is the sum of feature-weighted Euclidean distances to the prototypes, additionally weighted by constrained memberships. This component allows us to obtain compact clusters. From a feature-relevance point of view, this term is minimized when, in each cluster, only one feature is completely relevant, while all other features are irrelevant. The second component in Eq. (9) is the sum of the squared feature weights. The global minimum of this component is achieved when all the features are equally weighted. When both components are combined as in (9), and the coefficients δ_i are chosen properly, the final partition will minimize the sum of intra-cluster weighted distances, where the weights are optimized for each cluster.

To optimize J_1, with respect to $\mathbf{V}$, we use the Lagrange multiplier technique, and obtain

$$J_1(\mathbf{\Lambda}, \mathbf{V}) = \sum_{i=1}^{C}\sum_{j=1}^{N}(u_{ij})^m \sum_{k=1}^{n} v_{ik} d_{ijk}^2 + \sum_{i=1}^{C}\delta_i \sum_{k=1}^{n} v_{ik}^2 - \sum_{i=1}^{C}\lambda_i\left(\sum_{k=1}^{n} v_{ik} - 1\right),$$

where $\Lambda = [\lambda_1, \dots, \lambda_c]^t$. Since the rows of $\mathbf{V}$ are independent of each other, we can reduce the above optimization problem to the following C independent problems:

$$J_{1i}(\lambda_i, \mathbf{V}_i) = \sum_{j=1}^{N}(u_{ij})^m \sum_{k=1}^{n} v_{ik}(x_{jk} - c_{ik})^2 + \delta_i \sum_{k=1}^{n} v_{ik}^2 - \lambda_i\left(\sum_{k=1}^{n} v_{ik} - 1\right),$$

for $i = 1, \dots, C$, where $\mathbf{V}_i$ is the ith row of $\mathbf{V}$. By setting the gradient of J_{1i} to zero, we obtain

$$\frac{\partial J_{1i}(\lambda_i, \mathbf{V}_i)}{\partial \lambda_i} = \left(\sum_{k=1}^{n} v_{ik} - 1\right) = 0 \tag{12}$$

and

$$\frac{\partial J_{1i}(\lambda_i, \mathbf{V}_i)}{\partial v_{ik}} = \sum_{j=1}^{N} (u_{ij})^m d_{ijk}^2 + 2\delta_i v_{ik} - \lambda_i = 0. \tag{13}$$

Solving (12) and (13) for v_{ik}, we obtain

$$v_{ik} = \frac{1}{n} + \frac{1}{2\delta_i} \sum_{j=1}^{N} (u_{ij})^m \left[\frac{\|\mathbf{x}_j - \mathbf{c}_i\|^2}{n} - d_{ijk}^2 \right]. \tag{14}$$

The first term in (14), $(1/n)$, is the default value if all attributes are treated equally, and no feature discrimination is performed. The second term is a bias that can be either positive or negative. It is positive for compact features where the projected distance along the corresponding dimension is, on average, less than the average projected distance values along all the dimensions. In other words, if an attribute is very compact, compared to the other attributes, for most of the points that belong to a given cluster (high u_{ij}), then it is considered to be very relevant for that cluster.

The choice of δ_i in Eq. (9) is important to the performance of SCAD1 since it reflects the importance of the second term relative to the first term. If δ_i is too small, then the first term dominates, and only one feature in cluster i will be maximally relevant and assigned a weight of one, while the remaining features get assigned zero weights. On the other hand, if δ_i is too large, then the second term will dominate, and all features in cluster i will be relevant, and assigned equal weights of $1/n$. Hence, the values of δ_i should be chosen such that both terms are of the same order of magnitude. This can be accomplished by computing δ_i in iteration, t, as follows:

$$\delta_i^{(t)} = K \frac{\sum_{j=1}^{N} (u_{ij}^{(t-1)})^m \sum_{k=1}^{n} v_{ik}^{(t-1)} (d_{ijk}^{(t-1)})^2}{\sum_{k=1}^{n} (v_{ik}^{(t-1)})^2}. \tag{15}$$

In (15), K is a constant, and the superscript $(t-1)$ is used on u_{ij}, v_{ik}, and d_{ijk} to denote their values in iteration $(t-1)$.

It should be noted that depending on the values of δ_i, the feature relevance values v_{ik} may not be confined to [0,1]. This inequality constraint could be added to the objective function in (9), and Karush–Kuhn–Tucker (KKT) Theorem [15] could be used to derive the necessary conditions. A much simpler and practical heuristic, that proved almost as effective as the KKT optimization conditions is to simply set negative values to zero, and to clip values that are greater than one to one. However, if this constraint is violated very often, then it is an indication that the value of δ is too small, and that it should be increased (increase K).

To minimize J_1 with respect to $\mathbf{U}$, we rewrite the objective function in (9) as

$$J_1(\mathbf{C}, \mathbf{U}, \mathbf{V}; \mathbf{X}) = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^m \tilde{d}_{ij}^2 + \sum_{i=1}^{C} \delta_i \sum_{k=1}^{n} v_{ik}^2, \tag{16}$$

where $\tilde{d}_{ij}^2 = \sum_{k=1}^{n} v_{ik} d_{ijk}^2$, is the weighted Euclidean distance. Since the second term in (16) does not depend on u_{ij} explicitly, the update equation of the memberships is similar to that of the FCM (see Eq. (3)), i.e.,

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} (\tilde{d}_{ij}^2 / \tilde{d}_{kj}^2)^{1/(m-1)}}. \tag{17}$$

To minimize J_1 with respect to the centers, we fix $\mathbf{U}$ and $\mathbf{V}$, and set the gradient to zero. We obtain

$$\frac{\partial J}{\partial c_{ik}} = -2 \sum_{j=1}^{N} (u_{ij})^m v_{ik} (x_{jk} - c_{ik}) = 0. \tag{18}$$

Solving (18) for c_{ik}, we obtain

$$c_{ik} = \frac{v_{ik} \sum_{j=1}^{N} (u_{ij})^m x_{jk}}{v_{ik} \sum_{j=1}^{N} (u_{ij})^m}. \tag{19}$$

There are two cases depending on the value of v_{ik}.

Case 1: $v_{ik} = 0$. In this case, the kth feature is completely irrelevant relative to the ith cluster. Hence, regardless of the value of c_{ik}, the values of this feature will not contribute to the overall weighted distance computation. Therefore, in this situation, any arbitrary value can be chosen for c_{ik}. In practice, we set $c_{ik} = 0$.

