International Journal of Computer Applications (0975 - 8887)
Volume 56 - No. 09, October 2012

Overlapping Patterns Recognition with Linear and Non-Linear Separations using Positive Definite Kernels

Chiheb-Eddine Ben N'Cir
LARODEC, ISGT, University of Tunis
41 rue de la liberte,
2000 Le Bardo, tunis, Tunisie

Nadia Essoussi
LARODEC, ISGT, University of Tunis
41 rue de la liberte,
2000 Le Bardo, tunis, Tunisie

ABSTRACT

The detection of overlapping patterns in unlabeled data sets referred as overlapping clustering is an important issue in data mining. In real life applications, overlapping clustering algorithm should be able to detect clusters with linear and non-linear separations between clusters. We propose in this paper an overlapping clustering method based k-means algorithm using positive definite kernel. The proposed method is well adapted for clustering multi label data with linear and non linear separations between clusters. Experiments, performed on overlapping data sets, show the ability of the proposed method to detect clusters with complex and non linear boundaries. Empirical results obtained with the proposed method outperforms existing overlapping methods.

General Terms:

Pattern recognition, Clustering

Keywords:

Overlapping Clustering, Multi-labels data, k-means algorithm, Non-linear Boundaries, Kernel methods

1. INTRODUCTION

Clustering is an important task in data mining. It aims to divide data into groups where similar observations are assigned to the same group called cluster. It has been applied successfully in many fields such as marketing that finds groups of customers with similar purchasing behaviors, biology that groups unlabeled plants or animals into species and document classification that groups related documents into clusters. Many applications of clustering require assigning observations to several clusters. This kind of application is referred as overlapping clustering [1, 7].

Overlapping clustering is based on the assumption that an observation can really belong to several clusters. In this cluster configuration, an observation may belong to one or several clusters without any membership coefficient and the resulting clustering is a cover. The resolution of this problem contributes to solve many real life problems that require to find overlapping clusters in order to fit the data set structure. For example, in video classification, overlapping clustering is a necessary requirement while video can potentially have multiple genres. In emotion detecting, overlapping clustering methods should be able to detect several emotions for a specific piece of music.

Several overlapping clustering methods based on hierarchical [9, 4], graph-based [10, 11] and partitioning [5, 1] approaches are proposed in the literature. An overlapping clustering method based k-means algorithm was proposed in [7] and referred as Overlapping k-means(OKM). The OKM method introduces optimality of overlapping clusters in the objective function which is optimized iteratively over the learning process. The main drawback of this method, as well as k-means based clustering methods, is its inability to detect clusters with complex and non linear clusters shapes. This problem can be crucial in real life applications where shapes of clusters are generally non linear and non spherical. To detect non linear separations between overlapping clusters, an existing method (Kernel overlapping k-means)[3] extends OKM by kernelization of the metric. This method is not well adapted to discover overlapping clusters since prototypes are performed in the input space and only distances are performed in the feature space.

We propose in this paper, a kernel based overlapping k-means method referred as Kernel Overlapping k-means (KOKMϕ) able to produce overlapping clusters with non linear overlapping boundaries making the method adapted for clustering complex data. In the proposed KOKMϕ method, kernels induce an implicit mapping of the input patterns and the algorithm is applied in a new space. Therefore, the representative of clusters will be performed in the induced space as well.

This paper is organized as follows: Section 2 gives a summary of existing overlapping clustering methods based k-means algorithm and describes advantages of Positive Definite Kernels (Mercer Kernel) used with clustering method. Section 3 presents the kernel based overlapping k-means method that we propose. Experiments on different data sets and using different kernels are described and discussed in Section 4. Finally, Section 5 presents conclusion and future works.

2. PRELIMINARIES

2.1 Mercer Kernel

To solve the problem of non spherical and non linear separations between clusters, many methods have been modified incorporating kernel such as SVM [8], Kernel PCA [15] and kernel k-means [13]. These proposed classification methods use Mercer Kernel to implicitly map data from original space called input space into a high dimensional space called feature space. Computing a linear partitioning in this feature space results in a non-linear partitioning in the input space.

A function $K : X \times X \longrightarrow \mathbb{R}$ is called a Mercer kernel if and only if K is symmetric and the following equation holds:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} c_i c_j K_{ij} \geq 0 \quad \forall N \geq 2, \tag{1}$$

where $c_r \in \mathbb{R} \quad \forall r = 1, ..., N$ and K_{ij} represents the dot product of mapped data in feature space that can be expressed as follows:

$$K_{ij} = \phi(x_i).\phi(x_j), \tag{2}$$

where $\phi : X \longrightarrow F$ performs a mapping from the input space X to a high dimensional feature space F.
The use of Mercer kernel in clustering methods can be divided in three categories [12]: methods based on *kernelization of the metric* [17, 18] which look for centroids in input space and the distances between patterns and centroids are computed by means of kernels, methods based on *clustering in feature space* [14, 13] which map data into a higher feature space and then compute centroids using the Kernel Trick and methods based *support vectors* [6, 2] which use one class SVM to find a minimum enclosing sphere in feature space able to enclose almost all data in feature space.

