A Method for Dynamic Clustering of Data

Arnaldo J. Abrantes† and Jorge S. Marques‡

†Electronic and Communication Engineering
Instituto Superior Eng. de Lisboa
R. Conselheiro Emídio Navarro, Lisboa, Portugal
aja@cedet.isel.pt

‡Instituto Superior Tecnico/Instituto de Sistemas e Robótica
Torre Norte, R. Rovisco Pais, P-1096, Lisboa, Portugal
jsm@isr.ist.utl.pt

Abstract

This paper describes a method for the segmentation of dynamic data. It extends well known algorithms developed in the context of static clustering (e.g., the c-means algorithm, Kohonen maps, elastic nets and fuzzy c-means). The work is based on an unified framework for constrained clustering recently proposed by the authors in [1]. This framework is extended by using a motion model for the clusters which includes global and local evolution of the data centroids. A noise model is also proposed to increase the robustness of the dynamic clustering algorithm with respect to outliers.

1 Introduction

Many clustering algorithms try to approximate a set of data vectors $X = (x_1, ..., x_n)$ by a smaller set of prototypes $Y = (y_1, ..., y_m)$. The main problem consists of computing the number of prototypes and their locations in order to obtain the best fit, according to a specific criterion.

This paper addresses dynamic clustering problems in which data is gathered during a time interval. A new data set is observed at each time instant, requiring a dynamic update of the classification results. The simplest idea to address this problem is to perform an instantaneous classification of the data measured at each instant of time, neglecting the previous data. The classic clustering algorithms, minimizing a fitness measure between the model and the data, could be used at each instant of time (e.g., c-means, LBG method [3, 4]). Unfortunately this approach leads to unacceptable results in most cases, i.e., classical methods cannot cope with dynamic clustering problems and they are not able to deal with outliers.

This paper describes a more promising approach, based on the minimization of an extended criterion which measures the model fitness, as before, but also accounts for the available information about the motion of the clusters and prototypes. In addition, a noise model is used to represent the outliers, increasing the robustness of the classifier.

BMVC 1998 doi:10.5244/C.12.16

The choice of a motion model associated with the data clusters is a key issue. In some problems, the motion of different clusters can be considered as independent (e.g., in the case of a video sequence of moving objects, with a cluster associated to each object). However, in other problems there is a coherent motion of all the clusters, according to a global model (e.g., a change of illumination in a static scene modifies the RGB components of each pixel, and the color centroids in a global way, e.g., by scaling the RGB components by appropriate factors). In this case, all centroids suffer the same transformation. Both mechanisms will be considered in the motion model adopted in this paper, in order to deal with a large class of problems in an unified way.

The methods developed in this paper are applied in the context of image and video analysis. However, they can be used in other contexts as well. Previous applications in clustering algorithms to dynamic scene analysis can be found in [5, 6, 7].

2 Problem Formulation

Let $X_t = (x_1(t), \cdots, x_n(t)), t = 1, 2, \cdots$ be a sequence of data points organized in $m(t)$ clusters. We wish to estimate a set of centroids $Y_t = (y_1(t), \cdots, y_m(t))$ to approximate the data observed at the instant t.

It will be assumed that the centroid motion is due to a global transformation plus local displacements, i.e.,

$$y_j(t) = T[y_j(t-1)] + d_j(t) \tag{1}$$

where T is a transformation which depends on a vector of unknown parameters, $\alpha_t \in \Re^p$; $d_j(t)$ is the local motion of the j-th centroid which models the changes which can not be explained by the global motion T. It should be stressed that α_t is the same for all centroids. In (1), it is assumed that

$$\sum_{j=1}^{m} d_j(t) = 0 \tag{2}$$

to force that transformation T accounts for translation effect, which otherwise could be erroneously interpreted as a deformation.

To estimate the centroid locations we will define an energy function

$$E(D, \alpha) = E_{data}(Y) + E_{motion}(D, \alpha) \tag{3}$$

where E_{data} is a data energy which measures the average distortion between data points and centroids and E_{motion} is a motion energy which measures the motion changes; $D = (d_1, \cdots, d_m)$ being the sequence of local deformations[1].

The optimal motion parameters (global and local) are obtained by the minimization of the energy function with respect to both variables at each instant of time, i.e.,

$$(\widehat{D}, \widehat{\alpha}) = \arg\min_{D,\alpha} E(D, \alpha)$$

constrained by (2).