Case 2: $v_{ik} \neq 0$. For the case when the kth feature has some relevance to the ith cluster, Eq. (19) reduces to

$$c_{ik} = \frac{\sum_{j=1}^{N} (u_{ij})^m x_{jk}}{\sum_{j=1}^{N} (u_{ij})^m}.$$

To summarize, the update equation for the centers is

$$c_{ik} = \begin{cases} 0 & \text{if } v_{ik} = 0, \\ \dfrac{\sum_{j=1}^{N} (u_{ij})^m x_{jk}}{\sum_{j=1}^{N} (u_{ij})^m} & \text{if } v_{ik} > 0. \end{cases} \tag{20}$$

The SCAD1 algorithm is summarized below.

Simultaneous Clustering and Attribute Discrimination: SCAD1

```
Fix the number of clusters $C$;
Fix $m$, $m \in [1, \infty)$;
Initialize the centers;
Initialize the relevance weights to $1/n$;
Initialize the fuzzy partition matrix $\mathbf{U}$;
REPEAT
  Compute $d_{ijk}^2$ for $1 \leqslant i \leqslant C$, $1 \leqslant j \leqslant N$, and $1 \leqslant k \leqslant n$;
  Update the relevance weights $v_{ik}$ by using (14);
  Update the partition matrix $\mathbf{U}^{(k)}$ by using (17);
  Update the centers by using (20);
  Update $\delta_i$ by using (15);
UNTIL (centers stabilize);
```

3.2. *Simultaneous clustering and attribute discrimination—version 2 (SCAD2)*

The SCAD2 algorithm omits the second term from its objective function by incorporating a discriminant exponent,

q, and minimizing

$$J_2(\mathbf{B},\mathbf{U},\mathbf{V};\mathbf{X}) = \sum_{i=1}^{C}\sum_{j=1}^{N}(u_{ij})^m \sum_{k=1}^{n}(v_{ik})^q d_{ijk}^2, \tag{21}$$

subject to (1), and (10).

To optimize J_2 with respect to $\mathbf{V}$, we use the Lagrange multiplier technique, and obtain

$$J_2(\mathbf{\Lambda},\mathbf{V}) = \sum_{i=1}^{C}\sum_{j=1}^{N}(u_{ij})^m \sum_{k=1}^{n}(v_{ik})^q d_{ijk}^2 - \sum_{i=1}^{C}\lambda_i\left(\sum_{k=1}^{n} v_{ik} - 1\right),$$

where $\mathbf{\Lambda} = [\lambda_1,\ldots,\lambda_c]^t$ is a vector of Lagrange multipliers corresponding to the C constraints in (10). Since the rows of $\mathbf{V}$ are independent of each other and d_{ijk}^2 is independent of $\mathbf{V}$, we can reduce the above optimization problem to the following C independent problems:

$$J_{2i}(\lambda_i,\mathbf{V}_i) = \sum_{j=1}^{N}(u_{ij})^m \sum_{k=1}^{n}(v_{ik})^q d_{ijk}^2 - \lambda_i\left(\sum_{k=1}^{n} v_{ik} - 1\right) \quad \text{for } i = 1,\ldots,C,$$

where $\mathbf{V}_i$ is the ith row of $\mathbf{V}$. By setting the gradient of J_{2i} to zero, we obtain

$$\frac{\partial J_{2i}(\lambda_i,\mathbf{V}_i)}{\partial \lambda_i} = \left(\sum_{k=1}^{n} v_{ik} - 1\right) = 0, \tag{22}$$

and

$$\frac{\partial J_i(\lambda_i,\mathbf{V}_i)}{\partial v_{it}} = q(v_{it})^{(q-1)}\sum_{j=1}^{N}(u_{ij})^m d_{ijt}^2 - \lambda_i = 0. \tag{23}$$

Eq. (23) yields

$$v_{it} = \left[\frac{\lambda_i}{q\sum_{j=1}^{N}(u_{ij})^m d_{ijt}^2}\right]^{1/(q-1)}. \tag{24}$$

Substituting (24) back into (22), we obtain

$$\sum_{k=1}^{n} v_{ik} = [\lambda_i/q]^{1/(q-1)}\sum_{k=1}^{n}\left[\frac{1}{\sum_{j=1}^{N}(u_{ij})^m d_{ijk}^2}\right]^{1/(q-1)} = 1.$$

Thus,

$$[\lambda_i/q]^{1/(q-1)} = \frac{1}{\sum_{k=1}^{n}[1/\sum_{j=1}^{N}(u_{ij})^m d_{ijk}^2)]^{1/(q-1)}}.$$

Substituting this expression back in (24), we obtain

$$v_{it} = \frac{[1/\sum_{j=1}^{N}(u_{ij})^m d_{ijt}^2]^{1/(q-1)}}{\sum_{k=1}^{n}[1/\sum_{j=1}^{N}(u_{ij})^m d_{ijk}^2}]^{1/(q-1)}. \tag{25}$$

Simplifying (24) and using k to represent the dimension, we obtain

$$v_{ik} = \frac{1}{\sum_{t=1}^{n}(\tilde{D}_{ik}/\tilde{D}_{it})^{1/(q-1)}}, \tag{26}$$

where $\tilde{D}_{ik} = \sum_{j=1}^{N}(u_{ij})^m d_{ijk}^2$, can be interpreted as a measure of dispersion of the ith cluster along the kth dimension, and $\sum_{t=1}^{n}\tilde{D}_{it}$ can be viewed as the total dispersion of the ith cluster (taking all dimensions into account). Hence v_{ik} is inversely related to the ratio of the dispersion along the kth dimension to the total dispersion for the ith cluster. This means that the more compact the ith cluster is along the kth dimension (smaller $\tilde{D}_{ik}$), the higher will the relevance weight, v_{ik}, be for the kth feature.

Eq. (26) may be likened to the estimation and use of a covariance matrix in an inner-product norm-induced metric in various clustering algorithms [16,17]. In fact, if d_{ijk}^2 is defined to be the Euclidean distance between x_{jk} and c_{ik}, then $\tilde{D}_{ik}$ becomes a measure of the fuzzy variance of the ith cluster along the kth dimension, and SCAD2 becomes very similar to the Gustafson–Kessel (GK) algorithm [16] with a diagonal covariance matrix. However, the estimation of a covariance matrix relies on the assumption that the data has a multivariate Gaussian distribution. On the other hand, SCAD2 is free of any such assumptions when estimating the feature weights. This means that SCAD2 can incorporate more general dissimilarity measures.

The role of the attribute weight exponent, q, can be deduced from equation (26), and is subject to the following theorem:

Theorem 1.

$$\lim_{q\to 1^+} v_{ik} = \begin{cases} 1 & if\ \tilde{D}_{ik} = \min_{t=1}^{n}\tilde{D}_{it}, \\ 0 & otherwise. \end{cases}$$

Proof. (see Appendix).