2.2 Kernel K-Means

Kernel k-means is an extension of the standard algorithm k-means to solve the problem of non-linearly separable clusters. For a finite data sample X, the Kernel k-means aims at minimizing the sum of squared Euclidean errors in feature space given by :

$$J(\Pi) = \sum_{i=1}^{N}\sum_{c=1}^{C} P_{ic}\|\phi(x_i) - m_c^{\phi}\|^2, \tag{3}$$

where P_{ic} is a binary variable indicating membership of observation x_i to cluster c and m_c^{ϕ} is the prototype of cluster c in feature space. The prototype is defined as the gravity center, in the feature space, of observations that belong to cluster c. This prototype cannot be computed because the mapping function ϕ is generally unknown. However, the clustering error $\| \phi(x_i) - m_c^{\phi}\|$ can be computed using the Kernel Trick as follows:

$$\|\phi(x_i) - m_c^{\phi}\|^2 = \|\phi(x_i) - \frac{1}{W_c}\sum_{j=1}^{N} P_{jc}\phi(x_j)\|^2$$

$$= K_{ii} - \frac{2}{W_c}\sum_{j=1}^{N} P_{jc}K_{ij} + \frac{1}{(W_c)^2}\sum_{j=1}^{N}\sum_{g=1}^{N} P_{jc}P_{gc}K_{jg}, \tag{4}$$

where $W_c = \sum_{j=1}^{N} P_{jc}$ is the number of observations that belong to cluster c, $P_{jc} \in \{0, 1\}$ and $P_{gc} \in \{0, 1\}$ indicate membership of observation x_j and x_g to cluster c. Then, the clustering error function in kernel k-means can be presented as follows:

$$J(\Pi) = \sum_{i=1}^{N}\sum_{c=1}^{C} P_{ic}[K_{ii} - \frac{2}{W_c}\sum_{j=1}^{N} P_{jc}K_{ij} + \frac{1}{(W_c)^2}\sum_{j=1}^{N}\sum_{g=1}^{N} P_{jc}P_{gc}K_{jg}]. \tag{5}$$

To minimize this clustering error function, kernel k-means performs two principal steps: the determination of the nearest cluster from each observation in feature space and the update of membership matrix of each object. The stopping rule is defined by the maximal number of iterations and the minimal improvement of the objective function between two iterations.

2.3 Overlapping k-means and Kernelization of the Metric

Overlapping k-means (OKM) [7]is an extension of the k-means algorithm and aims to produce overlapping clusters. The minimization of the objective function is performed by iterating two principal steps: 1) computation of clusters prototypes and 2) multi assignment of observations to one or several clusters. Given a set of data vectors $X = \{x_i\}_{i=1}^{N}$ with $x_i \in \mathbb{R}^d$ and N is the number of data vectors, the aim of OKM is to find a set $\Pi = \{\pi_c\}_{c=1}^{k}$ of k overlapping clusters such that the following objective function is minimized:

$$J(\Pi) = \sum_{x_i \in X} \|x_i - im(x_i)\|^2. \tag{6}$$

This objective function minimizes the sum of squared Euclidean distances between observation x_i and its image $im(x_i)$ for all $x_i \in X$. Image $im(x_i)$ is defined as the gravity center of clusters prototypes to which observation x_i belongs as shown by eq. 7.

$$im(x_i) = \sum_{c \in A_i} m_c/|A_i|, \tag{7}$$

where A_i is the set of clusters to which x_i belongs and m_c is the prototype of cluster c. The stopping rule of OKM algorithm is characterized by two criteria: the maximum number of iterations and the minimum improvement of the objective function between two iterations. Although the performance of this method to detect overlapping clusters, OKM method is not appropriate for clusters that have non linear separations. OKM method fails when clusters have a complex boundaries or when clusters are concentric.
A recent proposed method referred as Kernel overlapping k-means (KOKM) [3], proposes a kernelization of the metric used in OKM using the kernel induced distance measure. The objective function of KOKM minimizes the sum of kernel induced distance between observation x_i and its image $im(x_i)$ for all $x_i \in X$. The image $im(x_i)$ is computed in input space and then mapped to the feature space using $\phi(im(x_i))$.
Unlike Kernel k-means, the KOKM method have the drawbacks that images and prototypes are performed in input space and only distances between observations are performed in the feature space. The KOKM method belongs to the family of methods based kernelization of the metric where kernels play a role only in the computation of distances. In fact, Methods based on kernelization of the metric are less efficient then methods based clustering in feature space where all the learning process is performed in the feature space

3. KOKMϕ : KERNEL OVERLAPPING K-MEANS IN FEATURE SPACE

We propose in this paper a Kernel based overlapping clustering method where the whole learning process is performed in a high dimensional space like kernel k-means.
The main algorithm of KOKMϕ method iteratively minimizes the distance between each observation and its image in the feature space. The principal function to minimize can be described by:

$$J(\Pi) = \sum_{x_i \in X} \|\phi(x_i) - im(\phi(x_i))\|^2. \tag{8}$$

The image $im(\phi(x_i))$ is defined by the gravity center of clusters prototypes where observation x_i belongs. To improve algorithm efficiency, we consider in KOKMϕ that image is also performed in the feature space and is described by:

$$im(\phi(x_i)) = \frac{\sum_{c=1}^{C} P_{ic}.m_c^{\phi}}{\sum_{c=1}^{C} P_{ic}}, \quad (9)$$

where $P_{ic} \in \{0,1\}$ is a binary variable that indicates membership of observation x_i to cluster c and m_c^{ϕ} is the prototype of cluster c in the feature space.

3.1 Prototypes Computation in Feature Space

The computation of images in feature space needs the definition of clusters prototypes in the same induced space. The clusters centroids are replaced by clusters medoids where each cluster prototype is defined as the medoid (observation) that minimizes all distances over all observations included in this cluster. The prototype is expressed as follows:

$$m_c = \min_{i \in N_c}(x_i) \frac{\sum_{j=1, j\neq i}^{N_c} w_j \|\phi(x_i) - \phi(x_j)\|^2}{N_c. \sum_{j=1, j\neq i}^{N_c} w_j}, \quad (10)$$

where N_c is the number of observations that belong to cluster c and $w_j = |A_j|$ is a weight of the distance between observation x_j and observation x_i depending on the number of clusters to which observation x_j belongs. This weight is more important if observation j belongs to more than one cluster to take into account that overlapping observation x_j have a small probability to be a prototype of the cluster. In this way, the prototype is determined in the feature space F and is member of initial set of observations.
Using kernel function, the prototype can be determined as follows:

$$m_c = \min_{i \in N_c}(x_i) \frac{\sum_{j=1, j\neq i}^{N_c} w_j [K_{ii} - 2K_{ij} + K_{jj}]}{N_c. \sum_{j=1, j\neq i}^{N_c} w_j}. \quad (11)$$

