Université de Tunis
Institut Supérieur de Gestion
Ecole Doctorale Sciences de Gestion

Efficient k-means based methods for overlapping clustering

THESE
En vue de l'obtention du Doctorat en
Informatique de Gestion

Présentée et soutenue publiquement par
Chiheb-Eddine Ben N'Cir

Le 09 Juin 2014

Membres du Jury

M.	Abdelwahed TRABELSI	Professeur	ISG, Tunis	Président
M.	Sadok BEN YAHIA	Professeur	FST, Tunis	Rapporteur
M.	Lamjed BEN SAID	Maitre de conférences	ISG, Tunis	Rapporteur
M.	Zied ELOUEDI	Professeur	ISG, Tunis	Membre
Mme.	Nadia ESSOUSSI	Maitre de conférences	FSEG, Nabeul	Directeur

Abstract

This thesis focuses on the identification of non-disjoint groups from unlabeled data, referred to as Overlapping Clustering, with a special emphasis on overlapping methods based on the partitional approach. Overlapping clustering has the characteristic that an object could be assigned to several groups which offers a richer model for fitting existing structures in several applications requiring non-exclusive partitioning. In this context, we firstly propose two methods, KOKMI and KOKMII, which are based on kernel methods in order to look for overlapping clusters with both *linear* and *nonlinear* separations. The identification of nonlinear separations becomes a necessary requirement given that data structuring in real life applications are usually complex. Next, the thesis goes beyond the control of the size of overlaps between clusters given that this size is not unique and depends on the considered application. We propose a generic overlapping clustering model with overlap regulation which supports different instantiations able to *restrict*, to *regulate* or to *auto-control* the overlapping boundaries between clusters. The third contribution of this thesis deals with the identification of non-disjoint groups from textual documents. We designed a kernel based method, referred to as KOKM based WSK, using the Word Sequence Kernel as similarity measure between documents. Proposed methods are integrated in the software Weka4OC offering for researchers and developers working on overlapping clustering the possibility to *build*, to *visualize* and to *evaluate* non-disjoint partitioning.

Key words: overlapping clustering, non-disjoint partitioning, multi-labeled data, size of overlaps, nonlinear separations, kernel methods, overlap regulation.

Résumé

La présente thèse porte sur l'identification des groupes non-disjoints à partir de données non étiquetées, appelée classification recouvrante, avec un intért particulier sur les méthodes recouvrantes basées sur le principe de partitionnement. Ce type de classification permet à un objet dappartenir à la fois à plusieurs groupes offrant ainsi un modèle plus riche pour des applications nécessitant une organisation non-exclusive des données. Dans ce cadre, nous proposons en premier lieu deux méthodes, KOKMI et KOKMII, basées sur les méthodes de noyaux et capables de produire des recouvrements de classes ayant des séparations aussi bien *linéaires* que *non-linéaires*. Lidentification des séparations non-linéaires savère nécessaire vu que les structurations des données dans les applications réelles sont généralement complexes. La deuxième contribution de cette thèse concerne la capacité à contrôler la taille des intersections entre les classes puisque cette taille dépend de lapplication concernée. Nous proposons un modèle générique avec recouvrement de classes offrant différentes instances capables de *restreindre*, de *régler* ou *d'auto-ajuster* les zones de chevauchement entre les classes. La troisième contribution porte sur lorganisation automatique des documents textuels en groupes non-disjoints. Nous proposons la méthode KOKM based WSK utilisant le noyau de séquence de mots comme mesure de similarité entre les documents. Les méthodes proposées sont intégrées dans le logiciel Weka4OC permettant ainsi, aux chercheurs de la communauté de classification recouvrante, la possibilité de *construire*, de *visualiser* et *d'évaluer* des regroupements non-disjoints.

Mots clés: classification recouvrante, partitionnement non-disjoints, données avec multi-labels, taille des chevauchements, séparations non-linéaires, méthodes à noyaux, ajustement des recouvrements.

Acknowledgment

I would like to express my sincerest thanks to my advisor, *Dr. Nadia Essoussi, Associate Professor at the FSEG Nabeul, University of Carthage*, for her great guidance, scientific inputs, support and understanding throughout the work of this thesis. I would like to thank her for encouraging my research and for allowing me to grow as a research scientist.

I would like to express my special appreciation and thanks to *Dr. Mohamed Limam, Professor at ISG Tunis, University of Tunis*, for his advice and assistance when starting the works of this Thesis.

I would like to thank *Dr. Guillaume Cleuziou, Associate Professor at the University of Orleans*, for the comments and the interesting discussions regarding overlapping clustering.

It is my honor that the jury members accepted the invitation and spent their precious time helping me to improve this thesis. I wish equally to express my gratitude to them.

I am also very grateful to have been a PhD student at the LARODEC Laboratory, which always provided me with a lively and intellectually stimulating environment in which to work. I would like to thank all my professors that have learned me.

A special thanks to my family. Words cannot express how grateful I am to my mother and father, for all of the sacrifices that you have made on my behalf. Your prayer for me was what sustained me thus far.

A special thanks to my fiancee that has selflessly given more to me than I ever could have asked for.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

ALS:	Alternating Least Square algorithms
ARFF:	Attribute-Relation File Format
BCM:	Belief c-means
DBMS:	Data Base Management System
DBSCAN:	Density Based Spatial Clustering of Applications with Noise
DENCLUE:	DENsity-based CLUstEring
ECM:	Evidential c-means
EM:	Expectation-Maximization algorithm
FCM:	Fuzzy c-means
FN:	False Negative
FP:	False Positive
GHZ:	Gigahertz
GUI:	Graphical User Interface
IMDB:	Internet Movie DataBase
IR:	Information Retrieval
KOKMI:	Kernel Ovelapping k-means I
KOKMII:	Kernel Ovelapping k-means II
LSK:	Latent Semantic Kernel
MAFIA:	Merging of Adaptive Finite IntervAls
MCLUST:	Model based CLUSTering
MEDLINE:	On-Line Medical Information database
OClustR:	Overlapping Clustering based on Relevance
OKMED:	Overlapping k-medoid
OKM:	Overlapping k-means
OPTICS:	Ordering Points To Identify the Clustering Structure
PAM:	Partitioning Around Medoid
PCA:	Principal Component analysis
PCL:	Principal Cluster Analysis
PCM:	Possiblistic c-means

RAM:	Random Access Memory
R-ASSIGN:	Regulated Assignments
RBF:	Radial basis Function
R-OKM:	Restricted OKM
SOM:	Self Organizing Maps
SQL:	Structured Query Language
SSK:	String Subsequence Kernel
SVM:	Support Vector Machine
STING:	STatistical INformation Grid-based Method
TF-IDF:	Term Frequecy- Inverse Document Frequency
TN:	True Negative
TP:	True Positive
URL:	Unified Resource Locator
VSM:	Vector Space Model
Weka4OC:	Weka For Overlapping Clustering
WOKM:	Weighted Overlapping k-means
WSK:	Word Sequence kernel
XML:	eXtensible Markup Language
XRFF:	eXtensible attribute-Relation File Format
2D:	Two Dimensions

Introduction

Given the explosion of data available from different sources such as web, scientific researches, companies and social networks, developing methods for analyzing data efficiently has become increasingly important. In this way, clustering is considered as an interesting technique in data mining and pattern recognition to predict and to summarize data. It has been applied successfully in many fields such as market segmentation (DeSarbo and Cron 1988; Hattum and Hoijtink 2009), social network analysis (Wang and Fleury 2011; Pérez-Suárez et al. 2013), document classification (Aliguliyev 2009; Liu et al. 2011), etc. Clustering aims to group similar observations into the same group and dissimilar observations into different groups. For example, given the task of clustering animals, one might group them together by type resulting in groups of mammals, reptiles and amphibians; or alternatively by size resulting in small-size and large-size groups. Typically, any particular dataset does not have a uniquely correct clustering, and the desired clustering may depend on the particular application.

The problem of clustering data has led to many methods which are widely used across many fields. Unfortunately, while clustering methods are an interesting tool for many applications, they are actually quite limited. Clustering methods traditionally assume that each observation belongs to one and only one cluster leading to k exhaustive and disjoint clusters explaining the data. In many situations the data being modeled can have a much richer and more complex hidden representation than this disjoint partitioning. For example, there may be overlapping regions where observations actually belong to multiple clusters. Solutions to this problem contribute to solve many real applications that require to find overlapping regions in order to fit the dataset structure. For example, in social network analysis, community extraction algorithms need to detect overlapping clusters where an actor can belong to multiple communities (Tang and Liu 2009; Wang et al. 2010; Fellows et al. 2011). In video classification, overlapping clustering is a necessary requirement where videos have potentially multiple genres (Snoek et al. 2006). In emotion detection, overlapping clustering methods need to detect different emotions for a specific piece of music (Wieczorkowska et al. 2006). In text clustering, learning methods should be able to group document, which discuss more than one theme, into several groups (Gil-García and Pons-Porrata 2010; Pérez-Suárez et al. 2013), etc. A trend in designing clustering methods called "overlapping clustering" has tried to solve these problems which have arisen naturally.

In fact, overlapping clustering is based on the assumption that an observation may belong to one or several clusters. In this cluster configuration, obtained groups are usually non-disjoints and are called covers or clusters. Several overlapping clustering methods based on hierarchical (Diday 1984; Bertrand and Janowitz 2003), graph (Suárez et al. 2009; Fellows et al. 2011; Hasan et al. 2011; Pérez-Suárez et al. 2013), generative (Banerjee et al. 2005; Heller and Ghahramani 2007; Fu and Banerjee 2008), correlation (Bonchi et al. 2011; Bonchi et al. 2013) and partitional (Mirkin 1987; Mirkin 1990; Depril et al. 2008; Masson and Denoeux 2008) approaches are proposed in the literature. Our work distinguishes from this body of research as it is developed within partitional methods, especially those based on k-means algorithm (MacQueen 1967; Jain 2010). A category of these methods extends results of uncertain partitioning methods, such as results of fuzzy c-means, possiblistic c-means (Krishnapuram and Keller 1993) and evidential c-means (Masson and Denoeux 2008) to obtain overlapping clusters. These methods need a post-processing treatment to generate the final overlapping clusters either by thresholding clusters memberships or by maximizing memberships evidence. More perfective methods (Cleuziou 2008; Cleuziou 2009; Depril et al. 2008) are based on the generalization of the optimized criterion of k-means to look for optimal covers. As opposed to uncertain partitioning methods, these methods produce hard overlapping clusters and do not need any post processing treatment.

Challenges of the Thesis

Although the ability of existing methods to build non-disjoint groups, many suffer from some limitations which motivate researchers to design more efficient methods. Almost existing methods perform linear and spherical separations between clusters (Masson and Denoeux 2008; Cleuziou 2008; Cleuziou 2009; Depril et al. 2008; Bonchi et al. 2011; Fellows et al. 2011; Liu et al. 2012), thus fail to model data having complex hidden representation with nonlinear and non-spherical boundaries. This can be a crucial issue in real life applications where data structuring generally have complex separations between groups. The ability of the method to look for nonlinear separations becomes a necessary requirement to detect relevant overlapping clusters.

Another challenge that motivates overlapping clustering is the ability to regulate the sizes of overlaps between clusters. Although the method should reveal the clustering that best fit to the data, existing overlapping methods produce clusterings without possibility of control of the size of the overlaps. The application of these methods in practise usually leads to clusters with large overlapping boundaries. In fact, clusters with too large overlaps are not appropriate for most of the target applications while obtained clusters become not well separated. Ideally, the size of overlaps should be controlled, depending on the requirements of the application, and should be used as a parameter of the pattern recognition process.

Moreover, many of existing methods assumed that data have a vectorial description. However, in many applications of clustering, data are described by other structures such as sequences, trees or graphs. For example, in "Document Clustering" application, the vectorial representation of textual data ignores the correlation between adjacent words and leads to the loss of information regarding words positions. More effective representation of documents uses a sequential representation and group documents based on the shared subsequences. For such type of application and data representation, the overlapping clustering algorithm should be adapted to detect non-disjoint groups from these specific structures of data.