[1] the index t was dropped for the sake of simplicity.

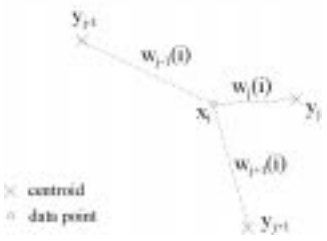

Figure 1: Fuzzy weights

3 Data Energy

Several criteria were proposed in the literature to measure the distortion between a data set X_t and a collection of centroids Y_t. All of them can be used as data energy functions in (3).

In some cases an explicitly link between the centroids and the observed data is considered, e.g., using m springs linking each centroid to the nearest data point [8]. In this paper, the associative process is not made explicitly but instead we adopt a fuzzy data dependent energy which pulls the centroids towards regions with highest data densities (see a related discussion in [9]). Some well known examples are the energies used in c-means, Kohonen maps, elastic nets and fuzzy c-means [10, 11, 12]. It was recently shown that all these energies are special cases of a fuzzy energy criterion defined by [1]

$$E_{data}(Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} w_j(i) \|x_i - y_j\|^2 \tag{4}$$

where $w_j(i), j = 1, \cdots, m$ are a set of weights which measure the influence of the i-th pattern on the j-th centroid (see Fig. 1). The weights $w_j(i)$ depend on the chosen method. In competitive learning algorithms, the weights $w_j(i)$ depend on the location of all the centroids.

In this paper, we shall adopt the fuzzy energy (4), choosing the appropriate weights if we wish the algorithm to behave like c-means, Kohonen maps, elastic nets or fuzzy c-means (see [1] for details).

4 Noise Model

The data energy defined in (4) is unable to deal with outliers. All data points are considered as valid data and attract the centroids with a scale factor which depend on $w_j(i)$. One way to overcome this diffficulty in static shape analysis is proposed in [2] by using a noise model. This method is also useful in dynamic scene analysis.

If we wish to deal with outliers (data points or clusters which are not to be followed) we must be able to segment the data, i.e., to separate valid data from outliers in a hard or in a fuzzy way.

One way to achieve this goal is by adding an additional model unit to represent the noise patterns (noise unit). When the data points are bidimensional, the noise unit can be interpreted as a plane in $\Re^3$ parallel to the data space. This interpretation can be extended to higher dimension data spaces, being the noise model a hyperplane in that case.

The distance of the data vectors to the hyperplane is constant, i.e., we can formally assume that

$$\|x_i - y_{m+1}\| = c$$

where the prototype y_{m+1} is the projection of x_i on the hyperplane. The noise hyperplane is used to represent data points which are far from centroids and which have a small link $w_j(i)$ with them.

The energy function used in this case is a simple extension of (4) including the additional unit

$$E_{data}(Y) = \sum_{i=1}^{n} \sum_{j=1}^{m+1} w_j(i) \|x_i - y_j\|^2 \tag{5}$$

(the weights $w_j(i)$ are computed by the same expressions, but with an additional competing unit).

This is an elegant way to solve the data validation/association problem commonly found in tracking applications (Bar-Shalom & Fortmann): the noise plane enables us to validate the data and to associate it in a fuzzy way, using a criterion which resembles the PDAF (Probabilistic Data Association Filter) algorithm proposed in [13].

5 Motion Model

It is assumed that centroids evolve dynamically according to

$$y_j(t) = T[y_j(t-1)] + d_j(t) \tag{6}$$

where T is a global transformation parametrized by a set of unknown variables α_t and $d_j(t)$ is a local motion or deformation. Two hypothesis will be made: i) the deformations $d_j(t)$ should be as small as possible and ii) the transformation parameters change slowly in time. These hypotheses suggest a motion energy defined by

$$\begin{array}{c} E_{motion}(\alpha, D) = \gamma_1 \sum_{j=1}^{m} \|d_j(t)\|^2 \\ +\gamma_2 \sum_{j=1}^{p} \|\alpha_j(t) - \alpha_j(t-1)\|^2 \end{array}$$

The parametric transformation T depends on the problem under consideration. Two examples are discussed bellow.