3.2 Clustering Algorithm of KOKMϕ

Given clusters prototypes in the feature space, the objective function J can be computed as shown in eq. 12.

$$\begin{aligned} J(\Pi) &= \sum_{x_i \in X} \|\phi(x_i) - im(\phi(x_i))\|^2 \\ J(\Pi) &= \sum_{x_i \in X} \|\phi(x_i) - \frac{\sum_{c=1}^{C} P_{ic}.\phi(m_c)}{L_i}\|^2 \\ &= \sum_{x_i \in X} d[\phi(x_i), im(\phi(x_i))], \end{aligned} \quad (12)$$

where $d[\phi(x_i), im(\phi(x_i))]$ is defined by:

$$\begin{aligned} &= \phi(x_i).\phi(x_i) - \frac{2}{L_i} \sum_{c=1}^{C} P_{ic}.\phi(m_c).\phi(x_i) + \\ &\frac{1}{(L_i)^2} \sum_{c=1}^{C} \sum_{l=1}^{C} P_{ic}.P_{il}.\phi(m_c).\phi(m_l) \\ &= K_{ii} - \frac{2}{L_i} \sum_{c=1}^{C} P_{ic}.K_{im_c} + \frac{1}{(L_i)^2} \sum_{c=1}^{C} \sum_{l=1}^{C} P_{ic}.P_{il}.K_{m_c m_l} \end{aligned} \quad (13)$$

and $L_i = \sum_{c=1}^{C} P_{ic}$. At each iteration, clusters prototypes are computed, observations are assigned to many clusters and the function J is evaluated. These steps are repeated until improvement of J is not significative or the maximum number of iterations is reached. The main algorithm of KOKMϕ can be described as follows:

Algorithm 1 $KOKM\phi(X, t_{max}, \varepsilon, C) \rightarrow \{\pi_c\}_{c=1}^{C}$

Require: X: set of vector in $\mathbb{R}^d$.
t_{max}: maximum number of iterations.
ε: minimal improvement in objective function.
C: number of clusters.
Ensure: Π: set of C clusters.
1: Choose the kernel function and its corresponding parameters.
2: Initialize prototypes of clusters with random clusters prototypes, initialize clusters memberships using "$ASSIGN\phi$" and derive value of the objective function $J_{t=0}(\Pi)$ in iteration 0 using eq. 12.
3: Compute clusters prototypes using eq. 11.
4: Assign observations to one or several clusters using "$ASSIGN\phi$".
5: Compute objective function $J_t(\Pi)$ using eq. 12.
6: **if** ($t < t_{max}$ and $J_{t-1}(\Pi) - J_t(\Pi) > \varepsilon$) **then**
7: go to step 3.
8: **else**
9: return the distribution of clusters memberships.
10: **end if**

3.3 Multi assignment of observation to one or several clusters

The function $ASSIGN\phi$ is used to assign an observation to one or several clusters in KOKMϕ method. This function consists in assigning an observation iteratively to the closest cluster while the distance in the feature space between the observation and its image decreases. The closest cluster from an observation x_i in feature space is defined by:

$$m_c^{\star} = \min_{\{m_c\}_{c=1}^{C}} \|\phi(x_i) - \phi(m_c)\|^2. \quad (14)$$

This equation can be computed in feature space as follows:

$$\begin{aligned} m_c^{\star} &= \min_{\{m_c\}_{c=1}^{C}} \phi(x_i).\phi(x_i) - 2\phi(x_i).\phi(m_c) + \phi(m_c).\phi(m_c) \\ &= \min_{\{m_c\}_{c=1}^{C}} K_{ii} - 2K_{im_c} + K_{m_c m_c}, \end{aligned} \quad (15)$$

The ASSIGNϕ function used in the KOKMϕ method to assign observations to one or many clusters is summarized in Algorithm. 2 .

4. EXPERIMENTS AND DISCUSSIONS

Experiments are performed on Iris, Movie[1] and Music emotion[2] data sets. For each data set, the number of clusters was set by the number of underlying labels in the labeled data set. Table 1

[1] cf. http://www.grouplens.org/node/76.
[2] cf.http://mlkd.csd.auth.gr/multilabel.html

Algorithm 2 $ASSIGN\phi(x_i, \{m_c^0\}_{c=1}^C, A_i^{old}) \rightarrow A_i$

Require: x_i: Vector in $\mathbb{R}^d$.
$\{C_1 ... C_c\}$: Set of clusters.
A_i^{old}: Old affectation of observation i.
Ensure: A_i: New affectation of x_i.
1: Given $A_i = \{m_c^\star\}$ where $m_c^\star = \min_{\{C_c\}_{c=1}^C} \|\phi(x_i) - \phi(m_c)\|^2$, Compute distance in feature space between observation $\phi(x_i)$ and it's image $im(\phi(x_i))$ with affectations A_i using eq. 13.
2: Look for the next nearest cluster $m_c^\star$ which is not included in A_i such that $m_c^\star = \min_{\{C_c\}_{c=1}^C / A_i} \|\phi(x_i) - \phi(m_c)\|^2$ and compute distance between observation $\phi(x_i)$ and it's image $im'(\phi(x_i))$ with affectations $A_i \cup m_c^\star$
3: **if** $\|\phi(x_i) - im'(\phi(x_i))\|^2 < \|\phi(x_i) - im(\phi(x_i))\|^2$ **then**
4: $A_i \leftarrow A_i \cup \{m_c^\star\}$ and go to step 2.
5: **else**
6: compute im^{old} with affectation A_i^{old}.
7: **if** $\|\phi(x_i) - \phi(im(x_i))\|^2 \leq \|\phi(x_i) - \phi(im^{old}(x_i))\|^2$ **then**
8: return A_i.
9: **else**
10: return A_i^{old}.
11: **end if**
12: **end if**