Contributions of the Thesis

Known all these shortcomings, the goal of this thesis is to design more efficient and perfective methods for overlapping clustering which address the above described limitations of prior works. The main contributions can be described as in the following.

The first contribution of this thesis put emphasis on the identification of overlapping clusters with nonlinear separations and non-spherical shapes. We propose (BenN'Cir and Essoussi 2011; BenN'Cir and Essoussi 2012b) two methods which use kernel functions to implicitly map data from an input space to a high dimensional feature space where separability of clusters becomes easier. First, we propose Kernel Overlapping K-means I (KOKMI) which is a centroid based method generalizing kernel k-means to produce overlapping clusters with nonlinear and non spherical shapes. Second, we propose Kernel Overlapping K-means II (KOKMII) which is a medoid based method improving KOKMI in terms of efficiency and complexity. Experiments are performed on artificial, non-spherical and real overlapping datasets in order to evaluate KOKMI and KOKMII against the existing methods.

The second contribution put emphasis on the identification of overlapping clusters with control of overlapping boundaries between clusters. We propose a generic clustering model that generalizes k-means to detect overlapping clusters with overlap regulation. Different instantiations (BenN'Cir et al. 2013a; BenN'Cir et al. 2013b; BenN'Cir et al. 2014a; BenN'Cir et al. 2014b) of the proposed model are defined which produce different layouts for the overlapping boundaries between clusters. These instantiations offer for users the possibility to restrict, to regulate or to auto-adjust the sizes of overlaps according their prior knowledge on data. Proposed instantiations are confronted with existing methods for evaluating both the quality of clustering and the size of overlaps known the actual overlap in different real multi-labeled benchmarks.

The third contribution of this thesis is an application of overlapping clustering on text document. We propose (BenN'Cir et al. 2013) a non supervised learning method, referred to as KOKM based WSK, which considers each textual document as a sequence of words and avoids the limitation of the Vector Space Model representation of text. The proposed method is based on the Word Sequence kernel (WSK) to evaluate similarity measure between documents. It has the advantages that the correlation between adjacent words in text and the possibility of document to belong to more than one cluster are not ignored. Experiments are performed in several text collections and using different text representation techniques to check its effectiveness.

The fourth contribution consists in proposing a software for overlapping clustering. This software is an extension of the "Weka" tool in which most of the clustering methods presented in this thesis and other existing ones were developed with the aim of making them accessible and useful to researchers and developers working on a diverse range of overlapping clustering. The proposed software is a Graphical User Interface (GUI) accessible as a java application or as a web based application (Applet) giving the possibility to look for overlapping clusters on different types of data such as vector, structured and unstructured data. It can be integrated in any java application and can be executed on different operating system such as Windows or Linux.

Organization of the Thesis

The rest of this thesis is organized as in the following:

- **Chapter 1** reviews existing methods in the literature which are able to produce a non-disjoint partitioning of data. A detailed description and comparison of overlapping methods based on the partitional approach are given. Furthermore, a comparison of existing techniques which are used to evaluate the quality of overlapping partitionings is introduced.

- **Chapter 2** describes the proposed methods KOKMI and KOKMII which are able to perform a non-disjoint partitioning of data with both linear and nonlinear separations between clusters.

- **Chapter 3** presents the proposed generic overlapping clustering model with overlap regulation which supports different instantiations able to reduce, to parameterize or to auto-adjust the sizes of overlaps.

- **Chapter 4** introduces the proposed KOKM based WSK method which is able to look for non-disjoint partitioning from textual documents by measuring similarity between documents based on the shared sequences of text.
- **Chapter 5** presents the proposed Weka4OC software which supports different methods for preprocessing, overlapping clustering and visualizing data.
- **Conclusion** summarizes the main contributions of this thesis and discusses ways for extensions of the proposed works which could be considered in future researches.

Chapter 1

Overlapping clustering

Contents

1.1 Introduction

This chapter focuses on fundamental concepts of clustering and overlapping clustering. First, we describe cluster analysis as an unsupervised learning method requiring data preparation and similarity measure definition. Then, we introduce the overlapping clustering research domain and we give a literature review of existing overlapping methods, especially those based on k-means and k-medoids algorithms. Finally, we give a literature review of existing techniques for the evaluation of non-disjoint partitioning.

1.2 Clustering: a non supervised learning

Given a training data, the aim of clustering, also referred to as cluster analysis or learning, is to detect hidden structures in the observed data. Usually, a clear distinction is made between learning problems that are supervised (Han and Kamber 2000), also referred to as classification, and those that are unsupervised, referred to as clustering. The first deals with only labeled data while the latter deals with only unlabeled data (Duda et al. 2001). In many practical learning domains, there is a large supply of unlabeled data but limited labeled data. This fact makes clustering more difficult than classification. However, there is a growing interest in a hybrid setting, called semi-supervised learning (Chapelle et al. 2006) where the unlabeled data, instead of being discarded, are also used in the learning process.

Practically, the goal of clustering is to discover the natural groupings of a set of observations, also referred to as data points, instances or objects. Han and Kamber (2000) defined *cluster analysis* as “The process of grouping a set of physical or abstract objects into classes of similar objects” and define a cluster as a “collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters". A formal definition of clustering can be stated as follows: given a description of n data over p variables, clustering aims to find k groups based on a measure of similarity such that similarities between data in the same group are high while similarities between data in different groups are low. An example of clustering's results are shown in Figure. 1.1 where the used clustering algorithm automatically discovers three natural clusters in the unlabeled data and summarizes each group by a cluster's representative.

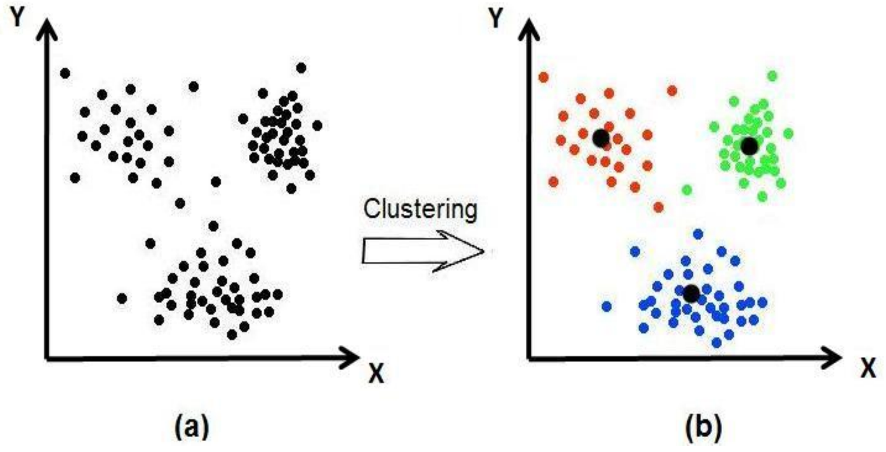


Figure 1.1: Identification of natural grouping in unlabeled data using clustering: (a) unlabeled data (b) 3 detected clusters (Han and Kamber 2000)

1.2.1 Data description and representation

Usually, data that often occur in cluster analysis are represented by one of these two structures: Data matrix and Proximity matrix.

Data matrix, also referred to as object-by-variable structure or vectorial structure (Rokach 2010), represents n objects by p variables which are also called measurements or attributes. Each row of the matrix represents a single observation described by p variables. For example, if data describe demographic information about a population, each row represents demographic data related to one person such as age, height, weight, gender, profession and so on. However, if data is a set of documents, each row represents a single document and variables are the words in all documents.

$$M(n,p) = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} \tag{1.1}$$

Proximity matrix, also referred to as object-by-object structure (Rokach 2010), stores a collection of similarities or dissimilarities that are available for all pairs of n objects. It is often represented by an n-by-n matrix:

$$M(n,n) = \begin{bmatrix} 0 & D(1,2) & \cdots & D(1,n) \\ D(2,1) & 0 & \cdots & D(2,n) \\ \vdots & \vdots & \ddots & \vdots \\ D(n,1) & D(n,2) & \cdots & 0 \end{bmatrix} \tag{1.2}$$

where $D(i,j)$ indicates the proximity of an observation x_i to an observation x_j.

The use of object-by-variable or object-by-object structures is determined based on the requirement of the learning method to have one of these two structures as input and based on the possibility of formatting data as proximity matrix or as data by variables matrix. The variables describing the data can have different types which can be classified into *qualitative* and *quantitative* data, and within each of these groups, data can be further categorized as shown below.

Qualitative data

Qualitative data are "categorical" data summarized using percentages or proportions. This type of data can be classified (Han and Kamber 2000) as *binary*, *nominal* and *ordinal*. Binary data are described only by two categories, e.g., male or female for the sex of a person. Nominal data generalizes the binary description and supports different categories explaining the data, e.g., black, white, red, yellow for describing the color of each car. However, ordinal data are inherently categorical in nature, but have a logical order of the categories. Example of ordinal data are the description of the class level of a student.

Quantitative data

Quantitative data are numerical descriptions of the data which can be classified (Han and Kamber 2000) as *discrete* or *continuous*. Discrete data only include integer values describing the count or quantities of a given finite variable. Example of discrete data are the number of products per box. However, continuous data are a decimal description of an infinite range of possible values. Examples of continuous data are the weight or the height of a person.

In fact, the type of data variables is an important characteristic to estimate whether two observations are similar or dissimilar. Since clustering requires to determine this relation for quantitative and qualitative data, many proximity measures are proposed in the literature.

1.2.2 Similarity and dissimilarity measures

Proximity measures can be classified into two main types: dissimilarity (distance) measures and similarity measures.

Dissimilarity measures

Methods based on distance measure estimate the degree of dissimilarity between any pair of observations. Given a set of observations $X = \{x_1, ..., x_n\}$, a distance measure $D : X \times X \rightarrow \mathbb{R}_0^+$ is a valid distance (Rokach 2010) if it is non-negative, symmetric and obtains its minimum value, usually zero, in case of identical observations:

1. $D(x_i, x_j) > 0 \quad \forall x_i, x_j \in X$(Non-negativity)
2. $D(x_i, x_j) = S(x_j, x_i) \quad \forall x_i, x_j \in X$(Symmetry)
3. $D(x_i, x_j) = 0 \Leftrightarrow x_i = x_j \quad \forall x_i, x_j \in X$ (Identity)

The distance measure is called a *metric* distance measure (Han and Kamber 2000) if it also satisfies the triangle inequality:

4. $D(x_i, x_j) + D(x_j, x_z) \geq D(x_i, x_z) \forall x_i, x_j \quad and \quad x_z \in X.$

If a mapping satisfies all the described properties except the third one, it is called a *pseudometric* (Rokach 2010) distance measure. However, if a mapping satisfies the following stronger version of the triangle inequality:

5. $max(S(x_i, x_j), S(x_j, x_z)) \geq S(x_i, x_z) \forall x_i, x_j and x_z \in X,$

it is called *ultrametric.*

Based on the type of data, many distance measures have been proposed. In the case of quantitative data, given two p-dimensional data objects $x_i = (x_{i1}, x_{i2}, ..., x_{ip})$ and $x_j = (x_{j1}, x_{j2}, ..., x_{jp})$, the distance between the two data objects can be calculated using the *Minkowski* measure (Han and Kamber 2000):

$$D(x_i, x_j) = (|x_{i1} - x_{j1}|^g + |x_{i2} - x_{j2}|^g + ... + |x_{ip} - x_{jp}|^g)^{1/g}, \tag{1.3}$$

which generalizes the *Euclidean* distance when $g = 2$, the *Manhattan* distance when $g = 1$, and the *Chebyshev* distance when $g \to \infty$.

