Kernel-based fuzzy and possibilistic c-means clustering

Dao-Qiang Zhang and Song-Can Chen
Department of Computer Science
Nanjing University of Aeronautics and Astronautics
Nanjing, 210016, People's Republic of China
E-Mail: daoqz@mail.com

Abstract: The 'kernel method' has attracted great attention with the development of support vector machine (SVM) and has been studied in a general way. In this paper, this 'method' is extended to the well-known fuzzy c-means (FCM) and possibilistic c-means (PCM) algorithms. It is realized by substitution of a kernel-induced distance metric for the original Euclidean distance, and the corresponding algorithms are called kernel fuzzy c-means (KFCM) and kernel possibilistic c-means (KPCM) algorithms. And some test results are given to illustrate the advantages of the proposed algorithms over the FCM and PCM algorithms.

1. Introduction

Clustering has long been a popular approach to unsupervised pattern recognition. The fuzzy c-means (FCM) algorithm [1], as a typical clustering algorithm, has been utilized in a wide variety of engineering and scientific disciplines such as medicine imaging, bioinformatics, pattern recognition, and data mining. Since the original FCM uses the squared-norm to measure similarity between prototypes and data points, it can only be effective in clustering 'spherical' clusters. And many algorithms are derived from the FCM in order to cluster more general dataset. Most of those algorithms are realized by replacing the squared-norm in the object function of FCM with other similarity measures (metric) [1] [2]. In this paper, a kernel-based fuzzy c-means algorithm (KFCM) is proposed. KFCM adopts a new kernel-induced metric in the data space to replace the original Euclidean norm metric in FCM. By replacing the inner product with an appropriate 'kernel' function, one can implicitly perform a nonlinear mapping to a high dimensional feature space without increasing the number of parameters. This 'kernel method' has been successfully applied into many learning systems, such as Support Vector Machines (SVMs), kernel principal component analysis and kernel fisher discriminant analysis [3].

On the other hand, the FCM uses the probabilistic constraint that the memberships of a data point across classes sum to one. While this is useful in creating partitions, the memberships resulting from FCM and its derivatives, however, do not always correspond to the intuitive concept of degree of belongingness or compatibility. Moreover, the FCM is sensitive to noise. To mitigate such an effect, Krishnapuram and Keller throw away the constraint of memberships in FCM and propose the possibilistic c-means (PCM) algorithm [4]. The advantages of PCM are that it overcomes the need to specify the number of clusters and is highly robust in a noisy environment. However, there still exist some weaknesses in the PCM, i.e., it depends highly on a good initialization and has the undesirable tendency to produce coincident clusters [5] [6]. Usually, the FCM can provide a reasonable initialization and an estimate for the scale parameter which determines the relative degree to which the second term in the objective function is important compared with the first. But when the data is heavily noisy, the situation is quite different because the FCM is severely sensitive to outliers. In this paper, we propose a kernel possibilistic c-means (KPCM) algorithm. The KPCM uses the KFCM to initialize the memberships. In this way, the afore-mentioned weaknesses of the PCM can be avoided.

The rest of this paper is organized as following: In Section 2, we introduce the KFCM algorithm, and Section 3 presents the KPCM algorithm. Some test results and conclusions are given in Section 4.

2. Kernel fuzzy c-means clustering

Given a dataset, $X = \{x_1,...,x_n\} \subset R^p$, the original FCM algorithm partitions X into c fuzzy subsets by minimizing the following objective function

$$J_m(U,V)=\sum_{i=1}^{c}\sum_{k=1}^{n}u_{ik}^{m}\parallel x_k-v_i\parallel^2 . \quad (1)$$

where c is the number of clusters and selected as a specified value in this paper, n the number of data points, u_{ik} the membership of x_k in class i, satisfying $\sum_{i=1}^{c}u_{ik}=1$, m the quantity controlling clustering fuzziness, and V the set of cluster centers or prototypes ($v_i\in R^p$). The function J_m is minimized by a famous alternate iterative algorithm.