Table 1. Statistics of used data sets

Data set	Observation	Dimension	Labels	Overlap
Iris	150	4	3	1
Movie	75	3	3	1.14
Music	548	72	6	1.81

shows the statistics of each data set. "labels" is the number of labels on the data set. "Overlap" is the average number of labels per observation.

$$Overlap = 1/|X| \sum_{x_i \in X} |\Pi_i|, \quad (16)$$

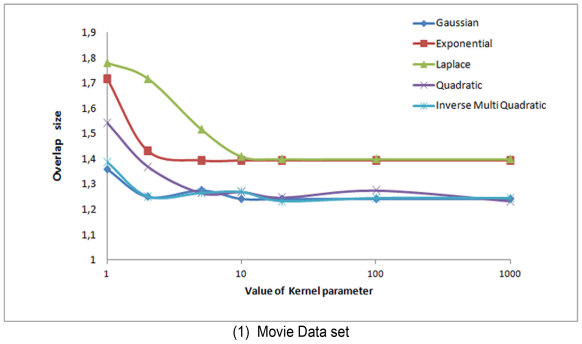
where $|X|$ is the number of observations and $|\Pi_i|$ is the number of label assignments of observation x_i. The size of overlap influences the performance of overlapping clustering methods when evaluated over external validation measures.
Results are compared according to three external validation measures: Precision, Recall and F-measure [16]. These validation measures attempt to estimate whether the prediction of categories is correct with respect to the underlying true categories in the data. Precision is calculated as the fraction of observation correctly labeled as belonging to the positive class divided by the total number of observations labeled as belonging to the positive class. Recall is the fraction of observations correctly labeled as belonging to the positive class divided by the total number of elements that actually belong to the positive class. The F-measure is the harmonic mean of Precision and Recall. All these measures are performed separately on each cluster than the average value of all clusters is reported.

$$\text{Precision} = NCLO/TNLO$$
$$\text{Recall} = NCLO/TNAC$$
$$\text{F-measure} = 2*\text{Precision}*\text{Recall} / \text{Precision}+\text{Recall}$$

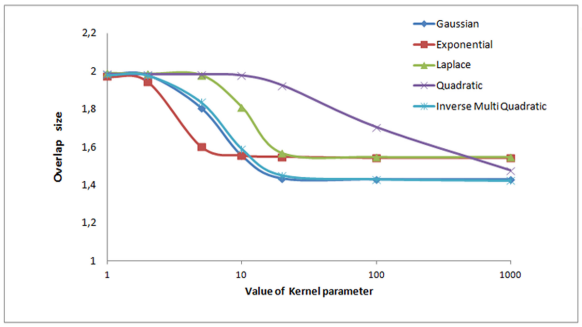
where NCLO, TNLO and TNAC are respectively the number of correctly labeled observations, the total number of labeled observations and the total number of observations that actually belong to the positive class.
Different widely used Positive Definite Kernels are implemented within the KOKMϕ method such as the Polynomial Kernel, the Gaussian kernel, the Exponential kernel, the Laplace kernel, the Quadratic and the Inverse Multi Quadratic kernel as described in Table 2.

Table 2. Examples of Positive Definite Kernels

Kernel function	Value
Linear Kernel	$K(x_i, x_j) = x_i.x_j$
Polynomial Kernel	$K(x_i, x_j) = ((x_i.x_j) + 1)^d$
Gaussian RBF Kernel	$K(x_i, x_j) = \exp(\frac{-\|x_i - x_j\|^2}{2\sigma^2})$
Exponential RBF Kernel	$K(x_i, x_j) = \exp(\frac{-\|x_i - x_j\|}{2\sigma^2})$
Laplace Kernel	$K(x_i, x_j) = \exp(\frac{-\|x_i - x_j\|}{\sigma})$
Quadratic Kernel	$K(x_i, x_j) = 1 - \frac{\| x_i - x_j \|^2}{\| x_i - x_j \|^2 + c}$
Inverse Multi quadratic Kernel	$K(x_i, x_j) = \frac{1}{\sqrt{\| x_i - x_j \|^2 + c^2}}$



(1) Movie Data set



(2) Music Data Set

Fig. 1. Impact of the value of kernel parameter used with the KOKMϕ method in Movie and Music data sets

To visualize structures of patterns detected by the proposed method with respect to the type of kernel, we build Voronoï cells (for 3 clusters) obtained with KOKMϕ method using different kernels with different parameters. Figure 2 proves the ability of KOKMϕ method used with the Polynomial and the Linear Kernel to detect overlapping clusters with linear boundaries. Figure 3 to Figure 7 prove the ability of KOKMϕ to detect overlapping clusters with non linear and non spherical separations. Some kernels have a similar behavior and can detect the same patterns : for example the Laplace kernel (with $\sigma = 100$) and the Quadratic kernel (with c=5500) build identical clusters shapes as shown in Figure 5 and Figure 6.
In Addition, the builded Voronoï cells show that overlapping boundaries between clusters become more smaller as well as the value of the kernel parameter increases. This result is also proved in real overlapping data sets as described in Figure 1 where the size of overlap builded by KOKMϕ method in both Movie and Music data sets decreases when the value of the Kernel parameter becomes larger.