Theorem 1 implies that as q approaches 1, v_{ik} tends to take binary values. This case is analogous to the winner-take-all situation where the feature along which the ith cluster is the most compact gets all the relevancy ($v_{ik} = 1$), while all other attributes get assigned zero relevance, and hence do not contribute to the distance or center computations. On the other hand, when q approaches infinity, it can easily be shown that $v_{ik} = 1/n$. This means that all attributes share the relevancy equally. This is equivalent to the situation where no feature selection/weighting takes place. As can be expected, for the case where q takes finite values in $(1,\infty)$, we obtain weights that provide a moderate level of feature discrimination. For this reason, we will refer to q as a "discrimination exponent".

To minimize J_2 with respect to $\mathbf{U}$, we follow similar steps and obtain

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} [\tilde{d}_{ij}^2/\tilde{d}_{kj}^2]^{1/(m-1)}}, \tag{27}$$

where

$$\tilde{d}_{ij}^2 = \sum_{k=1}^{n} (v_{ik})^q d_{ijk}^2. \tag{28}$$

Even though the discrimination exponent, q, is necessary in order to derive the update equation for the feature weights, v_{ik}, and is useful to control their degree of discrimination (see (26)), we have noticed that for the purpose of computing the fuzzy memberships, u_{ij}, it is preferable to set this exponent to one, and simply use:

$$\tilde{d}_{ij}^2 = \sum_{k=1}^{n} v_{ik} d_{ijk}^2. \tag{29}$$

That is, we compute the distance measure as in SCAD1.

To minimize J_2 with respect to the centers, we fix $\mathbf{U}$ and $\mathbf{V}$, and set the gradient to zero. We obtain

$$\frac{\partial J}{\partial c_{ik}} = -2 \sum_{j=1}^{N} (u_{ij})^m (v_{ik})^q (x_{jk} - c_{ik}) = 0. \tag{30}$$

Eq. (30) is similar to (18), and its solution yields the same equation to update the centers as in SCAD1.

The SCAD2 algorithm is summarized below.

```
Simultaneous Clustering and Attribute Discrimination:
  SCAD2
Fix the number of clusters $C$;
Fix the fuzzifier $m$, $m \in [1, \infty)$;
Fix the discrimination exponent $q$, $q \in [1, \infty)$;
Initialize the centers and the fuzzy partition matrix $\mathbf{U}$;
Initialize all the relevance weights to $1/n$;
REPEAT
  Compute $d_{ijk}^2$ for $1 \leqslant i \leqslant C$, $1 \leqslant j \leqslant N$, and $1 \leqslant k \leqslant n$;
  Update the relevance weights matrix $\mathbf{V}$ by using (26);
  Compute $\tilde{d}_{ij}^2$ by using (29);
  Update the partition matrix $\mathbf{U}$ by using (27);
  Update the centers by using (20);
UNTIL (centers stabilize);
```

Table 1
Two two-dimensional Gaussian clusters

Cluster No. 1		Cluster No. 2	
x_1	x_2	x_1	x_2
−0.33	1.11	4.66	6.11
−2.02	−0.73	2.97	4.26
−0.33	0.72	4.66	5.72
−0.25	0.04	4.74	5.04
−1.08	−0.37	3.91	4.62
0.15	−0.36	5.15	4.63
−1.22	0.11	3.77	5.11
1.80	1.43	6.80	6.43
−1.48	−0.70	3.51	4.29
−0.87	1.02	4.12	6.02
−0.21	−0.45	4.78	4.54
−0.28	1.06	4.71	6.06
0.45	0.16	5.45	5.16
−2.29	1.98	2.70	6.98
0.84	−0.68	5.84	4.31
1.49	1.61	6.49	6.61
−0.23	0.31	4.76	5.31
−0.46	−0.82	4.53	4.17
−1.58	−1.09	3.41	3.90
0.72	1.27	5.72	6.27

Table 2
Results of SCAD1 on the data set in Table 1

	Cluster No. 1		Cluster No. 2	
Features	x_1	x_2	x_1	x_2
Centers	−0.4	0.24	4.65	5.27
Relevance weights	0.49	0.51	0.48	0.52

Table 3
Results of SCAD2 on the data set in Table 1

	Cluster 1		Cluster 2	
Features	x_1	x_2	x_1	x_2
Centers	−0.37	0.27	4.64	5.28
Relevance weights	0.43	0.57	0.43	0.57

We now illustrate the performance of SCAD1 and SCAD2 on a simple data set. Table 1 contains the coordinates of 20 points of two synthetic Gaussian clusters with $(\mu_1, \sum_1) = ([0,0]^{\mathrm{T}}, \mathbf{I}_2)$ and $(\mu_2, \sum_2) = ([5,5]^{\mathrm{T}}, \mathbf{I}_2)$. In this example, m, K (for SCAD1), and q (for SCAD2) were set to 2.0, and the centers and the fuzzy partition matrix were initialized by running the FCM algorithm for two iterations. SCAD1 converged after four iterations, and SCAD2 converged after five iterations giving the results displayed in Tables 2 and 3, respectively. Since both features are relevant, SCAD1 and SCAD2 assigned high weights for both features, and the estimated center coordinates are close to those of the actual centers. To demonstrate the ability of the proposed algorithms to cluster and identify relevant features, we increase the number of features to four by adding two irrelevant features to each cluster which are highlighted in Table 4. The first two features of the first cluster are uniformly distributed in the intervals $[0, 20]$ and $[0, 10]$, respectively. Features two and four of the second cluster are uniformly distributed in the intervals $[0, 10]$ and $[0, 5]$, respectively. A traditional feature selection algorithm can only discriminate against the second feature since it is irrelevant for both clusters. Clustering the remaining three features will not provide a compact

Table 4
Two four-dimensional clusters

Cluster No. 1				Cluster No. 2			
x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4
19.00	2.09	−0.33	1.11	4.66	2.13	6.11	0.28
4.62	3.79	−2.02	−0.73	2.97	6.43	4.26	1.76
12.13	7.83	−0.33	0.72	4.66	3.20	5.72	4.06
9.71	6.80	−0.25	0.04	4.74	9.60	5.04	0.04
17.82	4.61	−1.08	−0.37	3.91	7.26	4.62	0.69
15.24	5.67	0.15	−0.36	5.15	4.11	4.63	1.01
9.12	7.94	−1.22	0.11	3.77	7.44	5.11	0.99
0.37	0.59	1.80	1.43	6.80	2.67	6.43	3.01
16.42	6.02	−1.48	−0.70	3.51	4.39	4.29	1.36
8.89	0.50	−0.87	1.02	4.12	9.33	6.02	0.99
12.30	4.15	−0.21	−0.45	4.78	6.83	4.54	0.07
15.83	3.05	−0.28	1.06	4.71	2.12	6.06	3.73
18.43	8.74	0.45	0.16	5.45	8.39	5.16	2.22
14.76	0.15	−2.29	1.98	2.74	6.28	6.98	4.65
3.52	4.98	0.84	−0.68	5.84	1.33	4.31	2.33
8.11	7.67	1.49	1.61	6.49	2.07	6.61	2.09
18.70	9.70	−0.23	0.31	4.76	6.07	5.31	4.23
18.33	9.90	−0.46	−0.82	4.53	6.29	4.17	2.62
8.20	7.88	−1.58	−1.09	3.41	3.70	3.90	1.01
17.87	4.38	0.72	1.27	5.72	5.75	6.27	3.36