Now consider the proposed kernel fuzzy c-means (KFCM) algorithm. Define a nonlinear map as $\Phi: x\to\Phi(x)\in F$, where $x\in X$. X denotes the data space, and F the transformed feature space with higher even infinite dimension. KFCM minimizes the following objective function

$$J_m(U,V)=\sum_{i=1}^{c}\sum_{k=1}^{n}u_{ik}^{m}\parallel\Phi(x_k)-\Phi(v_i)\parallel^2 . \quad (2)$$

Where

$$\parallel\Phi(x_k)-\Phi(v_i)\parallel^2=K(x_k,x_k)+K(v_i,v_i)-2K(x_k,v_i) . \quad (3)$$

Where $K(x,y)=\Phi(x)^T\Phi(y)$ is an inner product kernel function. If we adopt the Gaussian function as a kernel function, i.e., $K(x,y)=\exp(-\parallel x-y\parallel^2/\sigma^2)$, then $K(x,x)=1$, according to Eqs. (3), Eqs. (2) can be rewritten as

$$J_m(U,V)=2\sum_{i=1}^{c}\sum_{k=1}^{n}u_{ik}^{m}(1-K(x_k,v_i)) . \quad (4)$$

Minimizing Eqs. (4) under the constraint of u_{ik}, we have

$$u_{ik}=\frac{(1/(1-K(x_k,v_i)))^{1/(m-1)}}{\sum_{j=1}^{c}(1/(1-K(x_k,v_j)))^{1/(m-1)}} \quad (5)$$

$$v_i=\frac{\sum_{k=1}^{n}u_{ik}^{m}K(x_k,v_i)x_k}{\sum_{k=1}^{n}u_{ik}^{m}K(x_k,v_i)} \quad (6)$$

Here we just use the Gaussian kernel function for simplicity. If we use other kernel functions, there will be corresponding modifications in Eqs. (5) and (6).

In fact, Equ.(3) can be viewed as kernel-induced new metric in the data space, which is defined as the following

$$d(x,y)\overset{\Delta}{=}\parallel\Phi(x)-\Phi(y)\parallel=\sqrt{2(1-K(x,y))} \quad (7)$$

And it can be proven that $d(x,y)$ defined in Eqs. (7) is a metric in the original space in case that *K(x,y)* takes as the Gaussian kernel function. According to Eqs. (6), the data point x_k is endowed with an additional weight $K(x_k,v_i)$, which measures the similarity between x_k and v_i, and when x_k is an outlier, i.e., x_k is far from the other data points, then $K(x_k,v_i)$ will be very small, so the weighted sum of data points shall be more robust.

The full description of KFCM algorithm is as follows:

KFCM Algorithm

Step 1: Fix c, $t_{\max}$, $m>1$ and $\varepsilon>0$ for some positive constant;

Step 2: Initialize the memberships u_{ik}^0;

Step 3: For t=1,2,…, $t_{\max}$, do:

(a) Update all prototypes v_i^t with Eqs. (6);

(b) Update all memberships u_{ik}^t with Eqs. (5);

(c) Compute $E^t=\max_{i,k}|u_{ik}^t-u_{ik}^{t-1}|$, if $E^t\le\varepsilon$, stop; else t=t+1.

3. Kernel possibilistic c-means clustering

The original FCM uses the probabilistic constraint that the memberships of a data point across classes sum to one. While this is useful in creating partitions, the memberships resulting from FCM and its derivatives, however, do not always correspond to the intuitive concept of degree of belonging or compatibility. Krishnapuram and Keller relax this constraint and propose a possibilistic approach to clustering (PCM) by minimizing the following object function

$$J_m(U,V)=\sum_{i=1}^{c}\sum_{k=1}^{n}u_{ik}^{m}\parallel x_k-v_i\parallel^2+\sum_{i=1}^{c}\eta_i\sum_{k=1}^{n}(1-u_{ik})^m . \quad (8)$$

where η_i are suitable positive numbers. The first term demands that the distances from data points to the prototypes be as low as possible, whereas the second term

Table 1 Clustering result for Fig. 1 (c) and (d)

	FCM		PCM		KFCM		KPCM	
	C1	C2	C1	C2	C1	C2	C1	C2
1	0.995	0.005	0.958	0.082	0.996	0.004	0.866	0.032
2	0.968	0.032	0.818	0.066	0.983	0.017	0.632	0.027
3	0.978	0.022	0.937	0.073	0.995	0.005	0.875	0.029
4	0.984	0.014	0.998	0.082	1.000	0.000	1.000	0.032
5	0.982	0.018	0.974	0.093	0.996	0.004	0.867	0.036
6	0.970	0.030	0.876	0.105	0.979	0.021	0.624	0.041
7	0.965	0.035	0.952	0.082	0.995	0.005	0.876	0.032
8	0.005	0.995	0.082	0.958	0.004	0.996	0.029	0.872
9	0.030	0.970	0.105	0.876	0.015	0.985	0.037	0.694
10	0.018	0.982	0.092	0.974	0.002	0.998	0.033	0.920
11	0.016	0.984	0.082	0.998	0.001	0.999	0.029	0.995
12	0.022	0.978	0.073	0.937	0.008	0.992	0.027	0.836
13	0.032	0.968	0.066	0.818	0.021	0.979	0.024	0.603
14	0.035	0.965	0.082	0.952	0.007	0.993	0.029	0.879
15	0.500	0.500	0.207	0.207	0.483	0.517	0.073	0.082
16	0.500	0.500	0.009	0.009	0.500	0.500	0.008	0.009
	(62.8, 159.6)		(61.1,150.2)		(60.5,150.4)		(59.9,149.9)	
	(137.2,159.6)		(138.9,150.2)		(137.9,150.9)		(139.0,149.9)	