Table 3. Comparison between KOKMϕ and existing methods in Iris, Movie and Music data sets

Dataset	Method	Precision	Recall	F-measure	Overlap
Iris	k-means	0.897 ± 0.02	0.886 ± 0.02	0.891 ± 0.02	**1.0** ± 0.00
	kernel k-means	**0.927** ± 0.01	0.928 ± 0.02	**0.927** ± 0.01	**1.0** ± 0.00
	OKM	0.707 ± 0.06	0.900 ± 0.07	0.815 ± 0.06	1.34 ± 0.08
	KOKM with RBF kernel	0.712 ± 0.05	0.894 ± 0.08	0.821 ± 0.07	1.29 ± 0.11
	KOKMϕ with RBF kernel	0.830 ± 0.09	**0.971** ± 0.05	0.895 ± 0.07	1.22 ± 0.10
Movie	k-means	0.731 ± 0.04	0.544 ± 0.04	0.623 ± 0.04	1.0 ± 0.00
	kernel k-means	**0.777** ± 0.03	0.605 ± 0.04	0.680 ± 0.03	1.0 ± 0.00
	OKM	0.520 ± 0.08	**0.802** ± 0.09	0.643 ± 0.08	1.60 ± 0.10
	KOKM with RBF kernel	0.590 ± 0.04	**0.791** ± 0.09	0.671 ± 0.05	1.55 ± 0.08
	KOKMϕ with RBF kernel	0.703 ± 0.15	0.687 ± 0.14	**0.692** ± 0.10	**1.24** ± 0.02
Music	k-means	0.501 ± 0.02	0.233 ± 0.02	0.288 ± 0.02	1.00 ± 0.00
	kernel k-means	**0.586** ± 0.04	0.213 ± 0.06	0.313 ± 0.05	1.00 ± 0.00
	OKM	0.397 ± 0.11	**0.332** ± 0.05	0.362 ± 0.08	2.45 ± 0.12
	KOKM with RBF kernel	0.401 ± 0.09	0.341 ± 0.01	0.372 ± 0.05	2.40 ± 0.13
	KOKMϕ with RBF kernel	**0.570** ± 0.07	**0.356** ± 0.01	**0.438** ± 0.01	**1.98** ± 0.01

Table 4. Results on Movie Data set using KOKMϕ methods with Different types of kernels

Kernel	Value	Precision	Recall	F-measure	Overlap
Linear		0.707 ± 0.15	0.680 ± 0.13	0.691 ± 0.10	1.22 ± 0.05
Polynomial kernel	$d = 2$	0.705 ± 0.06	0.674 ± 0.14	0.684 ± 0.09	1.25 ± 0.11
	$d = 3$	0.725 ± 0.05	0.687 ± 0.16	**0.701** ± 0.09	1.22 ± 0.11
	$d = 4$	0.723 ± 0.06	0.681 ± 0.01	**0.697** ± 0.10	1.22 ± 0.10
	$d = 5$	0.704 ± 0.03	0.615 ± 0.13	0.655 ± 0.07	1.22 ± 0.13
Gaussian RBF kernel	$\sigma = 1$	0.646 ± 0.14	0.697 ± 0.12	0.668 ± 0.09	1.36 ± 0.13
	$\sigma = 10$	0.703 ± 0.15	0.687 ± 0.14	**0.692** ± 0.10	1.24 ± 0.02
	$\sigma = 20$	0.703 ± 0.15	0.687 ± 0.14	**0.692** ± 0.10	1.24 ± 0.02
	$\sigma = 100$	0.703 ± 0.15	0.687 ± 0.14	**0.692** ± 0.10	1.24 ± 0.02
	$\sigma = 10000$	0.703 ± 0.15	0.687 ± 0.14	**0.692** ± 0.10	1.24 ± 0.02
Exponential RBF kernel	$\sigma = 1$	0.588 ± 0.09	0.746 ± 0.14	0.654 ± 0.06	1.71 ± 0.13
	$\sigma = 10$	0.655 ± 0.13	0.736 ± 0.11	0.689 ± 0.08	1.39 ± 0.13
	$\sigma = 20$	0.655 ± 0.13	0.736 ± 0.11	0.689 ± 0.08	1.39 ± 0.13
	$\sigma = 100$	0.655 ± 0.13	0.736 ± 0.11	0.689 ± 0.08	1.39 ± 0.13
	$\sigma = 10000$	0.655 ± 0.13	0.736 ± 0.11	0.689 ± 0.08	1.39 ± 0.13
Laplace kernel	$\sigma = 1$	0.568 ± 0.08	0.715 ± 0.14	0.627 ± 0.09	1.78 ± 0.03
	$\sigma = 20$	0.649 ± 0.12	0.736 ± 0.11	0.686 ± 0.07	1.41 ± 0.13
	$\sigma = 100$	0.655 ± 0.13	0.736 ± 0.11	0.689 ± 0.08	1.40 ± 0.13
	$\sigma = 10000$	0.655 ± 0.13	0.736 ± 0.11	0.689 ± 0.08	1.40 ± 0.13
Quadratic kernel	$c = 1$	0.613 ± 0.11	0.738 ± 0.15	0.663 ± 0.06	1.54 ± 0.00
	$c = 10$	0.685 ± 0.11	0.675 ± 0.15	0.677 ± 0.14	1.27 ± 0.10
	$c = 20$	0.663 ± 0.13	$0.0.672 \pm 0.13$	0.663 ± 0.13	1.24 ± 0.16
	$c = 100$	0.672 ± 0.12	0.674 ± 0.14	0.669 ± 0.13	1.27 ± 0.10
	$c = 1000$	0.707 ± 0.15	0.685 ± 0.14	**0.693** ± 0.11	1.23 ± 0.02
Inverse multi quadratic kernel	$c = 1$	0.640 ± 0.13	0.707 ± 0.11	0.667 ± 0.08	1.38 ± 0.22
	$c = 10$	0.672 ± 0.13	0.670 ± 0.15	0.667 ± 0.14	1.27 ± 0.10
	$c = 20$	0.713 ± 0.16	0.687 ± 0.14	**0.697** ± 0.11	1.23 ± 0.02
	$c = 100$	0.644 ± 0.14	0.625 ± 0.17	0.630 ± 0.16	1.24 ± 0.08
	$c = 1000$	0.644 ± 0.14	0.625 ± 0.17	0.630 ± 0.16	1.24 ± 0.08

Table 3 presents results obtained with KOKMϕ method versus k-means, kernel k-means, OKM and KOKM methods in terms of precision, recall and F-measure for the data sets described in Table 1. Each reported result is an average over twenty runs of each algorithm with the same initialization on each run. For all kernel based methods, we use the Gaussian RBF Kernel with the best parameter value (determined empirically through different tests). The F-measure obtained with KOKMϕ method outperforms the F-measure obtained with existing methods. The improvement of F-measure using KOKMϕ compared to OKM and KOKM methods, is induced by the improvement of the Precision and the Recall. This result proves the theoretical finding that looking for separations between clusters in the feature space is better than looking for separations in the original space.