Table 5
Results of SCAD1 on the data set in Table 4

	Cluster No. 1				Cluster No. 2			
Features	x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4
Centers	13.06	5.56	−0.32	0.22	4.67	5.17	5.19	2.08
Relevance weights	0.00	0.23	0.38	0.40	0.28	0.16	0.29	0.27

Table 6
Results of SCAD2 on the data set in Table 4

	Cluster No. 1				Cluster No. 2			
Features	x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4
Centers	12.72	5.39	−0.40	0.26	4.62	5.26	5.26	2.03
Relevance weights	0.02	0.05	0.40	0.53	0.32	0.06	0.42	0.20

description of each cluster. SCAD1 converged after 10 iterations, and SCAD2 converged after five iterations, and their results are displayed in Tables 5 and 6, respectively. The first feature of the first cluster is correctly identified as irrelevant by both algorithms. The second feature of the first cluster is identified as irrelevant by SCAD2 (relevance $= 0.05$), but was assigned a larger weight by SCAD1 (relevance $= 0.23$) because it has a relatively smaller dynamic range. Feature four of the second cluster was not identified as totally irrelevant by both algorithms. This is because it has a dynamic range which is close to that of the actual features, and therefore this feature will be treated as almost equally important. Notice, however, that this feature was judged by SCAD2 to be more irrelevant compared to the remaining features than SCAD1.

We have used SCAD1 and SCAD2 to cluster several other data sets, and we have observed that they have similar behavior. Both algorithms succeed in identifying the relevant and irrelevant features, and assign similar weights. In the rest of this paper, we will extend only SCAD2 to the case of an unknown number of clusters, and illustrate its performance.

4. Simultaneous clustering and attribute discrimination: unknown number of clusters

The objective functions in (5) and (21) complement each other, and can easily be combined into one objective function. The algorithm that minimizes the resulting objective function would inherit the advantages of the CA and the SCAD2 algorithms. This algorithm, called SCAD2-CA, minimizes the following objective function:

$$J_3(\mathbf{B},\mathbf{U},\mathbf{V};\mathbf{X}) = \sum_{i=1}^{C}\sum_{j=1}^{N}(u_{ij})^2\sum_{k=1}^{n}(v_{ik})^q d_{ijk}^2 - \alpha\sum_{i=1}^{C}\left[\sum_{j=1}^{N}u_{ij}\right]^2, \qquad (31)$$

subject to (10), and $\sum_{i=1}^{C} u_{ij}=1, \forall j$. Since the second term in (31) does not depend on v_{ik}, minimization of (31) with respect to $\mathbf{V}$ yields the same update equation for v_{ik} as in SCAD2 (see Eq. (26)).

To minimize (31) with respect to $\mathbf{U}$, we rewrite it as

$$J(\mathbf{B},\mathbf{U},\mathbf{V};\mathbf{X}) = \sum_{i=1}^{C}\sum_{j=1}^{N}(u_{ij})^2\tilde{d}_{ij}^2 - \alpha\sum_{i=1}^{C}\left[\sum_{j=1}^{N}u_{ij}\right]^2,$$

where $\tilde{d}_{ij}^2$ is given by (28). Since $\tilde{d}_{ij}^2$ does not depend directly on u_{ij}, it can be shown that the update equation for u_{ij} is the same as the one for the CA algorithm if d_{ij}^2 is replaced with $\tilde{d}_{ij}^2$ in Eq. (6). Similarly, the agglomeration constant, α, can still be updated as in the CA algorithm using Eq. (7) and $\tilde{d}_{ij}^2$ instead of d_{ij}^2.

5. Application1: color image segmentation

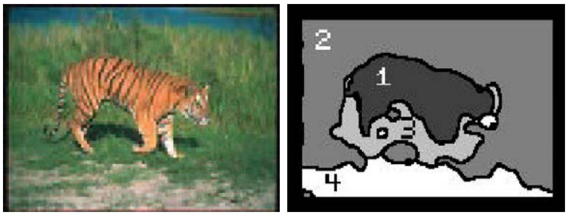
We illustrate the ability of SCAD2 to perform clustering and learn cluster-dependent feature weighting by using it to segment real color images. We start by mapping each pixel in the original image to an 8-dimensional feature vector consisting of color, texture, and position features. These are the same features used by Carson et al. in their Blobword content-based image retrieval system [18]. In the following, we will first give a brief description of the features, then show the results of using SCAD2 to segment several images.

5.1. Feature data extraction

Texture features. First, the image $I(x,y)$ is convolved with Gaussian smoothing kernels $G_\sigma(x,y)$ of several scales, σ, as follows:

$$M_\sigma(x,y) = G_\sigma(x,y) * (\nabla I(x,y))(\nabla I(x,y))^{\mathrm{t}}.$$

Then, the following three features are computed at each pixel location [18].



(a) (b)

Fig. 2. (a) Original color image, (b) results of SCAD2.

(1) *Polarity*: $p=|E_+ - E_-|/(E_+ + E_-)$, where E_+ and E_- represent the number of gradient vectors in the window $G_\sigma(x,y)$ that are on the positive and negative sides of the dominant orientation, respectively. For each pixel, an optimal scale value is selected such that it corresponds to the value where polarity stabilizes with respect to scale. Let p^* be the polarity at the selected scale.

(2) *Anisotropy*: $a = 1 - \lambda_2/\lambda_1$, where λ_1 and λ_2 are the eigenvalues of $M_\sigma(x,y)$ at the selected scale.

(3) *Normalized texture contrast*: $c = 2(\sqrt{\lambda_1 + \lambda_2})^3$.

Since the anisotropy and polarity are meaningless in regions of low contrast, these two features are scaled by the contrast value to yield the texture feature vector $[ac, p^*c, c]$.

Color features: The three color features are the $L^*a^*b^*$ coordinates of the color image computed after smoothing the image with a Gaussian kernel at the selected optimal scale. Note that smoothing is performed to avoid over-segmentation of regions due to local color variations.

Position features: The (x,y) coordinates of each pixel are used to reduce over-segmentation of some regions, and to obtain smoother regions. In [18], the authors reported that including position yielded better segmentation results than excluding it.