forces the u_{ik} to be as large as possible, thus avoiding the trivial solution. It is recommended to select η_i as

$$\eta_i = K \frac{\sum_{k=1}^{n} u_{ik}^m \| x_k - v_i \|^2}{\sum_{k=1}^{n} u_{ik}^m} \tag{9}$$

Typically, K is chosen to be 1. The updating of prototypes is the same as that in FCM, but the memberships of PCM are updated as follows

$$u_{ik} = \frac{1}{1 + (\| x_k - v_i \|^2 / \eta_i)^{1/(m-1)}} \tag{10}$$

By following similar steps in KFCM, we construct the kernel possibilistic c-means (KPCM) algorithm by minimizing the following object function

$$J_m(U,V) = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^m \| \Phi(x_k) - \Phi(v_i) \|^2 + \sum_{i=1}^{c} \eta_i \sum_{k=1}^{n} (1-u_{ik})^m \tag{11}$$

As in KFCM, we adopt the Gaussian kernel function. Then the updating of memberships is

$$u_{ik} = \frac{1}{1 + (2(1-K(x_k,v_i))/\eta_i)^{1/(m-1)}} \tag{12}$$

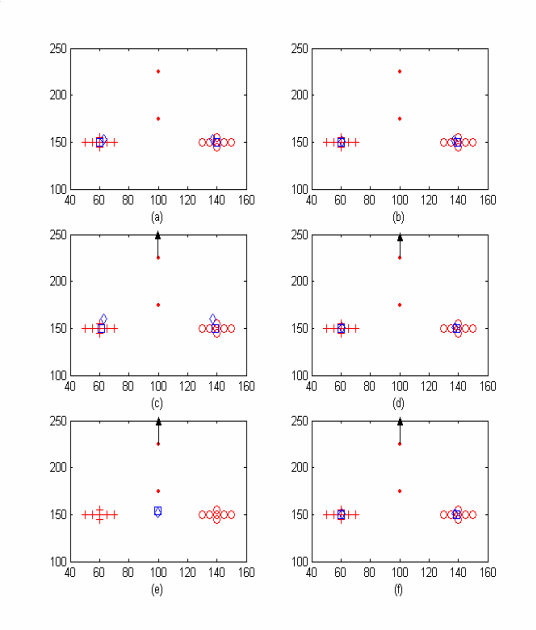

Fig. 1 Clustering result of Example 1 (Diamonds represent fuzzy prototypes, Squares the possibilistic one)

Here we use the Gaussian function as the kernel function, and η_i are estimated using

$$\eta_i = K \frac{\sum_{k=1}^{n} u_{ik}^m 2(1-K(x_k,v_i))}{\sum_{k=1}^{n} u_{ik}^m} \tag{13}$$

Typically, K is chosen to be 1, and the updating of prototypes is the same as Eqs. (6). We now summarize the KPCM algorithm as follows:

KPCM Algorithm

Step 1: Fix c, $t_{\max}$, $m > 1$ and $\varepsilon > 0$ for some positive constant;

Step 2: Initialize u_{ik}^0 using KFCM algorithm;

Step 3: Estimate η_i using Eqs. (13);

Step 4: For t=1,2,…, $t_{\max}$, do:

(a) Update all prototypes v_i^t with Eqs. (6);

(b) Update all memberships u_{ik}^t with Eqs. (12);

(c) Compute $E^t = \max_{i,k} | u_{ik}^t - u_{ik}^{t-1} |$, if $E^t \le \varepsilon$, stop; else t=t+1.