The choice of the kernel function and its parameters influences the performance of the proposed method and influences shapes of the detected boundaries. Table 4 and Table 5 summarize results obtained with KOKMϕ method using Positive Definite Kernels where excessive experiments with different values of the kernel parameter are reported. We note the variation of obtained results as well as the kernel and its parameter are modified. For some Kernels, the variation of the parameter becomes without influence in some intervals : for example obtained results using the Gaussian Kernel when $\sigma \in [20.. + \infty]$ are identical on both Music and Movie data sets.

Table 5. Results on Music Data set using KOKMϕ methods with Different types of kernels

Kernel	Value	Precision	Recall	F-measure	Overlap
Linear		0.492 ± 0.01	0.301 ± 0.02	0.374 ± 0.02	1.42 ± 0.05
Polynomial kernel	$d = 2$	0.480 ± 0.02	0.286 ± 0.02	0.358 ± 0.01	1.40 ± 0.05
	$d = 3$	0.486 ± 0.01	0.300 ± 0.01	0.370 ± 0.01	1.45 ± 0.01
	$d = 4$	0.482 ± 0.02	0.289 ± 0.01	0.364 ± 0.01	1.41 ± 0.05
	$d = 5$	0.486 ± 0.01	0.284 ± 0.01	0.363 ± 0.01	1.43 ± 0.03
Gaussian RBF kernel	$\sigma = 1$	0.570 ± 0.07	0.356 ± 0.01	$\mathbf{0.438} \pm 0.01$	1.98 ± 0.01
	$\sigma = 10$	0.481 ± 0.02	0.318 ± 0.02	0.383 ± 0.02	1.55 ± 0.04
	$\sigma = 20$	0.490 ± 0.01	0.291 ± 0.03	0.365 ± 0.03	1.43 ± 0.08
	$\sigma = 100$	0.494 ± 0.02	0.302 ± 0.02	0.375 ± 0.02	1.43 ± 0.05
	$\sigma = 10000$	0.493 ± 0.02	0.302 ± 0.02	0.375 ± 0.02	1.43 ± 0.05
Exponential RBF kernel	$\sigma = 1$	0.476 ± 0.03	0.375 ± 0.05	$\mathbf{0.420} \pm 0.04$	1.99 ± 0.01
	$\sigma = 10$	0.480 ± 0.02	0.319 ± 0.02	0.383 ± 0.02	1.55 ± 0.05
	$\sigma = 20$	0.480 ± 0.02	0.317 ± 0.02	0.382 ± 0.01	1.54 ± 0.05
	$\sigma = 100$	0.480 ± 0.02	0.317 ± 0.02	0.382 ± 0.01	1.54 ± 0.05
	$\sigma = 10000$	0.480 ± 0.02	0.317 ± 0.02	0.382 ± 0.01	1.54 ± 0.05
Laplace kernel	$\sigma = 1$	0.472 ± 0.04	0.365 ± 0.03	$\mathbf{0.412} \pm 0.04$	1.98 ± 0.01
	$\sigma = 20$	0.476 ± 0.02	0.342 ± 0.02	0.398 ± 0.02	1.81 ± 0.01
	$\sigma = 100$	0.478 ± 0.02	0.320 ± 0.01	0.383 ± 0.01	1.56 ± 0.05
	$\sigma = 10000$	0.480 ± 0.02	0.317 ± 0.02	0.382 ± 0.01	1.54 ± 0.05
Quadratic kernel	$c = 1$	0.476 ± 0.03	0.374 ± 0.05	$\mathbf{0.419} \pm 0.04$	1.98 ± 0.01
	$c = 10$	0.476 ± 0.03	0.371 ± 0.04	0.417 ± 0.04	1.97 ± 0.01
	$c = 20$	0.476 ± 0.03	0.355 ± 0.05	0.407 ± 0.04	1.92 ± 0.01
	$c = 100$	0.476 ± 0.02	0.325 ± 0.027	0.386 ± 0.02	1.70 ± 0.06
	$c = 1000$	0.486 ± 0.02	0.298 ± 0.04	0.369 ± 0.03	1.47 ± 0.06
Inverse multi Quadratic kernel	$c = 1$	0.476 ± 0.03	0.373 ± 0.05	$\mathbf{0.418} \pm 0.04$	1.98 ± 0.01
	$c = 10$	0.477 ± 0.01	0.317 ± 0.02	0.381 ± 0.02	1.59 ± 0.04
	$c = 20$	0.486 ± 0.03	0.308 ± 0.02	0.377 ± 0.02	1.45 ± 0.09
	$c = 100$	0.495 ± 0.01	0.302 ± 0.02	0.375 ± 0.02	1.43 ± 0.07
	$c = 1000$	0.494 ± 0.01	0.296 ± 0.01	0.370 ± 0.01	1.42 ± 0.05