5.2. Feature data clustering

The first color image to be segmented is shown in Fig. 2. This is the same image used to illustrate the need for adaptive relevance weights in Section 1 (See Fig. 1). As discussed in the introduction, the FCM succeeds in delineating the compact cluster corresponding to the tiger, but fails to correctly delineate the different regions in the background. This is because the FCM is unable to assign cluster dependent feature relevance weights. When the number of clusters is also fixed to 4, SCAD2 succeeds in simultaneously segmenting the image into 4 meaningful clusters as shown in Fig. 2(b), and in determining appropriate relevance weights for each cluster, as displayed in Table 7. Since a total of eight features are used in this experiment, the resulting feature weights should be compared to $1/8 \approx 0.12$, for the purpose of judging their relevancy. As can be seen, the x-position feature is found to be irrelevant ($v = 0.014$) for the "dirt" cluster (cluster #4 in

Table 7
Feature relevance weights of the segmented objects in Fig. 2(b)

Clusters	Color features			Texture features			Position features	
	C_1	C_2	C_3	T_1	T_2	T_3	x	y
Cluster No. 1	0.194	0.107	0.207	0.029	0.208	0.039	0.074	0.141
Cluster No. 2	0.142	0.156	0.038	0.301	0.232	0.108	0.009	0.013
Cluster No. 3	0.117	0.073	0.417	0.131	0.046	0.084	0.042	0.091
Cluster No. 4	0.048	0.069	0.295	0.207	0.093	0.173	0.014	0.102

Fig. 2(b)), while the y-position ($v = 0.102$) is deemed relevant for this cluster. This is because the "dirt" region forms a narrow strip that extends over the entire width of the image. For the "grass" cluster (cluster #2 in Fig. 2(b)), both the x and y features are found to be irrelevant because this region is scattered around the entire image. Similarly, the color and texture features receive unequal weights depending on the cluster. This is because the regions are not all characterized by the same features. For instance, the normalized texture contrast for the tiger region (cluster #1) was determined to be practically irrelevant ($v = 0.039$) because this feature has a large dynamic range. In fact, the value of this feature varies from small (in the uniformly textured areas) to large. Also, for the tiger cluster, the first texture feature, *ac*, was found to be practically irrelevant ($v = 0.029$). This means that the anisotropy feature ($a = 1 - \lambda_2/\lambda_1$) is irrelevant. This is because the orientation of the stripes varies significantly in some parts of the tiger cluster. In other words, some locations of this cluster have a dominant orientation ($\lambda_1 \gg \lambda_2$), while others do not.

In the next experiment, we compare the performance of CA and SCAD2-CA to illustrate the importance of the simultaneous clustering and feature weighting mechanism when the number of clusters is unknown. The six color images in Fig. 3(a) are to be segmented into an unknown number of clusters. Both CA and SCAD2-CA were initialized by first dividing the image into a grid of 16 subimages. Then, the average of each feature within the ith subimage is computed to yield the corresponding initial feature value of the ith prototype. For the agglomeration coefficient, we set $(\eta_0, \tau, k_0) = (1.0, 10, 20)$. Fig. 3(b), displays the segmentation results using CA, and Fig. 3(c) displays the segmentation results using SCAD2-CA. The identified regions (i.e. clusters) are enumerated in Fig. 3(c), and their feature relevance weights are displayed in Fig. 3(d). As can be seen, the feature relevance weights differ not only from one feature to another, but also from one cluster to another.

As in the previous experiment, we notice that the SCAD2-CA performs better than CA, and it is able to yield meaningful clusters corresponding to both the objects in the foreground and the different areas of the background. This is because unlike CA, SCAD2-CA is capable of assigning distinct feature weights to different clusters. For example, the position features x and y are useful to delineate clusters corresponding to compact regions such as the animals in some pictures. Hence, both algorithms are able to delineate these clusters. However, the x- position feature is irrelevant for most of the areas in the background such as the sky or the grass regions. Without an unsupervised feature weighting mechanism, the CA is unable to recognize this selective irrelevance. The same phenomenon can also be observed if an object in the foreground is too large. For example, the plane image in the last row of Fig. 3(b) gets over-segmented by CA into two clusters. In contrast, SCAD2-CA was capable of segmenting all the foreground and background regions in all the images in Fig. 3 into an appropriate number of clusters. These previous examples clearly demonstrate the effectiveness of SCAD2-CA in color image segmentation.

5.3. Comparison with other clustering methods

The segmentation results obtained using the FCM and K-means algorithms for all the images shown in Fig. 3(a) are similar to those shown in Fig. 1(b) and (c). In general, we observed an over-fragmentation effect when the position features are not used. On the other hand, when position features are taken into account, the clusters become more compact. Hence, fragmentation is reduced. However, many of these clusters do not correspond to homogeneous regions. We note in particular, that the problems observed with FCM and K-means are a result of giving equal relevance to all features, and are not due to any sub-optimal convergence of these algorithms. This leads us to conclude that other clustering algorithms that are unable to distinguish between less and more relevant features will yield results showing similar deficiencies with regard to region segmentation.

5.4. Impact on content-based image retrieval systems

Color image segmentation is an important step of many Content-Based Image Retrieval (CBIR) systems. When used for this task, SCAD2 and SCAD2-CA offer the following advantages.

Good segmentation: As our results showed, the features are generally not equally relevant for all objects/scenes in an image. Hence, learning cluster-dependent relevance weights