- Object Tracking

  Let $x_1(t), \cdots, x_n(t)$ denote the location of the edge points detected in an image sequence with a moving object. We wish to approximate the object boundary by a sequence of prototypes (image points) $y_1(t), \cdots, y_m(t)$ and track their evolution in time.

The problem can be formulated as before. An appropriate global motion model is the affine transform

$$T[y_j] = Ay_j + b \tag{7}$$

which includes rotation, translation, scaling and shearing of the object shape in every instant of time. The unknown parameters are: $\alpha = (A, b)$.

- Color Processing

  Let $x_1(t), \cdots, x_n(t)$ be vectors containing the RGB components of n pixels extracted from the t-th frame of a video sequence. Clusters in color space may correspond to different objects in the scene or to different parts of the same object. Changes in illumination modify the RGB components of each pixel. In a first approximation this effect can be modeled by a scaling operation

$$T[y_j] = \alpha y_j \tag{8}$$

  where α is a scale factor.

6 Optimization

The minimization of (3) constrained by (2) can be achieved in a number of ways. It is addressed in this paper by the optimization of an extended energy

$$\tilde{E}(D, \alpha) = E(D, \alpha) + \lambda \left\| \sum_{j=1}^{m} d_j \right\|^2 \tag{9}$$

where λ is a weight which tends to infinity during the minimization process (in the limit, D will have a zero mean). This approach is simpler than the minimization of a Lagrangian function for the same problem since it is not easy to obtain closed form expressions of the Lagrange multipliers.

The optimization often requires the computation of partial derivatives. To compute the partial derivatives of the data energy (5) we will use an auxiliary result. Given a fuzzy energy (5), it is shown in [1] that

$$\frac{\partial E_{data}(Y)}{\partial y_j} = \mu_j (y_j - \xi_j) \tag{10}$$

where μ_j is the mass of the data points associated with y_j (weighted by appropriate coefficients) and ξ_j is the corresponding mass center. The computation of μ_j, ξ_j depends on the weights $w_j(i)$, defined by the user and also on the location of all centroids.

Using (10), the partial derivatives of the extended energy (9) are easily obtained

$$\frac{\partial \tilde{E}}{\partial d_k} = \mu_k \left(y_k(t) - \xi_k \right) + \gamma_1 d_k + \lambda \sum_{j=1}^{m} d_j \tag{11}$$

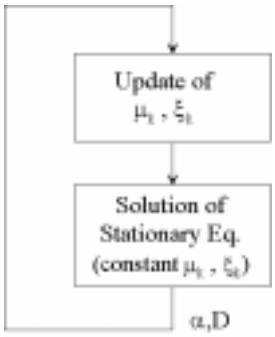

Figure 2: Two step optimization

$$\frac{\partial \tilde{E}}{\partial \alpha_j} = \sum_{k=1}^{m} \mu_k \left. \frac{\partial T}{\partial \alpha_j} \right|_{y_k} . (y_k(t) - \xi_k) + \gamma_2 \Delta\alpha_j \tag{12}$$

with $\Delta\alpha = (\alpha_j(t) - \alpha_j(t-1))$.

The motion parameters (α, D) can either be estimated by the minimization of (9) (e.g., using a gradient descent algorithm) or by solving a set of necessary conditions

$$\frac{\partial \tilde{E}}{\partial d_k} = 0 \qquad k = 1, \cdots, m \tag{13}$$

$$\frac{\partial \tilde{E}}{\partial \alpha_j} = 0 \qquad j = 1, \cdots, p \tag{14}$$

Several methods exist to solve systems of nonlinear equations (e.g., the Newton Raphson algorithm). A two step recursion is used instead. Each iteration consists of two steps (see Fig. 2): i) the update of the masses and mass centers μ_j, ξ_j, using the most recent estimates of the centroids Y and ii) the solution of (13,14) assuming that all μ_j, ξ_j are constant i.e., assuming that they do not depend on Y.

The first step is trivial. All we have to do is to use the expressions of μ_j, ξ_j. The second step depends on the motion model. If we choose one of the models described in section 2, the partial derivatives (11,12) become linear if μ_j, ξ_j are kept constant. Therefore, the second step is equivalent to the solution of $mN + p$ linear equations, where N is the dimension of the data space.

$$M \begin{bmatrix} D \\ \alpha \end{bmatrix} = r \tag{15}$$

(see the expressions for M and r in Appendix).