In the case of binary attributes, the distance between objects may be calculated based on a contingency table. The *Jaccard* coefficient can be used to evaluate the distance between two binary observation having binary attributes:

$$D(x_i, x_j) = \frac{r + s}{q + r + s}, \tag{1.4}$$

where q the number of attributes that equal to 1 for both objects while s and r the number of attributes that are unequal for both objects. In the case of nominal attributes, the *matching* technique can be used to evaluate the distance between two nominal data:

$$D(x_i, x_j) = \frac{p - m}{p}, \tag{1.5}$$

where p the total number of attributes and m the number of matches. Another technique for binary attributes consists in creating a binary attribute for each value of each nominal attribute and computing the binary dissimilarity. If attributes are ordinal, the evaluation of distances between data can be realized using quantitative distance measures after performing a mapping of data into a range of $[0, 1]$.

Similarity measures

An alternative measure to that of distance is the similarity measure. Methods based on similarity measure estimate the degree of similarity $S(x_i, x_j)$, rather than dissimilarity, between any pair of observations. A valid similarity measure should satisfy the property of symmetry and should have the largest value for identical vectors (Duda et al. 2001). A similarity measure where the target range is $[0, 1]$ is called a *dichotomous* similarity measure (Rokach 2010). The *Cosine* measure can be used to evaluate relativeness of two data objects, represented as vectors, based on their angles:

$$S(x_i, x_j) = \frac{x_i^t.x_j}{||x_i||.||x_j||}. \tag{1.6}$$

Other examples of similarity measure are the Pearson Correlation measure described by:

$$S(x_i, x_j) = \frac{(x_i - \overline{x_i})^t.(x_j - \overline{x_j})}{||x_i - \overline{x_i}||.||x_j - \overline{x_j}||} \tag{1.7}$$

where $\overline{x_j}$ denotes the average feature value of x over all dimensions and the Dice Coefficient Measure described by:

$$S(x_i, x_j) = \frac{2.x_i^t.x_j}{||x_i||^2 + ||x_j||^2}. \tag{1.8}$$

We showed in this section that performing a clustering process requires to define a similarity or a dissimilarity metric, based on type of variables describing the data, to measure the relatedness between observations. However, in many applications of clustering the data to be clustered cannot be described by a vectorial structure which requires to define more advanced similarities or dissimilarities in order to take into account some specificities of data like semantics or biological aspects. An example of these similarities are String Subsequence Kernel (SSK) (Lodhi et al. 2001) and Word Sequence Kernel (WSK) (Cancedda et al. 2003) which will be described in Chapter 4 when performing clustering of textual documents.

1.2.3 Clustering methods

There are more than thousands of clustering methods that exist in the literature. Commonly used methods can be classified according to the fundamental concepts on which clustering methods are based. Thus, clustering methods can be classified into the following categories (Han and Kamber 2000): hierarchical, partitional, density-based and grid-based.

The category of *hierarchical methods* aims to produce an hierarchy of clusters, called dendrogram, which shows how the clusters are related. By cutting the dendrogram at a desired level, a clustering of the observations into disjoint groups is obtained. A hierarchical method can be classified as being either agglomerative or divisive, based on how the hierarchical decomposition is formed. Agglomerative methods, also called bottom-up, start with one observation forming one group and proceed successively by merging smaller clusters into larger ones until all the groups are merged into one or a termination condition holds. However, contrary to the agglomerative methods the divisive methods, also called bottom-down, start with only one cluster containing all the data and proceed successively by splitting larger clusters into smaller ones until each observation forming one group or a termination condition holds. Examples of hierarchical methods are BIRCH (Zhang et al. 1996), CURE (Guha et al. 1998), ROCK (Guha et al. 2000), Chameleon (Karypis et al. 1999), etc. The main advantage of hierarchical methods is the visualization of structures in the data at multiple levels of granularity. However, they suffer from the fact that once a step of merge or split is done, it can not be undone in order to improve the partitioning.

The category of *partitional methods* attempts to decompose the dataset into a set of k disjoint clusters where the obtained partitioning optimizes a given criterion function. The criterion function is optimized iteratively and emphasizes the local or the global structure of the data. Partitioning methods create an initial partitioning and then use an iterative relocation technique that moves observations from one cluster to another in order to optimize the criterion function. Partitioning based methods can be subdivided into 3 sub categories which are respectively *probabilistic*, *centroid based* and *medoid based* methods. Probabilistic methods assume that data come from a mixture of several populations whose distributions and priors should be estimated. Centroid based methods, instead of an exhaustive enumeration of all the possible partitions that achieve the global optimality of the criterion function, obtained partitions are represented by the mean value of observations assigned to this partition. Similarly, k-medoid based methods use the same heuristic to prohibit an exhaustive enumeration of all the possible partitions, except that each partition is represented by one of the observation located near the center of cluster. Examples of partitioning methods are EM (Dempster et al. 1977), K-means (MacQueen 1967), PAM (Kaufman and Rousseeuw 2008), CLARANS (Ng and Han 2002), etc.

The category of *density-based methods* aims to detect high density and low density regions in the data where high density regions represent clusters and low density regions represent outliers. Contrary to most of clustering methods which are based on the distance between observations, density based methods are based on the notion of density. Clusters are built as long as the density, which corresponds to the number of observations, in the "neighborhood" exceeds some threshold. Therefore, for each observation within a given cluster, the neighborhood of a given radius has to contain at least a minimum number of observations. Examples of density based algorithms are DBSCAN (Density Based Spatial Clustering of Applications with Noise) (Ester et al. 1996), OPTICS (Ordering Points To Identify the Clustering Structure) (Ankerst et al. 1999), DENCLUE (DENsity-based CLUstEring) (Hinneburg and Keim 1998; Hinneburg and Gabriel 2007), MCLUST (Fraley and Raftery 2003), etc. The main advantage of this approach consists in its ability to detect clusters with arbitrary shapes while distance based methods can detect only spherical-shaped clusters.

The category of *grid-based methods* is mainly proposed for very large datasets. The grid-based clustering methods differ from the conventional clustering methods since they deal with the value space that surrounds the data points and not data points themselves. Their main characteristic is that they transform the space into a finite number of cells that form a grid structure on which all of the operations for clustering are performed. In general, a typical grid-based clustering method consists in creating the grid structure, calculating the cell density for each cell, sorting the cells according to their densities, identifying cluster centers and finally traversal of neighbor cells. The main advantage of this category of methods is its fast processing time. Examples of grid based methods are BANG-clustering (Schikuta and Erhart 2002), WaveCluster (Sheikholeslami et al. 2000), STING (STatistical INformation Grid-based method) (Wang et al. 1997), CLIQUE (Agrawal et al. 1998), MAFIA (Goil et al. 1999), etc.

1.2.4 An example of clustering methods: K-means

K-means (MacQueen 1967), also referred to as c-means, is a centroid based method mostly used for grouping data due to its simplicity and its linear complexity. Given a set of observations $X = \{x_i\}_{i=1}^N$ with $x_i \in \mathbb{R}^d$ and N the number of observations, the aim of k-means is to find a partition matrix $\Pi = \{\pi_c\}_{c=1}^k$ into k clusters that minimizes the within groups sum of squared errors. The minimization of the squared error J is performed iteratively. This process is formulated as optimizing the resolution of the following problem:

$$\begin{aligned}
&Minimize \qquad && J(\Pi, C) = \sum_{c=1}^{k}\sum_{i=1}^{N} \pi_{ic}.||x_i - m_c||^2 \\
&subject \quad to && \sum_{c=1}^{k} \pi_{ic} = 1, \forall i \in \{1,..,N\} \\
& && \pi_{ic} \in \{1,0\}, \forall i \in \{1,..,N\} \quad and \quad c \in \{1,..,k\},
\end{aligned} \tag{1.9}$$

where π_{ic} is a binary variable indicating that observation x_i belongs to cluster c and m_c represents the representative of cluster c. The minimization of the objective function J can be solved by iteratively solving the following two subproblems:

- subproblem $P1$: Fix $C = \hat{C}$ and minimize the objective function $J(\Pi, \hat{C})$.
- subproblem $P2$: Fix $\Pi = \hat{\Pi}$ and minimize the objective function $J(\hat{\Pi}, C)$.

The first subproblem $P1$ can be solved by:

$$\begin{aligned} \pi_{ic} &= 1 \qquad if ||x_i - m_c||^2 \leq ||x_i - m_l||^2, \forall l \in \{1, .., k\} \\ \pi_{ic} &= 0 \qquad\qquad \forall l \neq c. \end{aligned} \tag{1.10}$$

The second subproblem $P2$ can be solved by:

$$m_c = \frac{\sum_{i=1}^{N} \pi_{ic}.x_i}{\sum_{i=1}^{N} \pi_{ic}}. \tag{1.11}$$

The k-means algorithm iterates two main steps: update of clusters' memberships and update of centroids. It converges to a local minimum value after a finite number of iterations and the value of the objective function J is strictly decreasing (Selim and Ismail 1984). An example of the different steps of k-means in a two dimensional data are schematized in Figure 1.2. The main algorithm of k-means is described by Algorithm 1.

Algorithm 1 *k-means* $(X, \varepsilon, k) \to \Pi$

```
INPUT $X$: a dataset described over $\mathbb{R}^d$.
    $\varepsilon$: minimal improvement in the objective function.
    $k$: number of clusters.
OUTPUT $\Pi$: assignment of observations over $k$ clusters.
 1: Initialize representative of clusters with random clusters prototypes $C_0$, solve $J(\Pi, C_0)$
    to obtain $\Pi_0$ in iteration 0.
 2: Let $\hat{\Pi} = \Pi_t$ and solve $J(\hat{\Pi}, C)$ to obtain $C_{t+1}$. If $J(\hat{\Pi}, C_{t+1}) - J(\hat{\Pi}, C_t) \leq \varepsilon$ then
    return $\hat{\Pi}, C_{t+1}$ and Stop, Else go to step 3.
 3: Let $\hat{C} = C_{t+1}$ and solve $J(\Pi, \hat{C})$ to obtain $\Pi_{t+1}$.If $J(\Pi_{t+1}, \hat{C}) - J(\Pi_t, \hat{C}) \leq \varepsilon$ then
    return $\Pi, \hat{C}$ and Stop, Else go to step 2.
```

K-means algorithm has the following characteristics: it has a linear computational complexity, it needs to specify the number of clusters in advance, it often terminates at a

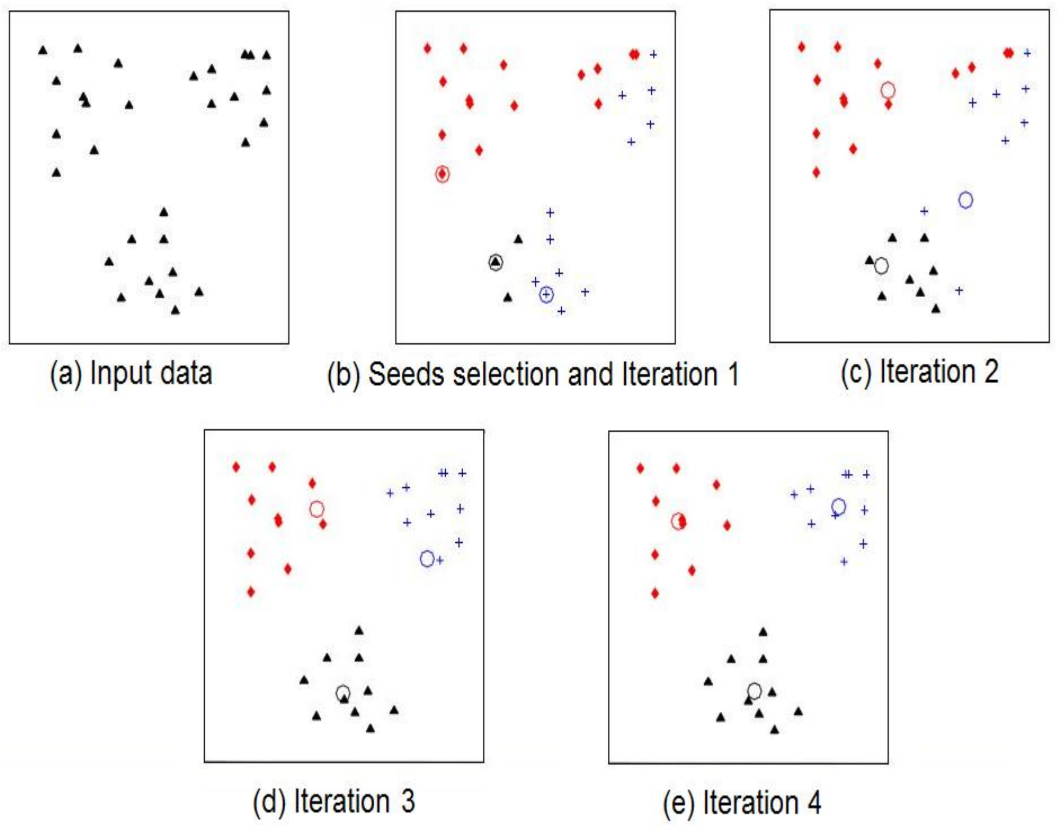


Figure 1.2: The different steps of k-means using 3 clusters in a 2 dimensional data (Jain 2010)

local optimum, it works only on numeric values, it is unable to handle outliers and it is unable to build non spherical shapes (Anderberg 1973).