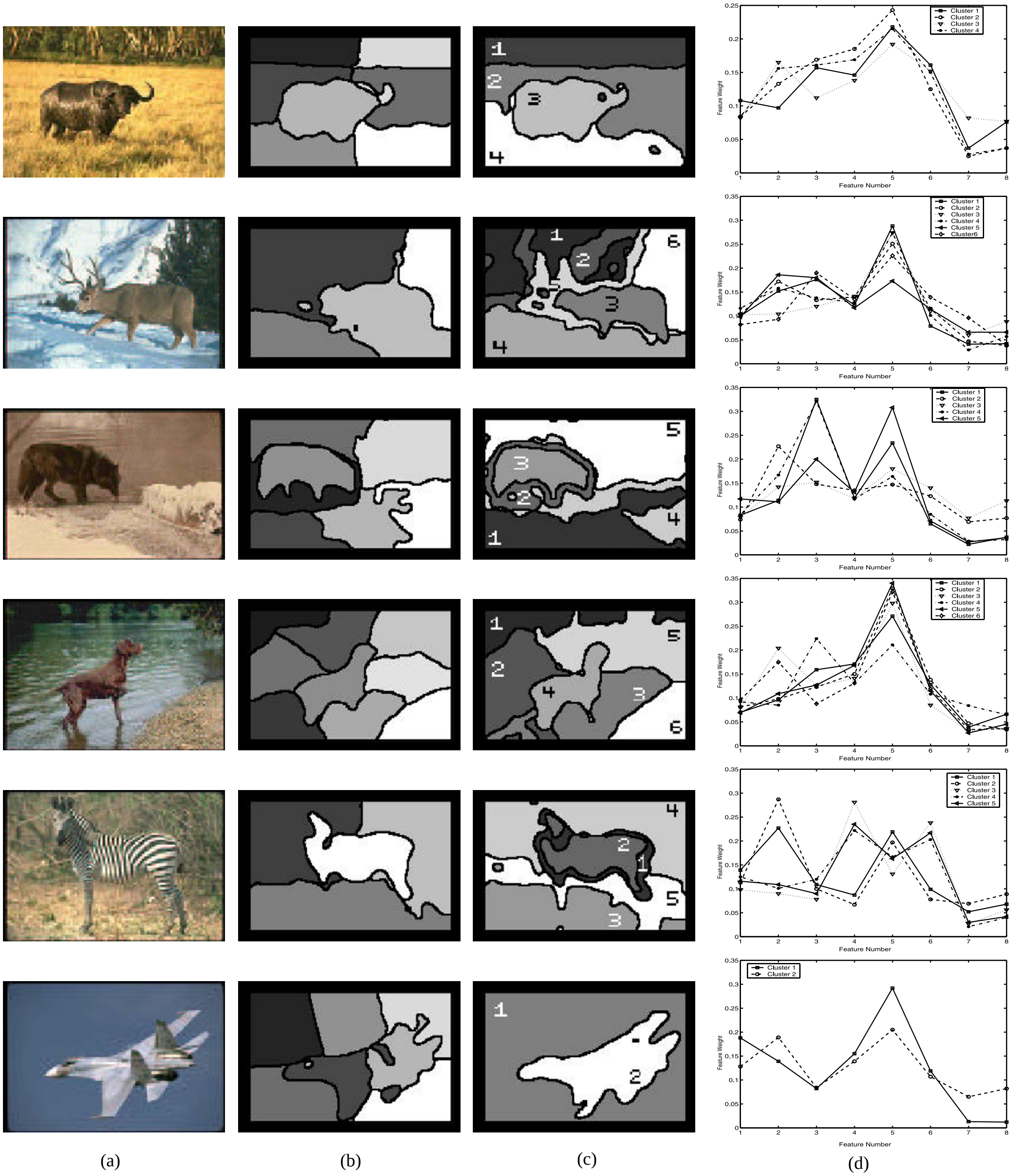


Fig. 3. Image segmentation with CA and SCAD2-CA: (a) Original color images, (b) CA segmentation results, (c) SCAD2-CA segmentation results, and (d) Feature relevance weights for the regions labeled in (c).

with SCAD2 will yield superior segmentation results. This translates into higher performance both in the indexing and retrieval phases of the CBIR system.

Compact and meaningful representation: With SCAD2, each segmented region can automatically be represented by a different set of relevance weights. Therefore, there is no need for any user intervention to specify the feature relevance weights as in the case of most CBIR systems. This means that the user does not need to be concerned about the internal representation of the color image. This is particularly helpful for the "general" (non-specialized) user.

More efficient retrieval: For the retrieval step, the segmented objects in the query image must be compared to the pre-segmented objects which are stored in the database. If all features are used indiscriminately, then the irrelevant features which can have unpredictable values, may have a contribution to the distance measure that is so large that even those images which are visually similar to the query image will not be retrieved. The cluster-dependent relevance weights provide an implicit mechanism for local similarity measures that take color, texture, and spatial context into account when comparing regions.

6. Application 2: supervised classification

In this application, we use SCAD2-CA for supervised classification. In this problem, the class labels for a set of (training) data are given, and the goal is to find the inter-class boundaries. Subsequently, any new (testing) data point can be classified into one of the known classes based on its position relative to the class boundaries. We will use SCAD2-CA for the construction of the class boundaries by building a Multiple Prototype Classifier (MPC). We call this classifier SCAD2-MPC.

The MPC is a variation of the K-nearest neighbor (K-NN) approach which offers the advantage of reduced computational and storage costs. Instead of retaining all training instances in primary memory, only a few typical instances are stored and used to determine the class of a new instance. A smaller number of prototypes can achieve comparable or superior predictive accuracy because noisy instances will not be used as prototypes, and limiting the number of prototypes may avoid overfitting the training data. The number of prototypes that is required depends on the complexity of the distribution of each class. Most existing MPC algorithms fix the number of prototypes for each class, and treat all features as being equally important. Unfortunately, fixing the number of prototypes limits the complexity of the model of the problem being addressed. Moreover, equal weights make the classifier sensitive to the distance function. Existing feature weighting algorithms can only assign one set of feature weights for each class, and ignore the fact that different local regions (clusters) of complex classes are best characterized by different subsets of feature weights. In the following, we show that classification accuracy can be improved if we use SCAD2-CA to learn the optimal number of prototypes, and the optimal set of feature weights for each prototype.

Four standard data sets[1] were used to evaluate the performance of the SCAD2-MPC classifier. These are the Iris data set, the Wisconsin Breast Cancer data set, the Pima Indians Diabetes data set, and the Heart Disease data set. The Iris data set has four features and three classes, namely, "setosa", "versicolor", and "virginica". Each class has 50 samples. The Wisconsin Breast Cancer data set has nine features and two classes, namely, benign and malignant. These two classes have 444 and 239 samples, respectively. The Pima Indian Diabetes data set has eight features and two classes. The first class, negative test, has 500 samples and the second class, positive test, has 268 samples. The Heart Disease data set has 13 features and two classes. The first class, absence of heart disease, has 150 samples, and the second class, presence of heart disease, has 120 samples. Since these data sets are not divided into training and testing sets, we use a 25% Jackknifing procedure. In other words, the training data set is created by leaving out every fourth data point. These left out data points are later used for testing. This process is repeated 4 times, each time leaving out a different set of points. The reported results are averages of the four jackknifing experiments.

During training, the CA and SCAD2-CA algorithms used the same agglomeration values for all the data sets, namely, $(\eta_0, \tau, k_0) = (1.0, 10, 20)$. Since the data sets have different cardinalities and dimensions, we use different values of $C_{\max}$ (initial number of clusters) for the different classes of the data sets. This value is set automatically, depending on the cardinality (N) and the dimensionality (n) of the class being clustered, to $C_{\max} = N/2n$.

Table 8 shows the classification rates obtained using the CA-based classifier (CA-MPC), and the SCAD2-CA based classifier (SCAD2-MPC) for the four data sets used in this experiment. It can be seen that SCAD2-MPC results in better rates of correct classification for all data sets. Most importantly, the improvement is particularly significant when the features are assigned different weights depending on the clusters. This confirms the importance of the unsupervised feature weighting mechanism of SCAD2. For instance, the improvement in the correct classification rate of the "wisconsin breast cancer" data set is small, as expected, because the feature weights do not vary widely as can be seen in Table 9. On the other hand, a significant improvement can be observed for the "heart disease" data set because the feature weights determined by SCAD2-CA vary widely as shown in Table 10.