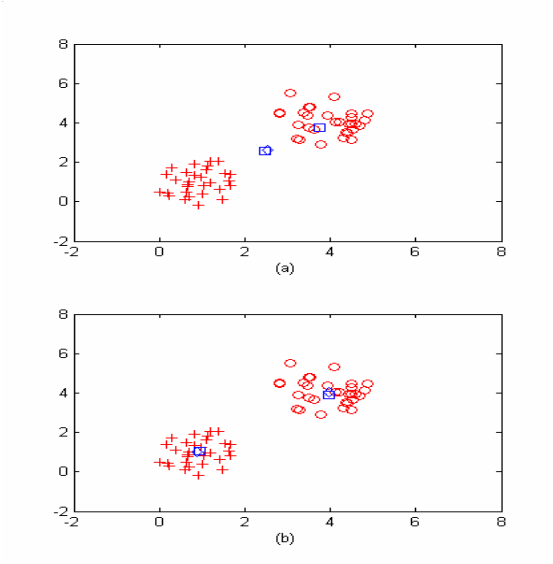

Fig. 2 Clustering result of Example 2 (Diamonds represent fuzzy prototypes, Squares the possibilistic one)

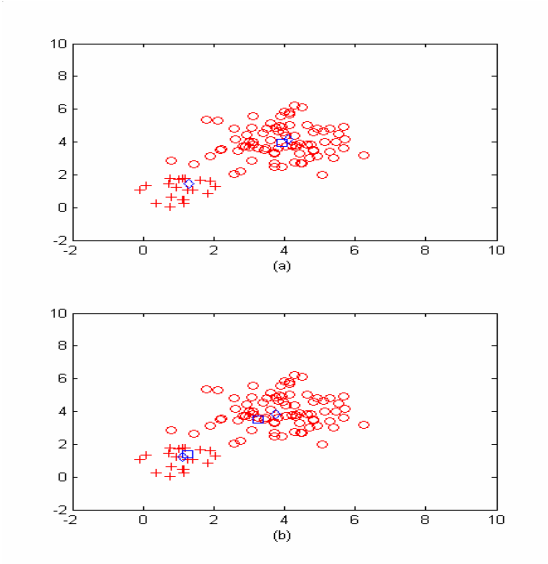

Fig. 3 Clustering result of Example 3 (Diamonds represent fuzzy prototypes, Squares the possibilistic one)

4. Simulation results and conclusion

In this section, we show several examples to illustrate the ideas presented in the previous sections. The first example involves two well-separated clusters. We number the data point according to the order in which they would be encountered from left to right and top to down in each clusters. Two outliers are added. One lies in (100,175), and the other is variable. Clustering results using FCM, PCM, KFCM, and KPCM algorithms are shown in Fig. 1. We add the other outlier (100,225) in Fig. 1(a) and (b), outlier (100,400) in Fig. 1(c) and (d), and outlier (100,800) in Fig. 1(e) and (f), respectively. Fig. 1(a), (c) and (e) are the results of FCM and PCM, and Fig. 1(b), (d) and (f) are gotten by KFCM and KPCM. The parameter used in the Gaussian kernel function is $\sigma = 150$. Table 1 gives the corresponding memberships and cluster centers of Fig. 1(c) and (d). It can be seen from Fig. 1 and Table 1 that KFCM and KPCM have much better robustness than FCM and PCM when the data sets contains one or more very large outliers.

The second example is two equal-sized clusters containing three outliers as shown in Fig. 2, where the outliers (40,0), (0,40) and (40,40) are not plotted in the figure. Fig. 2(a) and (b) shows the results using FCM, PCM and KFCM, KPCM, respectively. The FCM algorithm actually puts the farthest outlier points as one cluster, and lumps all the rest into another cluster, as shown in Fig. 2(a). In this case, using FCM as an initialization in PCM leads to a bad result. However, KFCM and KPCM are little affected by the outliers, as shown in Fig. 2(b). The third example involves two unequal-sized clusters. Fig. 3(a) is the result of FCM and PCM, and Fig. 3(b) the result using KFCM and KPCM. In this case, FCM and KFCM both represent the inherent structure of the dataset, but KFCM is a little superior to FCM. When PCM is used, the results are quite different. The prototypes got by PCM are nearly identical. However, that case in not appear in KPCM.

References

1. J.C. Bezdek. Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York, 1981
2. K.L. Wu, M.S. Yang, Alternative c-means clustering algorithms, Pattern Recognition 2002 35: 2267-2278
3. N. Cristianini, J.S. Taylor. An Introduction to SVMs and other kernel-based learning methods. Cambridge Univ. Press, 2000.
4. R. Krishnapuram, J.M. Keller. A possibilistic approach to clustering, IEEE Trans. Fuzzy Systems 1: 98-110, 1993.
5. M. Barni, V. Cappellini, A. Mecocci. Comments on "A possibilistic approach to clustering". IEEE Trans. Fuzzy Systems 4: 393-396, 1996.
6. R. Krishnapuram, J.M. Keller. The possibilistic c-means: insights and recommendations. IEEE Trans. Fuzzy Systems 4: 385-393, 1996.