1.3 Overview of overlapping clustering

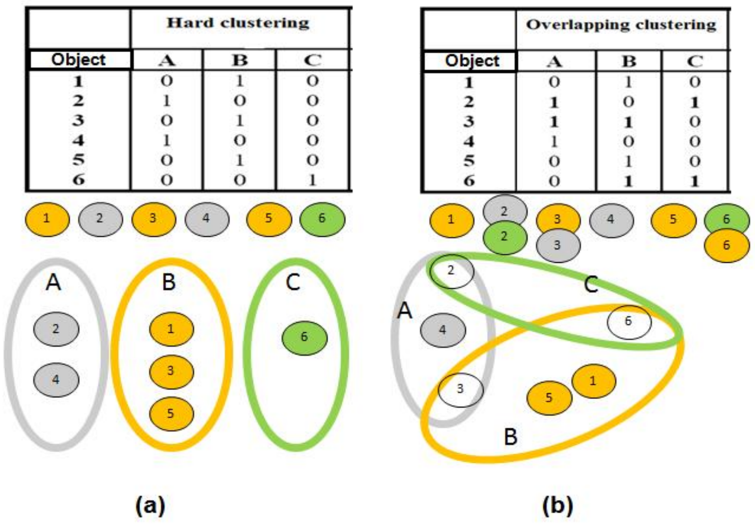
Although that thousand of clustering methods exist in the literature, many active challenges motivate researchers to propose more perfective and efficient clustering methods. The use of clustering in real life applications involve many typical requirements to make an efficient analysis of the data. For example, a reasonable runtime of clustering methods in large datasets and their ability to use a limited core memory become especially important. In addition, the ability of the method to look for clusters with nonlinear separations is an important characteristic while data structuring usually have complex shapes. Other researches are concerned by designing clustering methods able to perform partitioning in data having mixed types of attributes or building robust partitioning in the presence of outliers.

The works presented in this thesis are motivated by another important challenge in clustering that motivates several applications. We deal in this thesis with the challenge of identifying non-disjoint partitioning from unlabeled data. Most of existing clustering methods assume that each data observation belongs to one and only one cluster leading to k disjoint clusters explaining the data. However, in many applications the data being modeled can have a much richer and more complex hidden representation where observations actually belong to multiple clusters. Designing clustering methods able to perform non-disjoint partitioning of data would be benefic for several applications requiring such type of clustering.

1.3.1 Overlapping clustering *vs* hard clustering

Traditional learning methods, referred to as *hard* or *strict* clustering methods, ignore the possibility that an observation can belong to more than one cluster and lead to k exclusive clusters representing the data. Although this approach has been successfully applied in unsupervised learning, there are many situations in which a richer model is needed for representing the data. For example, in social network analysis, community extraction algorithms need to detect overlapping clusters where an actor can belong to multiple communities (Tang and Liu 2009; Wang et al. 2010; Fellows et al. 2011). In video classification, overlapping clustering is a necessary requirement where videos have potentially multiple genres (Snoek et al. 2006). In emotion detection, overlapping clustering methods need to detect different emotions for a specific piece of music (Wieczorkowska et al. 2006). In text clustering, learning methods should be able to group document, which discuss more than one topic, into several groups (Gil-García and Pons-Porrata 2010; Pérez-Suárez et al. 2013), etc. The corresponding research domain has been referred to as *overlapping clustering.*

Figure 1.3 shows an example of obtained partitioning using both hard clustering and overlapping clustering. In hard clustering, every observation belongs to exactly one cluster. However, in overlapping clustering all constraints on the memberships are removed and any observation can belong to one or several clusters using binary membership coefficient. For example, the observation 2 in Figure 1.3.(b) is assigned to both group A and C. Given that many real life applications require assigning data to multiple clusters, overlapping clustering has been studied through various approaches while it offers a richer model to fit existing structures in data.



	Hard clustering		
Object	A	B	C
1	0	1	0
2	1	0	0
3	0	1	0
4	1	0	0
5	0	1	0
6	0	0	1

	Overlapping clustering		
Object	A	B	C
1	0	1	0
2	1	0	1
3	1	1	0
4	1	0	0
5	0	1	0
6	0	1	1

Figure 1.3: Partitioning using hard and overlapping clustering

1.3.2 Classification of overlapping clustering methods

In this section, we propose a classification of existing overlapping methods based on the conceptual approach to build non-disjoint partitioning of data. Figure 1.4 shows a classification tree of these methods where the depth of the tree represents the progression in time and the width of the tree represents the different categories and subcategories. We classify the existing methods into 6 categories which are respectively: hierarchical, graphical, generative, partitional, correlation and topological. We detail in the following the main characteristics of each category.

The overlapping variants of hierarchies aim to reduce the discrepancy between the original dissimilarities over the considered dataset and the ones induced by the hierarchical structure. Although the flexibility in visualization offered by hierarchical methods, they still too restrictive in overlaps while they do not study all the possible combinations of clusters for each observation. Examples of these methods are the pyramids (Diday 1984) and more generally the weak-hierarchies (Bertrand and Janowitz 2003).

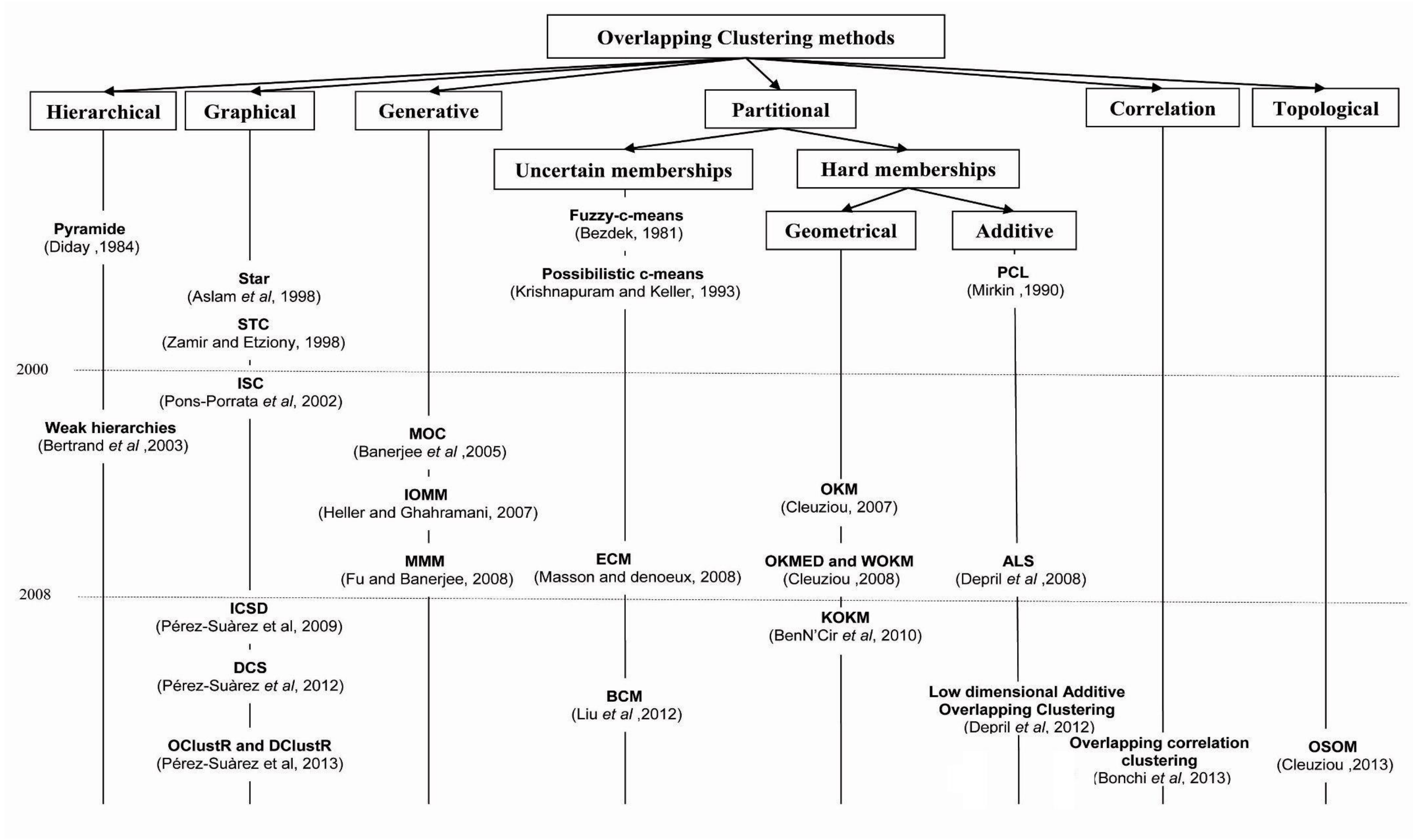

Overlapping Clustering methods
Hierarchical
Graphical
Generative
Partitional
Correlation
Topological
Uncertain memberships
Hard memberships
Geometrical
Additive
Pyramide (Diday ,1984)
Star (Aslam et al, 1998)
STC (Zamir and Etziony, 1998)
2000
ISC (Pons-Porrata et al, 2002)
Weak hierarchies (Bertrand et al ,2003)
MOC (Banerjee et al ,2005)
IOMM (Heller and Ghahramani, 2007)
MMM (Fu and Banerjee, 2008)
Fuzzy-c-means (Bezdek, 1981)
Possibilistic c-means (Krishnapuram and Keller, 1993)
ECM (Masson and denoeux, 2008)
OKM (Cleuziou, 2007)
OKMED and WOKM (Cleuziou ,2008)
PCL (Mirkin ,1990)
ALS (Depril et al ,2008)
2008
ICSD (Pérez-Suàrez et al, 2009)
DCS (Pérez-Suàrez et al, 2012)
OClustR and DClustR (Pérez-Suàrez et al, 2013)
BCM (Liu et al ,2012)
KOKM (BenN'Cir et al, 2010)
Low dimensional Additive Overlapping Clustering (Depril et al, 2012)
Overlapping correlation clustering (Bonchi et al, 2013)
OSOM (Cleuziou ,2013)

Figure 1.4: Classification of overlapping clustering methods based on their conceptual approach to build non-disjoint partitioning of data.