It should be noted that in addition to computing cluster-dependent feature weights, SCAD2-CA determines the number of sub-classes or prototypes within each class in an unsupervised manner. For instance,

[1] Theses data sets are available via anonymous ftp: ftp.ics.uci.edu.

Table 8
Average percentage of correct classification of 2 classifiers

Data set	CA-MPC		SCAD2-MPC		MLP	
	Train	Test	Train	Test	Train	Test
Iris	95.11	92.67	95.11	94.00	95.18	95.14
Wisconsin Breast cancer	95.90	95.46	96.83	96.78	96.93	95.74
Pima Indians Diabetes	74.04	71.74	76.30	74.87	75.87	74.48
Heart disease	81.85	81.48	85.19	85.19	85.84	82.02

Table 9
Feature relevance weights of the "Wisconsin Breast Cancer" data set

Jacknife	Class	Prototype	Attribute weights								
			1	2	3	4	5	6	7	8	9
1	1	1	0.05	0.14	0.11	0.10	0.12	0.08	0.09	0.11	0.21
	2	1	0.12	0.13	0.13	0.09	0.12	0.09	0.14	0.09	0.09
		2	0.11	0.11	0.12	0.08	0.12	0.09	0.13	0.08	0.16
2	1	1	0.05	0.11	0.11	0.13	0.11	0.07	0.09	0.11	0.21
	2	1	0.13	0.11	0.12	0.09	0.13	0.09	0.14	0.08	0.12
3	1	1	0.05	0.11	0.09	0.10	0.12	0.08	0.09	0.10	0.25
	2	1	0.11	0.11	0.12	0.08	0.13	0.08	0.13	0.08	0.15
		2	0.14	0.13	0.12	0.09	0.13	0.08	0.14	0.09	0.09
4	1	1	0.05	0.11	0.10	0.10	0.10	0.08	0.08	0.09	0.29
	2	1	0.11	0.11	0.12	0.08	0.12	0.09	0.13	0.08	0.15
		2	0.12	0.12	0.13	0.09	0.13	0.09	0.14	0.09	0.09

Table 10
Feature relevance weights of the "heart disease" data set

Jackknife	Class	Prototype	Attribute weights												
			1	2	3	4	5	6	7	8	9	10	11	12	13
1	1	1	0.09	0.03	0.05	0.12	0.13	0.04	0.03	0.13	0.04	0.19	0.05	0.07	0.03
	2	1	0.12	0.04	0.06	0.11	0.21	0.04	0.03	0.12	0.03	0.09	0.06	0.05	0.04
2	1	1	0.09	0.03	0.05	0.12	0.17	0.04	0.03	0.13	0.04	0.15	0.05	0.08	0.03
	2	1	0.12	0.04	0.07	0.11	0.22	0.04	0.03	0.11	0.03	0.10	0.06	0.05	0.03
3	1	1	0.09	0.03	0.05	0.13	0.16	0.04	0.03	0.13	0.04	0.15	0.05	0.08	0.04
	2	1	0.12	0.04	0.07	0.12	0.21	0.04	0.03	0.12	0.03	0.09	0.06	0.05	0.03
4	1	1	0.09	0.02	0.05	0.12	0.18	0.04	0.03	0.13	0.04	0.13	0.05	0.09	0.04
	2	1	0.12	0.04	0.07	0.11	0.22	0.06	0.03	0.11	0.03	0.09	0.06	0.05	0.03

Table 9 shows that for the "wisconsin breast cancer" data set, class 1 is probably compact since SCAD2-CA finds a single prototype in all four jackknives. On the other hand, class 2 may have a more complex distribution since it gets split into two clusters in all but one jackknife.

6.1. Comparison with neural network based classification

The nearest-prototype classification results obtained with CA-MPC and SCAD2-MPC were compared with those obtained using a multilayer perceptron (MLP) classifier. For all data sets, we used a neural network architecture consisting of a single hidden layer with 4 nodes. Each network was trained with standard backpropagation learning using gradient descent (available in the *Matlab Neural Network Toolbox*). Each data set was divided into training and testing sets using a 25% Jackknifing procedure as in the case of CA-MPC and SCAD2-MPC. Training was terminated when the total error fell below $1.0\mathrm{e}^{-3}$ or when the number of epochs exceeded 3000.

The classification results, listed in last two columns of Table 8, show that the SCAD2-MPC classification approach is slightly better than the standard MLP classification. The MLP classification results may be improved by changing the network architecture (e.g. more hidden layers and/or more hidden nodes), the network learning algorithm, using a subset of the data for cross validation, ..., etc. However, our goal here is to illustrate that the classification results of SCAD2-MPC are comparable to well-known techniques while providing additional advantages. For instance, the SCAD2-MPC approach is more suitable for applications where there are constraints on training time, and where a more comprehensive representation and interpretation (Gaussian components and feature relevance weights) of the data are desirable. In addition, the proposed clustering technique can be fused with other classifiers to take advantage of the completely different approaches to classification and to improve the classification.

7. Conclusion

In this paper, we presented a new approach to perform clustering and feature weighting simultaneously. When used as part of an unsupervised learning system, SCAD1 and SCAD2 can categorize the unlabeled data while determining the best feature weights within each cluster. They can also enhance supervised learning by modeling complex classes by several clusters and determining the optimal feature weights within each one of those clusters in an unsupervised fashion. In addition to the above advantages, SCAD2-CA can determine the "optimal" number of clusters automatically. Our experimental results on both synthetic and real examples showed that SCAD2 and SCAD2-CA could successfully cluster data and determine cluster dependent feature weights simultaneously. Our results demonstrated that the proposed paradigms outperform similar clustering algorithms specifically because of their ability to determine cluster-dependent feature weights. Both SCAD1 and SCAD2 minimize one objective function for both the optimal prototype parameters and feature weights for each cluster. This optimization is done iteratively by dynamically updating the prototype parameters and the feature weights in each iteration. The optimization is done based on gradient descent, which makes the proposed clustering approach simple and fast.

The proposed approach to simultaneous clustering and attribute discrimination (especially SCAD2) may be likened to existing clustering algorithms that estimate and use a covariance matrix. However, unlike these algorithms, our approach does not assume that the data has a multivariate Gaussian distribution, and can incorporate a variety of distance measures to find clusters of arbitrary shapes.

Since the objective functions of SCAD1 and SCAD2 are based on that of the FCM, they inherit most of the advantages of FCM-type clustering algorithms, such as ease of computation, and simplicity. In the future, we plan to extend SCAD1 and SCAD2 to take into account possible correlations between features, as well as robustness to noise.