The motion parameters (D, α) obtained from (15) are then used to update Y according to (6) and to update μ_j, ξ_j. The convergence of the recursive algorithm is not guaranteed. In practice, a small number of iterations (typically less than 10) is enough to achieve stationary estimates for centroids.

The two step optimization algorithm is an example of a larger class of optimization methods based on a set of auxiliary variables studied by Cohen in [14]. The two step

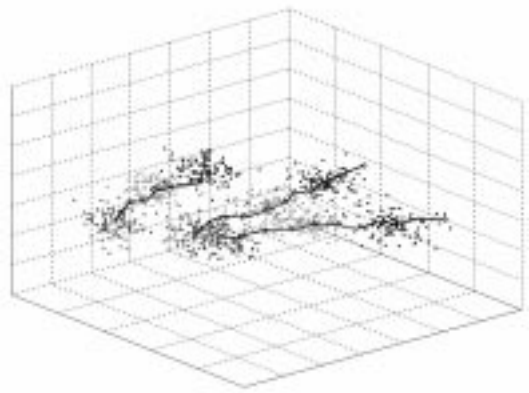

Figure 3: Dynamic clustering of synthetic data with a global motion model

approach is also similar to the EM method proposed by Dempster et al. in the context of statistic inference based on Fisher ML principle [15].

7 Results

To illustrate the performance of the method described in the paper, three examples are presented bellow (one with synthetic data and two with real data).

In the first example, the data points are organized in three clusters. The true centroids were generated with the motion model of section 5, using a scale factor α as the only unknown parameter (see (8); $\alpha = \frac{t+7}{t+6}$ was used to synthesize the centroid trajectories [2]). Three Gaussian clusters were generated at each instant of time, centered at the centroids. Figure 3 shows the trajectory of the true centroids, the estimates obtained by the method described in the paper (small circles) and the data sets at three instants of time (t=1,5,9). It is observed a good agreement between the true trajectories of the centroids and the estimates derived from the observed data. It should be stressed that no matching was performed between the data points obtained at different instants of time. The experiment was performed using the weights of the c-mean algorithm.

Figures 4,5 show examples based on image sequences: a traffic sequence and an ultrasound sequence. Data points are the edges detected in the images by the Sobel edge detector (noise reduction was performed in the ultrasound images). Thirty centroids were used to approximate the object boundary. The unknown centroids are initialized near the object boundary in the first image and they try to track the object assuming an affine motion model, as described in section 5. Figure 4,5 show the tracking results obtained by the application of (15). Good results were obtained with real data, even when there is a partial occlusion of the objects or a large number of outliers (edges belonging to other objects or background). These results were obtained with the fuzzy c-means weights $w_j(i)$.

[2] the evolution law adopted for α keeps the average cluster velocity constant; the instantaneous velocity of each centroid is a random variable due to the influence of d_j

Figure 4: Dynamic clustering of a traffic sequence (affine model)

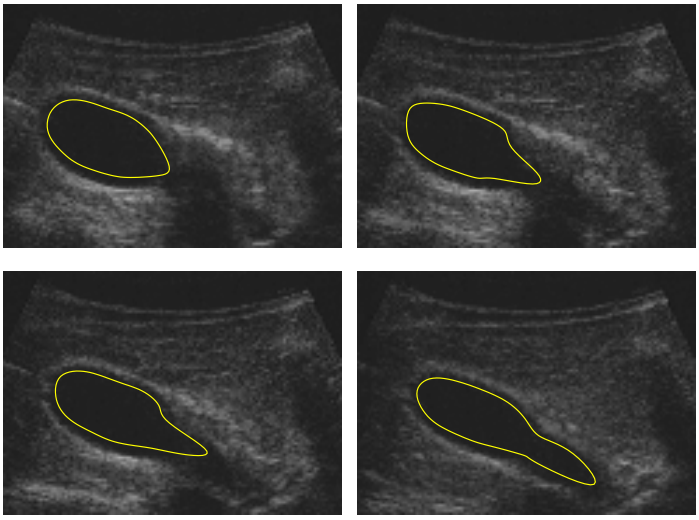

Figure 5: Dynamic clustering of an ultrasound sequence (affine model)

8 Conclusions

This paper presented a class of algorithms for dynamic clustering. These algorithms are obtained by an extension of a unified framework recently proposed by the authors for static clustering applications. A dynamic model encompassing both the global and local evolution of the centroids is considered, as well as the use of a noise model to achieve robust clustering in the presence of outliers. The proposed algorithms were tested with different motion models, using synthetic and real data. Accurate results were obtained in problems which are not easily solved using static clustering methods.