Overlapping methods based on graph are mostly used in the context of community detection in complex networks (Baumes et al. 2005; Gregory 2007; Zhang et al. 2007a; Davis and Carley 2008; Gregory 2008; Goldberg et al. 2010; Fellows et al. 2011; Magdon-Ismail and Purnell 2011; Wang and Fleury 2011). For this category of methods, a network is represented as a graph, where vertices are the studied observations and edges are links between the observations. All these graph-based methods use a greedy heuristic for covering the similarity graph. The difference between them consists in defining the criterion for ordering and selecting the sub-graphs. The main shortcoming of these methods is the computational complexity which is usually exponential and could be reduced to $O(N^2)$ as the case for OClustR (Overlapping Clustering based on Relevance) (Pérez-Suárez et al. 2013).

Overlapping clustering methods using generative mixture models have been proposed (Banerjee et al. 2005; Heller and Ghahramani 2007; Fu and Banerjee 2008) as extensions of the EM algorithm (Dempster et al. 1977). These models are supported by biological processes; they hypothesize that each data is the result of a mixture of distributions . The mixture can be additive (Banerjee et al. 2005) or multiplicative (Heller and Ghahramani 2007; Fu and Banerjee 2008) and the probabilistic framework makes possible to use not only gaussian components but any exponential family distributions. On the other hand, generative models are not parameterizable and do not allow the user to control the requirements of the overlaps.

Other methods for overlapping clustering extend recent approaches to address the problem of overlapping clustering. For example, an extension of correlation clustering (Bonchi et al. 2011; Bonchi et al. 2013) and topological maps (Cleuziou 2013) have been recently proposed. Overlapping correlation clustering has been defined as an optimization problem that extends the framework of correlation clustering to allow overlaps by relaxing the function which measures the similarity of assigned set of labels, instead of one single label, for each data object. For topological maps, an extension of the Self-Organizing-Maps (SOM) has been proposed by allowing for each data to be assigned to one or several neurons on the grid by searching for a subset of neurons winners rather than a single neuron. The main advantage of both correlation and topological methods consists in their ability to learn the right number of overlapping clusters.

Despite the use of all these approaches to build non-disjoint partitioning of data, the Partitional approach remains the most commonly used while several methods are based on. This category of methods consists either in modifying the clusters resulting from a standard method into overlapping clusters or in proposing new objective criteria to model overlaps. This thesis put emphasis on overlapping partitional clustering methods, specifically those extending and generalizing k-means and K-medoid methods (MacQueen 1967; Jain 2010). In this way, we present in the next section a description of these methods.

1.4 Literature review of overlapping partitional methods

Several works have focused on partitional clustering in order to build overlapping clusters leading to two main categories of methods: the category of *uncertain memberships* and the category of *hard memberships*. We denote by uncertain memberships the solutions which model clusters' memberships for each data object as uncertainty function using fuzzy, possibilistic or evidential frameworks. The uncertainty function measures the degree of belonging of each data to the underlying group. However, we denote by *hard memberships* the solutions which lead to *hard* and *overlapping* partitioning by considering a binary function to model clusters' memberships.

1.4.1 Uncertain memberships based-methods

Uncertain memberships based-methods consist either in extending results of uncertain methods into overlapping clusters, typically the extension of fuzzy-c-means (FCM) (Lingras and West 2004; Zhang et al. 2007b) and possiblistic c-means (PCM) (Krishnapuram and Keller 1993), or in proposing new objective criterion that takes into account the possibility of overlaps between clusters. The Evidential c-means (ECM) (Masson and Denoeux 2008) and the Belief c-means (BCM) (Liu et al. 2012) are two distinctive examples of such criteria where their optimization processes lead to generate overlapping clusters. All uncertain methods need a post-processing treatment to generate the final overlapping clusters.

We detail in the following the principal uncertain clustering methods which are able to produce non-disjoint partitioning. We consider for all the detailed methods a set of observations $X = \{x_i\}_{i=1}^{N}$ with $x_i \in \mathbb{R}^d$ and N the number of observations where the aim, of each method, is to find a non-disjoint partitioning matrix $\Pi = \{\pi_c\}_{c=1}^{k}$ into k clusters and a set $C = \{m_c\}_{c=1}^{k}$ of k clusters' representatives minimizing an objective criterion.

Fuzzy c-means (FCM) and Possiblistic c-means (PCM) methods

The FCM (Bezdek 1981) identifies clusters as fuzzy sets where the objective function J_{FCM} allows that an observation belongs to many clusters with a coefficient indicating membership degrees to all clusters in the [0,1] interval (0 stands for no membership and 1 for total membership). FCM is based on the minimization of the following function:

$$J_{FCM}(\Pi, C) = \sum_{c=1}^{k} \sum_{i=1}^{N} \pi_{ic}^{\beta}.||x_i - m_c||^2, \tag{1.12}$$

where Π is the fuzziness membership matrix that indicates the coefficient of closeness of an object to every cluster under the constraints:

$$\begin{gathered} \pi_{ic} \in [0..1] \;\; \forall i, \quad \forall c \\ \sum_{c=1}^{k} (\pi_{ic}) = 1, \forall i \\ \beta > 1. \end{gathered} \tag{1.13}$$

The parameter β controls the fuzziness of the memberships: for high values of β the algorithm tends to set all the memberships equals while for β tending to one it has the behavior of k-means algorithm with crisp memberships. The minimization of

Equation 1.12 is done iteratively using an alternating least square optimization of the two parameters Π and C. The optimal fuzziness membership matrix Π and the optimal clusters' representatives C are computed in each step as the following.

$$\pi^*_{ic} = \frac{1}{\sum_{l=1}^{k} \left(\frac{||x_i - m_c||^2}{||x_i - m_l||^2} \right)^{(\frac{1}{\beta-1})}} \tag{1.14}$$

$$m^*_c = \frac{\sum_{i=1}^{N} \pi^{\beta}_{ic} x_i}{\sum_{i=1}^{N} \pi^{\beta}_{ic}} \tag{1.15}$$

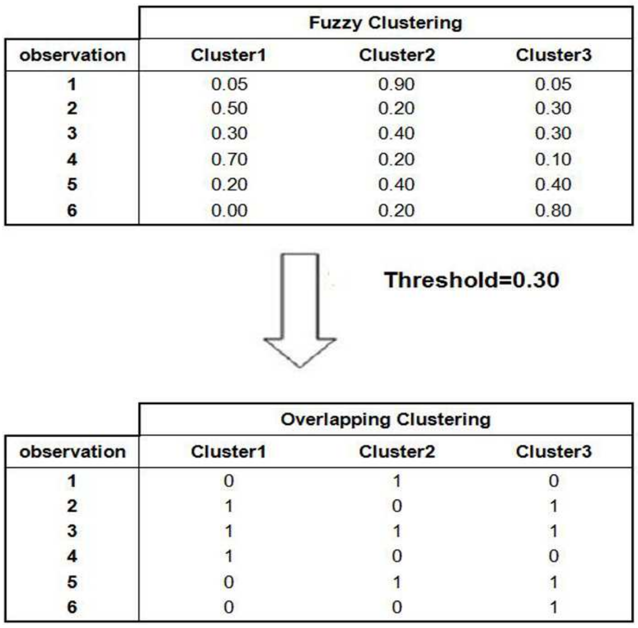
The extension of FCM to overlapping clustering can be done by fixing a threshold where all observations having memberships' degrees that exceed this threshold are assigned to the respective clusters. An example of this transformation is shown in Figure 1.5 where the obtained clusters' memberships are non-disjoints. We note that in some cases, when the fixed threshold is somewhat large, observations having all the memberships lower than this threshold will not be assigned to any cluster. Obtained clusters are usually much sensitive to the threshold's value.

In the same way that FCM, the PCM (Krishnapuram and Keller 1993) method is proposed to relax the constrained condition of the fuzzy partition ($\sum_{c=1}^{k}(\pi_{ic}) = 1$) in order to obtain a possiblistic type of memberships matrix. The objective function of PCM[1] is described by:

$$J_{PCM}(\Pi, C) = \sum_{c=1}^{k} \sum_{i=1}^{N} \pi^{\beta}_{ic} ||x_i - m_c||^2, \tag{1.16}$$

under the constraints:

[1]The original objective function of PCM takes into account the identification of outliers, whereas we give in this report a short structure of the objective function to facilitate the comparison of PCM with the other described methods.



	Fuzzy Clustering		
observation	Cluster1	Cluster2	Cluster3
1	0.05	0.90	0.05
2	0.50	0.20	0.30
3	0.30	0.40	0.30
4	0.70	0.20	0.10
5	0.20	0.40	0.40
6	0.00	0.20	0.80

Threshold=0.30

	Overlapping Clustering		
observation	Cluster1	Cluster2	Cluster3
1	0	1	0
2	1	0	1
3	1	1	1
4	1	0	0
5	0	1	1
6	0	0	1

Figure 1.5: Extension of the results of fuzzy clustering to obtain overlapping clustering using a threshold value equals to 0.3

$$\begin{aligned} &\pi_{ic} \in [0..1] \ \forall i, \quad \forall c \\ &\sum_{c=1}^{k} (\pi_{ic}) \quad \in [0, k] \ \forall \ i \\ &\beta > 1. \end{aligned} \tag{1.17}$$

Similar to FCM, the objective function J_{PCM} is minimized iteratively where memberships and clusters' representatives are updated as follows:

$$\pi_{ic}^{*} = \frac{1}{1 + (||x_i - m_c||^2)^{1/(\beta-1)}}, \tag{1.18}$$

$$m_c^{*} = \frac{\sum_{i=1}^{N} \pi_{ic}^{\beta} x_i}{\sum_{i=1}^{N} \pi_{ic}^{\beta}}. \tag{1.19}$$

Each observation can belong to several clusters with a possiblistic membership. The possiblistic memberships can be extended to overlapping ones by setting memberships with higher values to 1 and the other are replaced with 0.

Evidential c-means (ECM) and Belief c-means (BCM) methods

ECM (Masson and Denoeux 2008) is based on the concept of credal partition which is a general extension of fuzzy and possiblistic partitioning. As opposed to FCM and PCM, the ECM method evaluates all the possible combinations of clusters, denoted A_j, from the set of single clusters $\Omega = \{\omega_1, \ldots, \omega_k\}$ by allocating a mass of belief π_i within each possible combination.