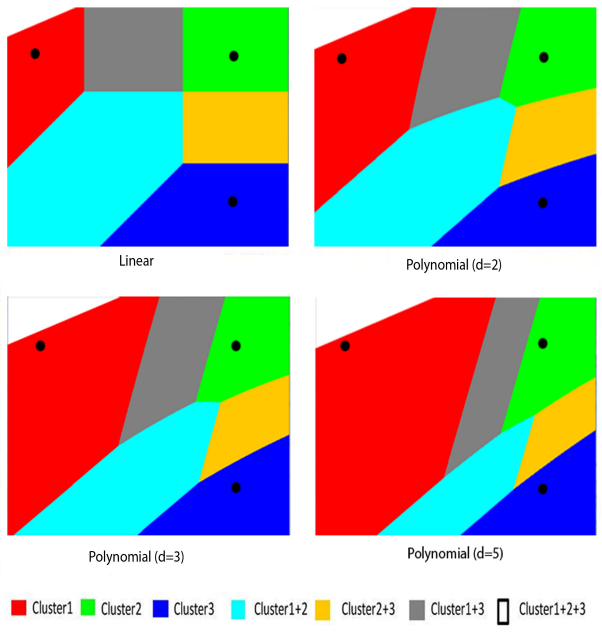


Fig. 2. Voronoi cells obtained with Linear and Polynomial kernel using KOKMϕ method

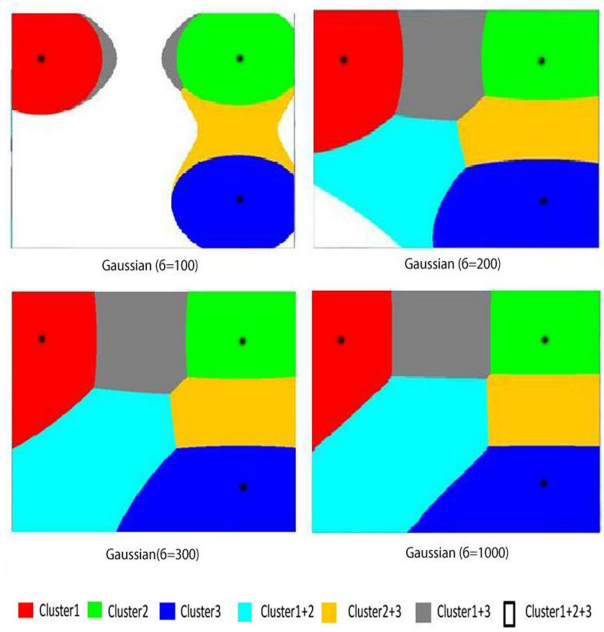


Fig. 3. Voronoi cells obtained with Gaussian RBF kernel using KOKMϕ method

In fact, the Kernel contains all information about structures of patterns in the feature space through the Kernel Matrix. By varying values of the kernel parameter, two extreme situations may be reached : the overfitting or the underfitting. Geometrically, the overfitting corresponds to patterns being mapped to orthogonal points in the feature space, while in the second situation all points are merged into the same feature mapping. Numerically, the overfitting situation is reached when the off-diagonal entries of the Kernel Matrix become very small and the diagonal en-

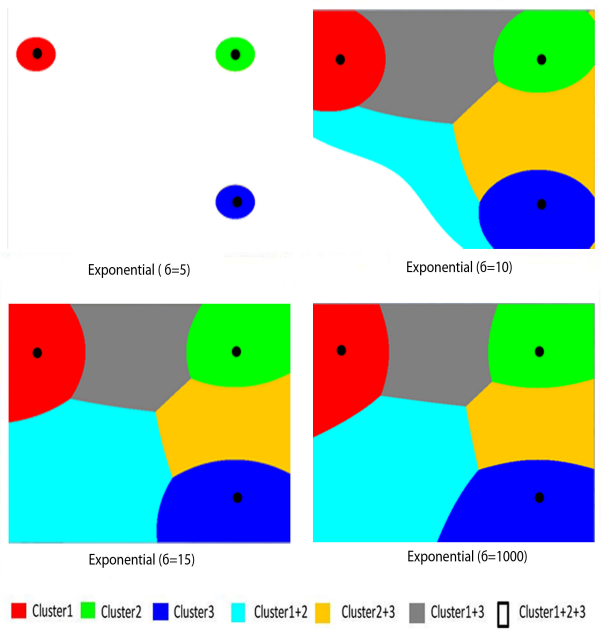


Fig. 4. Voronoi cells obtained with Exponential kernel using KOKMϕ method

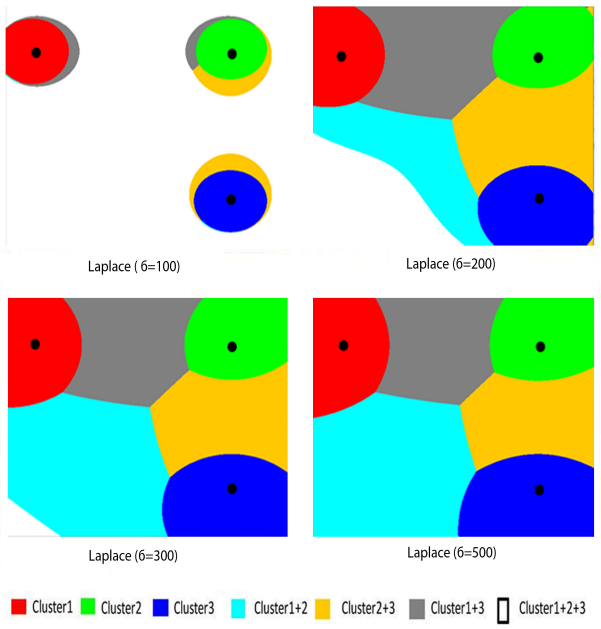


Fig. 5. Voronoi cells obtained with Laplace kernel using KOKMϕ method

tries are close to 1. However, if a Kernel Matrix is completely uniform, the underfitting situation is reached.

5. CONCLUSION

We proposed in this paper a kernel based overlapping k-means method to detect overlapping patterns in unlabeled data sets. The proposed method performs all the learning process in a high dimensional feature space where data are explicitly mapped using Positive Definite Kernels. Experiments prove the efficiency of KOKMϕ to detect clusters with linear and non linear boundaries making the method adapted for real life applications of overlapping clustering where separations between clusters are complex.