The relationship with well known algorithms is clear: the proposed method provides elegant extensions for several algorithms (e.g., c-means, Kohonen maps, elastic nets and fuzzy c-means) converting these algorithms into useful tools for the classification of noisy and dynamic data.

Appendix

Using the affine model (7), the partial derivatives (11,12) can be easily written as $M\,X - r$, in matrix notation, where

$$M = \begin{bmatrix} \Lambda + \gamma_1 I + \lambda 1 1^T & \Lambda Y^- & \Lambda 1 \\ (Y^-)^T \Lambda & \gamma_2 I + (Y^-)^T \Lambda Y^- & (Y^-)^T \Lambda 1 \\ 1^T \Lambda & 1^T \Lambda Y^- & \gamma_2 + 1^T \Lambda 1 \end{bmatrix}$$

$$r = \begin{bmatrix} \Lambda \xi \\ \gamma_2 A^- + (Y^-)^T \Lambda \xi \\ \gamma_2 b^- + 1^T \Lambda \xi \end{bmatrix}$$

where $\Lambda = diag(\mu_1, \cdots, \mu_m)$, A^-, b^- are the affine parameters at the previous instant of time, ξ is a matrix of centroids given by $\xi = [\xi_1, \cdots, \xi_m]^T$ and 1 is $m \times 1$ vector of ones.

Similar results can be obtained for the scaling global transformation (8) used in the first example of section 7.

References

[1] A. Abrantes, J. Marques, A Class of Constrained Clustering Algorithms for Object Boundary Extraction, *IEEE Trans. Image Processing*, 1507-1521, November, 1996.

[2] A. Abrantes, J. Marques, Pattern Recognition Methods for Object Boundary Detection, *Proc. British Machine Vision Conference, 1998*.

[3] McQueen, Some Methods for Classification and Analysis of Multivariate Observations, *Proc. 5th Berkeley Symp. Math. Stat. and Prob.*, Vol. I, Univ. California Press, pp. 281-286, 1967

[4] R. Gray, Vector Quantization, *IEEE ASSP Magazine*, 4-24, 1984.

[5] B. Duc, P. Schroeter, J. Bigun, Motion Estimation and Segmentation by Fuzzy Clustering, *Proc. IEEE Int. Conf. Image Processing*, Vol. III, 472-475, 1995.

[6] J. Ohm, P. Ma, Feature-Based Cluster Segmentation of Image Sequences, *IEEE Int. Conf. on Image Processing*, Vol. I, pp. 178-181, 1997.

[7] N. Oliver, A. Pentland, F. Berard, LAFTER: Lips and Face Real Time Tracker, *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 123-129, 1997.

[8] D. Terzopoulos, R. Szeliski, Tracking with Kalman Snakes, *Active Vision*, A. Blake and A. Yuille, eds., MIT Press, 1992.

[9] L. Cohen, I. Cohen, Finite-Element Methods for Active Contour Models and Balloons for 2-D and 3-D Images, *IEEE Trans. Pattern Analysis and Machine Intelligence*,Vol. 15, 1131-1147, 1993.

[10] T. Kohonen, The Self-Organizing Map, *Proceedings of the IEEE*, Vol. 78, N. 9, pp. 1464-1480, September, 1990.

[11] R. Durbin, D. Willshaw, An Analogue Approach to the Travelling Salesman Problem Using an Elastic Net Method, *Nature*, Vol. 326, pp. 689-691, April, 1987.

[12] J. Bezdek, A Convergence Theorem for the Fuzzy Isodata Clustering Algorithms, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-2, N. 1, January, 1980.

[13] Y. Bar-Shalom, T. Fortmann, *Tracking and Data Association*, Academic Press, 1988.

[14] L. Cohen, Auxiliary Variables and Two-Step Iterative Algorithms in Computer Vision Problems, *Journal of Mathematical Imaging and Vision*, 58-83, 1996.

[15] A. Dempster, N. Laird, D. Rubin, Maximum Likelihood from incomplete data via the EM algorithm, *Ann., Royal Stat. Soc.* , pp. 1-38, 1977.