Let the credal partition matrix $\Pi = (\pi_1, \ldots, \pi_N) \in R^{N \times 2^k}$ and the matrix $C = (m_1, \ldots, m_k)$ of clusters' representatives, ECM[2] is based on the minimization of the following objective function:

$$J_{ECM}(\Pi, C) = \sum_{i=1}^{N} \sum_{j/A_j \subseteq \Omega} c_j^{\alpha} \pi_{ij}^{\beta} \left\| x_i - \overline{m_j} \right\|^2 \tag{1.20}$$

under the constraint:

$$\sum_{j/A_j \subseteq \Omega} \pi_{ij} = 1 \;\forall i \in \{1, .., N\}\,, \tag{1.21}$$

where π_{ij} denotes the mass of belief for associating the observation x_i to the specific set A_j which can be either a single cluster or a combination of single clusters, c_j denotes the cardinality $|A_j|$ of each set which aims at penalizing the combination of clusters with high cardinality, α and β are two parameters used respectively to control the penalization

[2]The original objective function of ECM takes into account the identification of outliers by considering $\pi_{i\emptyset}$ a mass of belief to belong to any cluster, whereas we consider in this report that all combinations of clusters are tolerated except the empty set ($A_j \neq \emptyset$) in order to facilitate the comparison of ECM with the other described methods.

term c_j and the fuzziness degree π_{ij} and $\|x_i - \overline{m_j}\|^2$ is the distance between x_i and the combination of clusters' representatives $\overline{m_j}$ of the focal set A_j defined by:

$$\overline{m_j} = \frac{\sum\limits_{j/A_j \subseteq \Omega} m_j}{c_j}. \tag{1.22}$$

To minimize the objective function of ECM, an alternating optimization scheme is designed as in FCM where the update of the mass of belief π_{ij} and the clusters' representatives C_k is computed as described in Equations 1.23 and 1.24 .

$$\pi^*_{ij} = \frac{c_j^{-\alpha/(\beta-1)} \|x_i - \bar{m_j}\|^{-2/(\beta-1)}}{\sum\limits_{A_k} c_k^{-\alpha/(\beta-1)} \|x_i - \bar{m_k}\|^{-2/(\beta-1)}} \ \forall i \in \{1,..,N\} \ \forall j/A_j \subseteq \Omega, \tag{1.23}$$

$$C^*_{k\times d} = H^{-1}_{k\times k} B_{k\times d}, \tag{1.24}$$

where the elements B_{lq} of the matrix $B_{k\times d}$ for $l \in \{1,..,k\}$, $q \in \{1,..,d\}$ and the elements H_{lh} of the matrix $H_{k\times k}$ for $l, h \in \{1,..,k\}$ are defined respectively by:

$$B_{lq} = \sum_{i=1}^{N} x_{iq} \sum_{j/\omega_l \in A_j} c_j^{\alpha-1} \pi_{ij}^{\beta} \qquad H_{lh} = \sum_{i=1}^{N} \sum_{j/\omega_h, \omega_l \subseteq A_j} c_j^{\alpha-2} \pi_{ij}^{\beta}. \tag{1.25}$$

Similar to ECM, a more recent method, referred to as Belief c-means (BCM) (Liu et al. 2012), is also developed within the framework of belief function which is an extension of ECM that improves the quality of the credal partitions by assigning only relatively distant observations to the combination of clusters. This method evaluates the distance inter-prototypes before building the mass of believe π_{ij} related to each observation. The objective function optimized by BCM is described by:

$$J_{BCM}(\Pi, C) = \sum_{i=1}^{N} \sum_{j/A_j \subseteq \Omega, |A_j|=1} \pi_{ij}^{\beta} \|x_i - m_j\|^2 + \sum_{i=1}^{N} \sum_{j/A_j \subseteq \Omega, |A_j|>1} c_j^{\alpha} \pi_{ij}^{\beta} \widehat{d_{ij}}^2, \tag{1.26}$$

where $\widehat{d_{ij}}^2$ evaluates the distance x_i with respect to inter-distances of clusters' prototypes to which x_i belongs to:

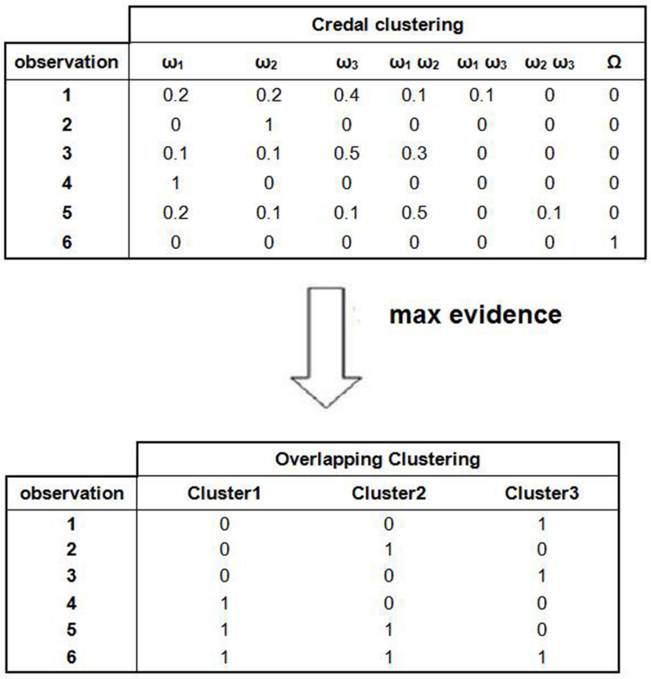
$$\widehat{d_{ij}}^2 = \frac{\sum_{k \in A_j} \|x_i - m_k\|^2 + \sum_{l,p \in A_j} \|m_l - m_p\|^2}{|A_j| + \gamma C^2_{|A_j|}}, \tag{1.27}$$

with γ the weighting factor of the distances among the clusters' prototypes and $C^2_{|A_j|} = \frac{|A_j|!}{2!(|A_j|-2)!}$ the number of possible combinations of pairs from the set of assignments A_j.

In fact, both ECM and BCM lead to credal partitioning of N data into 2^k possible combinations of k clusters. An example of credal partitioning is reported in Figure 1.6. We note that both possiblistic and fuzzy partitions can be recovered from the credal partition. A possiblistic partition can be obtained by computing from each m_i the plausibilities (possibilities) of the different clusters and a fuzzy partition can be obtained by calculating the pignistic probabilities of the different clusters from each m_i. The extension of both ECM and BCM to overlapping clustering, called hard credal partition by the authors, can be done by assigning each object x_i to the set of clusters A_j with the highest mass. Overlapping observations are those having highest mass for $|A_j| >= 2$. The process that leads to hard credal partitioning is called "Upper" (Masson and Denoeux 2008) approximation.

1.4.2 Hard memberships based-methods

The category of hard memberships based methods generalizes the strict k-means to look for optimal overlapping clusters. As opposed to fuzzy, possiblistic, and evidential clustering, these methods produce hard overlapping clusters and do not need any post processing treatment. Two kind of hard memberships based-methods have been proposed: *additive* and *geometrical* methods. We denote by *additive* the methods which hypothesize that overlaps result in the addition of the representatives of the related clusters. These methods group observations into overlapping clusters while minimizing the sum of distances between each observation and the *sum* of clusters' representatives to which the observation belongs to. In contrast, we denote by *geometrical* methods those formalizing overlaps



	Credal clustering						
observation	ω_1	ω_2	ω_3	$\omega_1\ \omega_2$	$\omega_1\ \omega_3$	$\omega_2\ \omega_3$	Ω
1	0.2	0.2	0.4	0.1	0.1	0	0
2	0	1	0	0	0	0	0
3	0.1	0.1	0.5	0.3	0	0	0
4	1	0	0	0	0	0	0
5	0.2	0.1	0.1	0.5	0	0.1	0
6	0	0	0	0	0	0	1

max evidence

	Overlapping Clustering		
observation	Cluster1	Cluster2	Cluster3
1	0	0	1
2	0	1	0
3	0	0	1
4	1	0	0
5	1	1	0
6	1	1	1

Figure 1.6: Example of the extension of credal partition containing 6 observations under 3 clusters: observations are assigned to the combination of clusters having the max mass of evidence.

as a barycenter on the related cluster representatives. This category of methods is based on a geometrical reasoning in the data space and groups observations into overlapping clusters while minimizing the sum of distances between each observation and the *average*, instead of the sum, of clusters' representatives to which the observation belongs to. Figure 1.7 shows an illustrative example of modeling overlaps using additive and geometrical models. Given a two dimensional dataset with two clusters' representatives $C1(2,2)$ and $C2(4,4)$ and a data point $X(2.5,3.5)$ to assign as described in Figure 1.7(a). Geometrical methods evaluate the distance $d(X,\overline{X})$, represented by the red line in Figure 1.7(c), between X and the average of clusters' representatives $\overline{X}$. However, additive methods evaluate the distance $d(X,\overline{X})$, described by the green line in Figure 1.7(d), between X and the sum of clusters' representatives $\overline{X}$. For both models, additive and geometrical, the data point X will be assigned to both clusters $C1$ and $C2$ if the following condition holds:

$$d(X, \overline{X}) < d(X, C1) \quad \& \quad d(X, \overline{X}) < d(X, C2).$$

Additive methods

Several additive methods are proposed in the literature such as Principal Cluster Analysis (PCL) (Mirkin 1987; Mirkin 1990), Alternating Least Square algorithms (ALS) (Depril et al. 2008) and Lowdimensional Additive Overlapping Clustering.

- **PCL method**

PCL (Mirkin 1990) introduces the possibility that an observation belongs to more than one cluster based on the Additive model by considering variable values of an observation equals to the sum of the clusters' representatives to which the observation belongs to. Given a dataset X, a model matrix $M = \Pi C$ is looked for to optimally approximate X. The matrix M can be estimated by minimizing the least squares loss function:

$$J_{PCL}(\Pi, C) = \| X - \Pi C \|_F^2 = \sum_{x_i \in X} \| x_i - \sum_{c \in \Pi_i} m_c \|^2, \tag{1.28}$$

where $\| \, . \, \|_F^2$ is the Frobenius norm, also called the Euclidean norm, of a matrix and m_c is the representative of cluster c.

To minimize the objective criterion (1.28), PCL proceeds by building clusters one by one from a dataset until achieving the expected number of clusters k. PCL builds the memberships of each cluster c independently from the memberships of the other clusters by minimizing the following criterion for each cluster c:

$$J_{PCL}^c = \sum_{x_i \in X} \pi_{ic} \| x_i - m_c \|^2 \, . \tag{1.29}$$

The minimization process starts with an empty cluster (i.e., with $\pi_{ic} = 0$, $\forall x_i \in X$) and sequentially add observations to it in a greedy way leading to the smallest value of

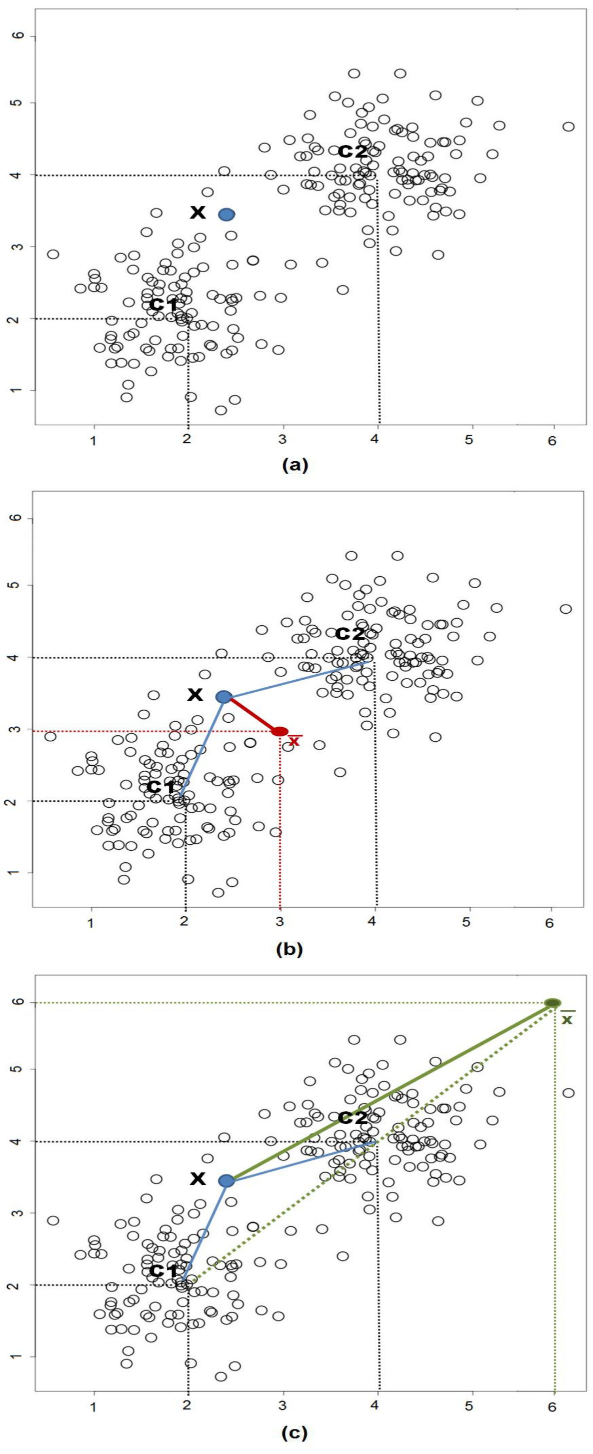


Figure 1.7: Geometrical *vs* Additive model for overlapping clustering: (a) Input data with 2 representatives C1(2,2) and C2(4,4) and a data point X(2.5,3.5) to assign (b) Geometrical model formalizing overlaps as the *average* of clusters' representatives (C) Additive model formalizing overlaps as the *sum* of clusters' representatives

criterion (1.29). For every observation that is considered for joining the cluster, a new representative m_c and a new value of J^c_{PCL} have to be calculated. The process is continued until there is no further decrease of J^c_{PCL}. The computation of clusters' representatives is also done locally for each cluster independently from the other clusters by:

$$m^*_c = \frac{\sum\limits_{x_i \in X} \pi_{ic} x_i}{\sum\limits_{x_i \in X} \pi_{ic}}. \tag{1.30}$$

The main characteristic of this method consists in its high computational complexity evaluated by $O(2^{N.k})$ which makes the method under-used in real life applications of overlapping clustering.