Appendix

Theorem 1.

$$\lim_{q\to 1^+} v_{ik} = \begin{cases} 1 & \text{if } \tilde{D}_{ik} = \min_{t=1}^{n} \tilde{D}_{it}, \\ 0 & \text{otherwise.} \end{cases}$$

Proof. The relevance weight of feature k in cluster i is defined as

$$v_{ik} = \frac{1}{\sum_{t=1}^{n} (\tilde{D}_{ik}/\tilde{D}_{it})^{1/(q-1)}}. \tag{A.1}$$

Since the exponent $1/(q-1) \to \infty$ as $q \to 1^+$, the individual terms in the denominator have the following tendency

$$\lim_{q\to 1^+} (\tilde{D}_{ik}/\tilde{D}_{it})^{1/(q-1)} = \begin{cases} \infty & \text{if } \tilde{D}_{ik} > \tilde{D}_{it}, \\ 0 & \text{if } \tilde{D}_{ik} < \tilde{D}_{it}, \\ 1 & \text{if } \tilde{D}_{ik} = \tilde{D}_{it}. \end{cases}$$

Therefore, two cases arise:

Case 1:

$$\exists t \neq k \mid \tilde{D}_{ik} > \tilde{D}_{it} \Rightarrow \tilde{D}_{ik} \neq \min_{t=1}^{n} \tilde{D}_{it}.$$

The denominator in (A.1) becomes infinite, and

$$\lim_{q\to 1^+} v_{ik} = 0.$$

Case 2:

$$\nexists t \neq k \mid \tilde{D}_{ik} > \tilde{D}_{it} \Rightarrow \tilde{D}_{ik} = \min_{t=1}^{n} \tilde{D}_{it}.$$

hence,

$$v_{ik} = \frac{1}{(\tilde{D}_{ik}/\tilde{D}_{it})^{1/(q-1)} + \sum_{t=1,t\neq k}^{n} (\tilde{D}_{ik}/\tilde{D}_{it})^{1/(q-1)}}$$

$$= \frac{1}{1 + \sum_{t=1,t\neq k}^{n} (\tilde{D}_{ik}/\tilde{D}_{it})^{1/(q-1)}}.$$

Since $\lim_{q\to 1^+}(\tilde{D}_{ik}/\tilde{D}_{it})^{1/(q-1)} = 0$ for all $t \neq k$,

$$\lim_{q\to 1^+} v_{ik} = 1.$$

Therefore, we conclude that

$$\lim_{q\to 1^+} v_{ik} = \begin{cases} 1 & \text{if } \tilde{D}_{ik} = \min_{t=1}^{n} \tilde{D}_{it}, \\ 0 & \text{otherwise.} \end{cases}$$

References

[1] H. Almuallim, T.G. Dietterich, Learning with many irrelevant features, in: Ninth National Conference on Artificial Intelligence, 1991, pp. 547–552.

[2] K. Kira, L.A. Rendell, The feature selection problem: traditional methods and a new algorithm, in: Tenth National Conference on Artificial Intelligence, 1992, pp. 129–134.

[3] L.A. Rendell, K. Kira, A practical approach to feature selection, in: International Conference on Machine Learning, 1992, pp. 249–256.

[4] D. Wettschereck, D.W. Aha, T. Mohri, A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms, Artif. Intell. Rev. 11 (1997) 273–314.

[5] P. Domingos, Context sensitive feature selection for lazy learners, Artif. Intell. Rev. 11 (1997) 227–253.

[6] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York, 1981.

[7] H. Frigui, R. Krishnapuram, Clustering by competitive agglomeration, Pattern Recognition 30 (7) (1997) 1223–1232.

[8] G. John, R. Kohavi, K. Pfleger, Irrelevant features and the subset selection problem, in: Eleventh International Machine Learning Conference, 1994, pp. 121–129.

[9] R. Kohavi, D. Sommerfield, Feature subset selection using the wrapper model: overfitting and dynamic search space topology, in: First International Conference on Knowledge Discovery and Data Mining, 1995, pp. 192–197.

[10] K. Fukunaga, Statistical Pattern Recognition, Academic Press, San Diego CA, 1990.

[11] D. Skalak, Prototype and feature selection by sampling and random mutation hill climbing algorithms, in: Eleventh International Machine Learning Conference (ICML-94), 1994, pp. 293–301.

[12] M.S. Lee, A.W. Moore, Efficient algorithms for minimizing cross validation error, in: W. Cohne, H. Hirsh (Eds.), Machine Learning: Proceedings of the Eleventh International Conference, 1994, pp. 190–198.

[13] I. Kononenko, Estimation attributes: analysis and extensions of relief, in: European Conference on Machine Learning, 1994, pp. 171–182.

[14] H. Frigui, R. Krishnapuram, A robust competitive clustering algorithm with applications in computer vision, IEEE Trans. Pattern Anal. Mach. Intell. 21 (5) (1999) 450–465.

[15] H.W. Kuhn, A.W. Tucker, Nonlinear programming, in: J. Neyman (Ed.), Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, 1950, pp. 481–492.

[16] E.E. Gustafson, W.C. Kessel, Fuzzy clustering with a fuzzy covariance matrix, in: IEEE CDC, San Diego, California, 1979, pp. 761–766.

[17] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. Ser. B 39 (1) (1977) 1–38.

[18] C. Carson, S. Belongie, H. Greenspan, J. Malik, Color- and texture-based image segmentation usin em and its application to image querying and classification, Submitted to: IEEE Trans. on Pattern Anal. Mach. Intell.

About the Author—HICHEM FRIGUI received the B.S. degree in Electrical and Computer Engineering in 1990 (with honors), the M.S. degree in Electrical Engineering in 1992, and the Ph.D. degree in Computer Engineering and Computer Science in 1997, all from the University of Missouri, Columbia. He joined the University of Memphis in 1998, where he is currently an assistant professor in the Electrical and Computer Engineering Department. His research interests include pattern recognition, data mining, application of fuzzy set theory, and scalable robust clustering techniques with applications to content-based image retrieval, and land mine detection. He is a member of IEEE, IEEE Computer Society, and ACM.

About the Author—OLFA NASRAOUI earned her Ph.D. in Computer Engineering and Computer Science in 1999 from the University of Missouri-Columbia. She is currently Assistant Professor of Electrical and Computer Engineering at the University of Memphis. Her research interests lie in the fields of Data Mining, Web Mining, and Soft Computing, particularly Evolutionary Computation. In 2002, she was awarded the National Science Foundation CAREER award. She is a member of IEEE, ACM, Society of Women Engineers (SWE), North American Fuzzy Information Processing Society (NAFIPS), and Web Intelligence Consortium (WIC).