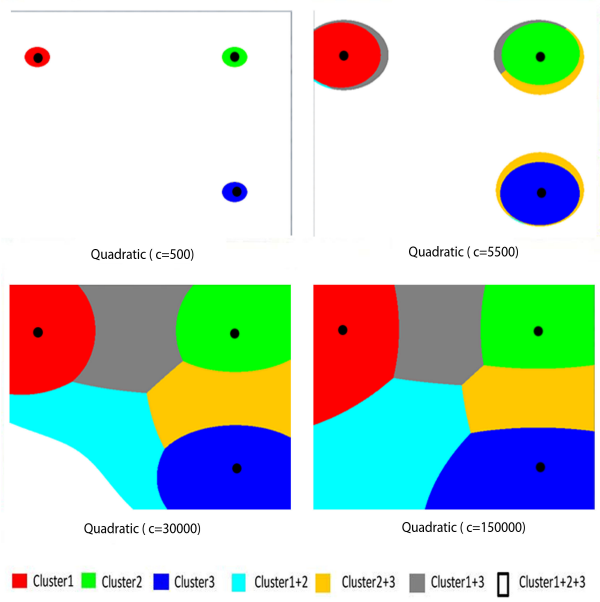


Fig. 6. Voronoi cells obtained with Quadratic kernel using KOKMϕ method

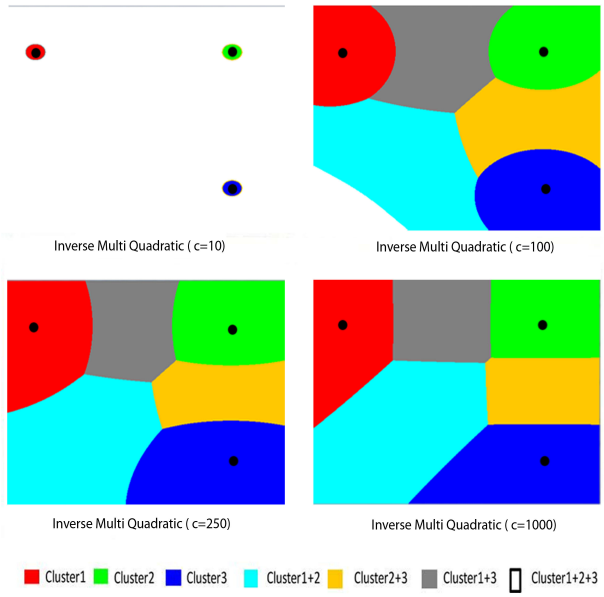


Fig. 7. Voronoi cells obtained with Inverse Multi Quadratic kernel using KOKMϕ method

This proposed method can be applied for many other application domains where observations needs to be assigned to more than one cluster and where patterns cannot be described by explicit feature vectors such as images and texts. For such data sets, we plan to conduct experiments on structured non vectorial data using a specific designed kernels such as Strings and Histograms.

6. REFERENCES

[1] Arindam Banerjee, Chase Krumpelman, Sugato Basu, Raymond J. Mooney, and Joydeep Ghosh. Model based overlapping clustering. In *International Conference on Knowledge Discovery and Data Mining*, Chicago, USA, 2005. SciTePress.

[2] Asa Ben-Hur, David Horn, Hava T. Siegelmann, and Vladimir Vapnik. Support vector clustering. *Journal Of Machine Learning Re-*

search, 2:125–137, 2001.

[3] Chiheb BenN'Cir, Nadia Essoussi, and Patrice Bertrand. Kernel overlapping k-means for clustering in feature space. In *International Conference on Knowledge discovery and Information Retrieval KDIR*, pages 250–256, Valencia, SPA, 2010. SciTePress Digital Library.

[4] P. Bertrand and M. F. Janowitz. The k-weak hierarchical representations: an extension of the indexed closed weak hierarchies. *Discrete Applied Mathematics*, 127(2):199–220, 2003.

[5] James C. Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers amp; Geosciences*, 10(23):191 – 203, 1984.

[6] Francesco Camastra and Alessandro Verri. A novel kernel method for clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:801–804, 2005.

[7] Guillaume Cleuziou. An extended version of the k-means method for overlapping clustering. In *International Conference on Pattern Recognition ICPR*, pages 1–4, Florida, USA, 2008. IEEE.

[8] Corinna Cortes and Vladimir Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[9] E. Diday. Orders and overlapping clusters by pyramids. Technical Report 730, INRIA, France, 1984.

[10] Walter Didimo, Francesco Giordano, and Giuseppe Liotta. Overlapping cluster planarity. In *Proceedings of the six International Asia-Pacific Symposium on Visualization*, pages 73–80, 2007.

[11] Michael R. Fellows, Jiong Guo, Christian Komusiewicz, Rolf Niedermeier, and Johannes Uhlmann. Graph-based data clustering with overlaps. *Discrete Optimization*, 8(1):2–17, 2011.

[12] Maurizio Filippone, Francesco Camastra, Francesco Masulli, and Stefano Rovetta. A survey of kernel and spectral methods for clustering. *Pattern Recognition*, 41(1):176 – 190, 2008.

[13] Mark Girolami. Mercer kernel-based clustering in feature space. *IEEE Transactions on Neural Networks*, 13(13):780–784, 2002.

[14] A. K. Qinand and P. N. Suganthan. Kernel neural gas algorithms with application to cluster analysis. *International Conference on Pattern Recognition*, 4:617–620, 2004.

[15] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[16] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Mining Multi-label Data. In *Data Mining and Knowledge Discovery Handbook*, chapter 34, pages 667–685. Boston, MA, 2010.

[17] Daoqiang Zhang and Songcan Chen. Kernel-based fuzzy and possibilistic c-means clustering. In *International Conference on Artificial Neural Networks (ICANN03)*, pages 122–125, Istanbul, Turkey, 2003.

[18] Daoqiang Zhang and Songcan Chen. A novel kernelized fuzzy c-means algorithm with application in medical image segmentation. *Artificial Intelligence in Medicine*, 32(1):37–50, 2004.