- **ALS method**

ALS (Depril et al. 2008) is based on the same objective criterion of PCL described in Equation 1.28. However, ALS proposes two other algorithms for minimizing this objective criterion referred to as ALS$_{lf1}$ and ALS$_{lf2}$. For both algorithms, the minimization of the objective criterion starts from an initial binary membership matrix Π_0. This membership matrix can be initialized using a Bernoulli distribution with parameter $\tau = 0.5$ or by computing the conditionally optimal memberships upon k randomly drawn representatives from the initial data. Then, ALS estimates the conditionally optimal representatives C upon Π; subsequently it estimates the conditionally optimal memberships Π upon C, and this process will be repeated until convergence.

The estimation of optimal memberships are computed separably for each observation x_i by enumerating all possible binary configurations that lead to decrease the objective criterion given the conditionally optimal representative and the conditionally optimal memberships for the other observations. The algorithm repeats this procedure for the next observation and so on. After a pass through all observations, the new value of the objective criterion computed with new memberships Π_1 is compared to the one computed with old memberships Π_0. The process stops when there is no further decrease in the objective criterion. We note that ALS$_{lf1}$ differs from ALS$_{lf2}$ in that for each membership update

Π_i the conditionally optimal representatives C are recalculated immediately, whereas in ALS$_{lf2}$ the representatives are only updated at the end of the membership updating step.

In the other side, the optimal representatives are updated equivalently for ALS$_{lf1}$ and ALS$_{lf2}$ based on the memberships matrix Π as follows:

$$C^* = (\Pi'\Pi)^{-1}\Pi'X. \tag{1.31}$$

In fact, we notice that ALS with its two variants explores all the possible 2^k combinations of clusters and takes the optimal one that leads to decrease the objective criterion. This characteristic makes this method highly time consuming when the number of clusters becomes large. The computational complexity of ALS$_{lf1}$ and ALS$_{lf2}$ are respectively evaluated by $O(N^3.2^k)$ and $O(N^2.2^k)$.

- **Low dimensional Additive Overlapping Clustering method**

The Low dimensional Additive Overlapping Clustering (Depril et al. 2012) extends ALS method by establishing an overlapping clustering of the observations and a dimensional reduction of the variables (or dimensions) simultaneously. This method is designed in order to perform relevant non-disjoint partitioning when data contains a high number of dimensions. Given a set of observations X described over d variables, the aim of this method is to find a recovery Π of k overlapping clusters and a matrix $\widetilde{C}$ of clusters' representatives described over $\widetilde{d} < d$. The low dimensional additive Overlapping Clustering is based on the same objective criterion used for ALS and PCL. The process of optimizing this objective criterion uses an alternating optimization procedure similar to that of ALS, except that optimal reduced clusters' representatives are computed by:

$$\widetilde{C}^* = (\Pi'\Pi)^{-1}\Pi'TT'X, \tag{1.32}$$

where columns of matrix T represent the $\widetilde{d}$ orthogonal eigenvectors of the product of matrixes $ZXX'Z$ with $Z = \Pi(\Pi'\Pi)^{-1}\Pi'$ denotes the orthogonal projector operator.

Geometrical methods

Many Geometrical methods are proposed in the literature such as Overlapping k-means (OKM) (Cleuziou 2008), Weighted Overlapping k-means (WOKM) (Cleuziou 2009) and Overlapping k-Medoid (OKMED) (Cleuziou 2009). We note that ECM and BCM, described with uncertain memberships methods, can be categorized with geometrical methods while they are based on a geometrical reasoning to model the different combinations of clusters.

- **OKM method**

The OKM (Cleuziou 2008) method is an extension of the k-means algorithm which allows observations to belong to one or different clusters. Given a set of N observations $X = \{x_i\}_{i=1}^N$ with $x_i \in \mathbb{R}^d$, OKM aims to find a recovery $\Pi = \{\pi_c\}_{c=1}^k$ of k overlapping clusters such that the following objective function is minimized:

$$J_{OKM}(\Pi, C) = \sum_{i=1}^{N} \|x_i - \overline{(x_i)}\|^2. \tag{1.33}$$

This objective function minimizes the sum of squared Euclidean distances between each observation x_i and its representatives $\overline{(x_i)}$ for all $x_i \in X$. The representative $\overline{(x_i)}$ is defined as the barycenter of clusters' representatives to which the observation x_i belongs to:

$$\overline{(x_i)} = \sum_{c \in \Pi_i} \frac{m_c}{|\Pi_i|}. \tag{1.34}$$

where Π_i is the set of clusters to which x_i belongs to and m_c is the representative of cluster c. The minimization of the objective function is performed by alternating two principal steps: computation of clusters' representatives (C) and the assignment of observations to one or several clusters (Π). The update of representatives is performed locally for each cluster as described in Equation 1.35.

$$m_c^* = \frac{\sum\limits_{x_i \in \pi_c} \frac{1}{|\Pi_i|^2} . \tilde{x_i}^c}{\sum\limits_{x_i \in \pi_c} \frac{1}{|\Pi_i|^2}}, \tag{1.35}$$

where $\tilde{x_i}^c = |\Pi_i|.x_i - (|\Pi_i| - 1).\overline{(x_i)_{\Pi \setminus c}}$ and π_c denotes the set of observations assigned to cluster c. For the multiple assignment step, OKM uses an heuristic to explore part of the combinatorial set of possible assignments. The heuristic consists, for each observation, in sorting clusters from closest to the farthest, then assigning the observation in the order defined while assignment minimizes the distance between the observation and its representative. OKM is characterized by a linear computational complexity evaluated by $O(N.k. \lg k)$.

- **WOKM method**

The WOKM (Cleuziou 2009) method is a generalization of both OKM and Weighted k-means (Chan et al. 2004) methods to detect overlapping clusters. The WOKM introduces a vector of local feature weighting λ_c, relative to each cluster c, which allows data to be assigned to a cluster as regards to a subset of attributes that are important for the cluster concerned. Given a set of N observations $X = \{x_i\}_{i=1}^N$ with $x_i \in \mathbb{R}^d$, WOKM aims to find a recovery $\Pi = \{\pi_c\}_{c=1}^k$ of k overlapping clusters such that the following objective function is minimized:

$$J_{WOKM}(\Pi, C) = \sum_{i=1}^{N} \sum_{v=1}^{d} \gamma_{i,v}^{\beta} \|x_{i,v} - \overline{(x_{i,v})}\|^2, \tag{1.36}$$

where d the number of features and γ_i a vector of weights relative to the representative $\overline{(x_i)}$ which is defined for each feature v as the following:

$$\gamma_{i,v} = \frac{\sum\limits_{c \in \Pi_i} \lambda_{c,v}}{|\Pi_i|}. \tag{1.37}$$

The representative $\overline{(x_{i,v})}$ is defined, for each feature v, as the weighted barycenter of clusters' representatives to which the observation x_i belongs to:

$$\overline{(x_{i,v})} = \frac{\sum\limits_{c \in \Pi_i} \lambda_{c,v}^{\beta}.m_{c,v}}{\sum\limits_{c \in \Pi_i} \lambda_{c,v}^{\beta}}. \tag{1.38}$$

The optimization of the objective criterion is performed by iterating three steps. The first step consists in assigning each data object to the nearest cluster while minimizing the local error $\sum_{v=1}^{d} \gamma_{i,v}^{\beta} \|x_{i,v} - \overline{(x_{i,v})}\|^2$. The second step consists in updating clusters' representatives using the following criterion:

$$m_{c,v}^{*} = \frac{\sum\limits_{x_i \in \pi_c} \frac{\lambda_{i,v}^{\beta}}{|\Pi_i|^2}.\tilde{x_i}^{c}}{\sum\limits_{x_i \in \pi_c} \frac{\lambda_{i,v}^{\beta}}{|\Pi_i|^2}}, \tag{1.39}$$

The third step concerns the update of the set of clusters weights $\{\lambda_c\}_{c=1}^{k}$ by using:

$$\lambda_{c,v} = \frac{(\sum\limits_{x_i \in \pi_c} \|x_{i,v} - m_{c,v}\|^2)^{1/(1-\beta)}}{\sum\limits_{u=1}^{d} (\sum\limits_{x_i \in \pi_c} \|x_{i,u} - m_{c,u}\|^2)^{1/(1-\beta)}}. \tag{1.40}$$

The computational complexity of WOKM stills linear, similar to OKM, evaluated by $O(N.k.\lg k)$.

- **OKMED method**

OKMED (Cleuziou 2009) extends the method Partitioning Around Medoid (PAM) (Kaufman and Rousseeuw 2008) for overlapping clustering. It consists in aggregating the data around representatives of the clusters denoted as medoids which are chosen among the data themselves. The objective criterion of OKMED is based on the optimization of the following objective criterion:

$$J_{OMED}(\Pi, C) = \sum_{x_i \in X} \|x_i - \overline{(\chi_i)}\|^2. \tag{1.41}$$

where $\overline{(\chi_i)}$ is defined as the data from X that minimizes the sum of the dissimilarities with all the medoids of the clusters where x_i belongs to:

$$\overline{(\chi_i)} = \arg\min_{x_j \in X} \sum_{m_c \in \Pi_i} \|x_j - m_c\|^2. \tag{1.42}$$

The optimization of the objective function of OKMED is realized using an alternating optimization between two steps: assignment of each data to its nearest medoid and updating of the medoid for each cluster. The update of medoids consist in searching among the set of data belonging to the cluster, the one that minimizes the sum of the distances with any other data into the cluster.

$$m_c^* = \arg\min_{x_i \in \pi_c}(x_i) \sum_{x_j \in \pi_c} \|x_j - \overline{(\chi_j)_{x_i}}\|^2, \tag{1.43}$$

where $\overline{(\chi_j)_{x_i}}$ denotes the representative of observation x_j computed by considering x_i the medoid of the cluster. The use of medoids as representatives of clusters makes OKMED more robust to outliers and offers the possibility to use any metric since it only requires a proximity matrix over the data. However, OKMED is characterized by a high computational complexity evaluated by $O(N^3.k)$.

1.4.3 Illustrative example of the different steps of OKM

We give in the following an illustrative example of the different iterations and steps of OKM method with two clusters in a small dataset containing 6 observations described by two dimensions $D1$ and $D2$ as reported in Table 1.1.

Iteration 1 of OKM

At the first iteration, the OKM method requires to initialize random clusters' representatives. We consider that observations Y and W are the representative of cluster 1 and

Table 1.1: Description of the different observations in the dataset

observation	$D1$	$D2$
X	1	2
Y	1	1
Z	2	2
W	2	1
N	3	2
M	3	1

Cluster 2 respectively described by $C1(1,1)$ and $C2(2,1)$. The next step is the assignment step which assigns each observation to cluster $C1$, $C2$ or $C1 \cap C2$. For each observation x_i, OKM orders clusters nearest to farthest and then assigns x_i to the scheduled clusters while the local error $J_i(A_i) = (x_i - \overline{x_i})^2$ decreases with A_i the set of assignments. We give in the following the numeric application of the assignment of observation Z to the different clusters:

1. scheduling clusters respectively to Z: the distance $d(Z, C1)^2 = 2$ and the distance $d(Z, C2)^2 = 1$. The scheduling is $C2$ then $C1$.
2. assignment of Z to the nearest cluster, which is $C2$, and evaluation of the local error $J_Z(C2) = (Z - \overline{Z})^2 = (Z - C2)^2 = (\sqrt{(2-2)^2 + (2-1)^2})^2 = 1$.
3. assignment of Z to the next nearest cluster, which is $C1$, and evaluation of the local error $J_Z(C1, C2) = (Z - \overline{Z})^2 = (Z - \frac{(C1+C2)}{2})^2 = (\sqrt{(2-1.5)^2 + (2-1)^2})^2 = 1.25$.
4. Given that the value of the local error J_Z is not minimized, OKM returns the old assignment which is $C2$.

Once assignments are performed for all observations, OKM evaluates the optimized objective function known the obtained partitioning of data as described in Figure 1.8(a). The evaluation of the objective function gives $J_{OKM}(iteration1) = 5$.

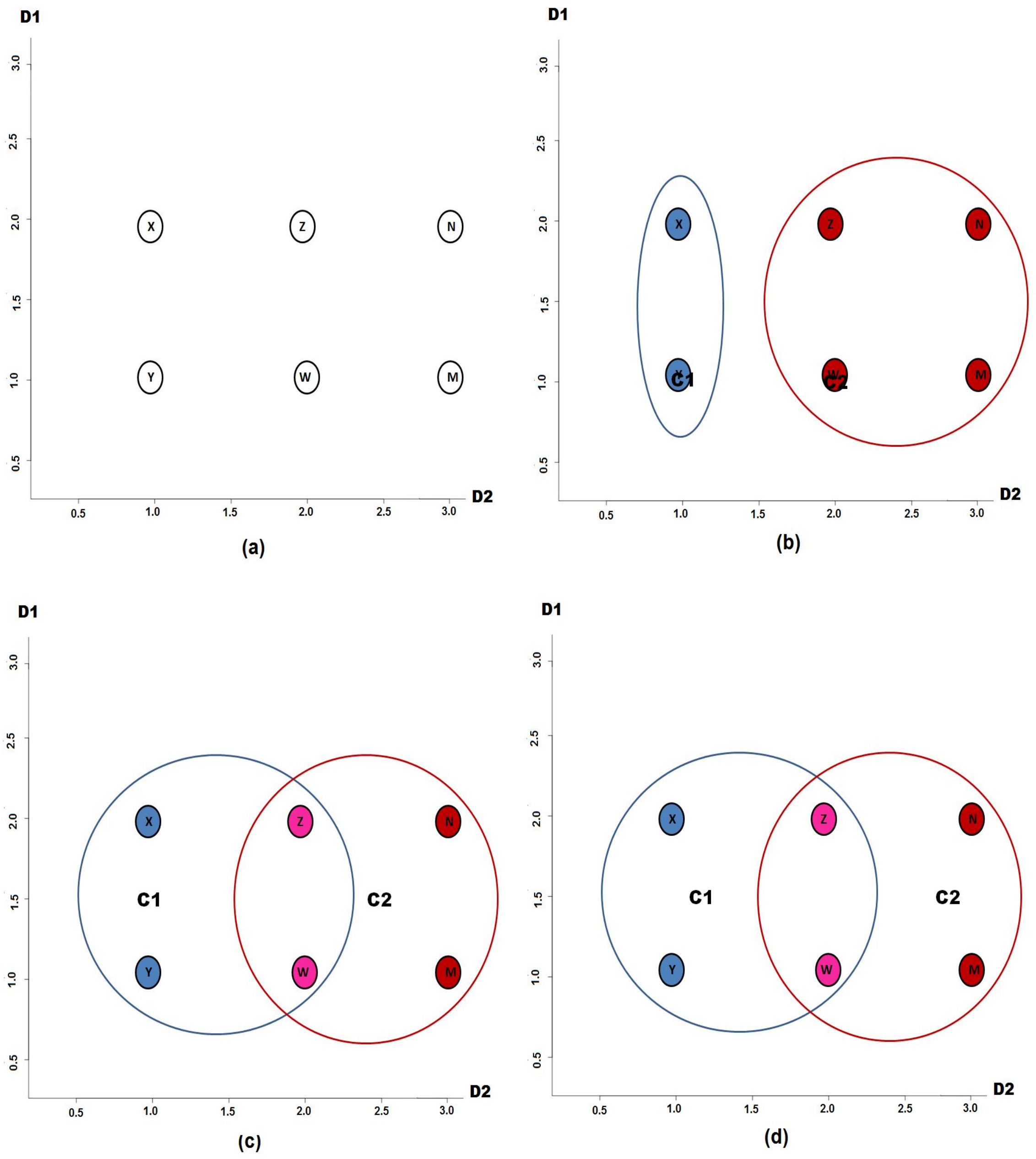


Figure 1.8: Non-disjoint clusters obtained using OKM: (a) input data (b) obtained clusters at iteration 1 (C) obtained clusters at iteration 2 (d) obtained clusters at iteration 3 and 4

Iteration 2 of OKM

At the second iteration, OKM begins by updating the representative of each cluster separably using Equation 1.35. The update of representatives gives $C1(1, 1.5)$ and $C2(2.5, 1.5)$.

We note here that the obtained representative for each cluster is the barycenter of the respective observations in the cluster, similarly to k-means, given that obtained clusters in the last iteration are disjoint. After that, the step of update of assignments gives the partitioning described in Figure 1.8(b). We show in this figure that observations Z and W belong to both clusters $C1$ and $C2$. The new evaluation of the objective function gives $J_{OKM}(iteration2) = 2.125$ which is improved compared to the objective criterion in iteration 1. Therefore, OKM continues the iteration of these different steps.

Iteration 3 of OKM

At the third iteration, OKM updates the representative of the different clusters which resulted in the new representatives $C1(1.1, 1.5)$ and $C2(2.9, 1.5)$. However, the update of assignments gives the same non-disjoint partitioning obtained in iteration 2 as described in Figure 1.8(c). The evaluation of the objective function gives $J_{OKM}(iteration3) = 1.52$ which is improved compared to the objective criterion in iteration 2. The OKM method repeats the same steps in iteration 4.

Iteration 4 of OKM

At the fourth iteration, the update of representatives of clusters and the update of assignments give same values obtained in iteration 3. The new evaluation of the objective function gives $J_{OKM}(iteration4) = 1.52$ which does not show improvement compared to that of iteration 3. Therefore, the OKM method converges and returns the obtained non-disjoint partitioning as described in Figure 1.8(c).

1.4.4 Synthesis of overlapping partitional methods

This section offers a synthesis of the main characteristics of overlapping partitional clustering methods presented in a comparative way. Table 1.2 summarizes these main characteristics. Our study is based on the following features of the methods: 1) model of overlaps in the objective criterion which could be *additive*, *geometrical* or *no overlap model*, 2) requirement of the method to use a post-assignment step to generate the final overlapping clusters which could be *threshold*, *max evidence* or *no post-assignment step* , 3) type of clusters' representatives which could be *centroid* or *medoid*, 4) type of data supported by each method which could be *numeric* or *any type*, 5) computational complexity, 6) ability to handle noise and outliers and 7) ability to regulate the sizes of overlaps. We also notice that all the studied methods performs *linear* separations between clusters.

We remark that some methods do not model overlaps in their optimized criteria like FCM and PCM. However, methods which supports an overlapping model lead to two main categories, additive and geometrical, which differer in the assumptions and in the context of use. The adoption of additive or geometrical methods is motivated by the requirement of the application. Additive methods have been well applied in grouping patients into diseases (Depril et al. 2008; Wilderjans et al. 2011). Each patient may suffer from more than one disease and therefore could be assigned to multiple syndrome clusters. Thus, the final symptom profile of a patient is the sum of the symptom profiles of all syndromes he is suffering from. However, this type of methods needs to prepare data to have zero mean to avoid false analysis. For example, if symptom variable represents the body temperature, then when a patient simultaneously suffers from two diseases, it is not realistic to assume that his body temperature equals to the sum of body temperatures as associated with two diseases.

Conversely, geometrical methods have been well applied to group music signals into different emotions and films into several genres. These methods consider that overlapping observations must appear in the extremity surface between overlapping clusters. For example, if a film belongs to action and horror genres, it should have some shared properties with these categories of films but it can neither be a full action film neither a full horror one. So, overleaping films belonging to action and horror categories may appear in the limit surface between full horror and full action films.

Table 1.2: The main characteristics of the overlapping partitional clustering methods.

Method	Overlap model	Post-assignment step	Representatives	Type of data	Complexity	Outliers identification	overlap regulation
FCM	no overlap model	threshold	centroid	numeric	$O(N.k)$	no	yes
PCM	no overlap model	threshold	centroid	numeric	$O(N.k)$	yes	yes
ECM	geometrical	max evidence	centoid	numeric	$O(N.2^k)$	yes	yes
BCM	geometrical	max evidence	centoid	numeric	$O(N.k.2^k)$	yes	yes
OKM	geometrical	no post-assignment step	centroid	numeric	$O(N.k.\lg k)$	no	no
OKMED	geometrical	no post-assignment step	medoid	any type	$O(N^3.k)$	no	no
WOKM	geometrical	no post-assignment step	centroid	numeric	$O(N.k.\lg k)$	no	no
PCL	additive	no post-assignment step	centroid	numeric	$O(2^{N.k})$	no	no
ALS$_{lf1}$	additive	no post-assignment step	centroid	numeric	$O(N^3.2^k)$	yes	no
ALS$_{lf2}$	additive	no post-assignment step	centroid	numeric	$O(N^2.2^k)$	yes	no

While all described overlapping partitional clustering methods offer a richer model to fit the existing structures in data, some parameters need to be estimated before performing the learning. All the described methods require to configure the number of clusters in prior while this number is usually unknown. Different heuristics (Depril et al. 2012; Wilderjans et al. 2012) for determining the optimal number are used in the literature. For example, the user can test different clusterings with increasingly number of clusters and then, takes the clustering having the best balance between the minimization of the objective function and the number of clusters (Wilderjans et al. 2011).

Furthermore, all the described overlapping partitional methods need to initialize the clusters' representatives or the primary clusters memberships. Clusters' representatives can be set randomly from data themselves or can be determined using another clustering method such as k-means. For initializing memberships, a Bernoulli distribution of parameter $\tau = 0.5$ can be used to generate random memberships. Using either a representative initialization or memberships initialization, the result of the presented methods may be a local optimum of the objective criterion, rather than the global optimum. To deal with this problem, the user should adopt a multi-start procedure by testing different initializations and keeping only the clustering which has the lowest value of the objective criterion.

All the presented overlapping methods are essentially based on the Euclidean distance between the sets of observations. Since they use numerical data, these methods are able to perform only linear and spherical separations between overlapping clusters. In fact, in real life applications data may have complex organization with non-spherical shapes such as ellipsoid and concentric shapes. In such cases, the performance of existing overlapping methods is considerably reduced while nonlinear separations with complex shapes are expected.

The study of partitional overlapping methods shows also that most of the described methods ignore the possibility to regulate the sizes of overlaps. This characteristic has an important effect in real applications because the expected size of overlaps differs from one application to another. For example, in social network analysis large overlaps are expected while an actor usually belongs to several active communities, however in video grouping small sizes of overlaps are expected since a movie or film is usually related to one genre. Although the importance of the regulation of overlaps, almost described methods, except uncertain memberships based methods like FCM, ECM and BCM, build a fixe size of overlaps between clusters. However, the regulation of overlaps in uncertain methods is much sensitive as the number of clusters increases.

Finally, we notice that all the presented partitional methods have the assumption that data support a vectorial description with numerical variables. This fact could limit the use of overlapping methods in many applications where data can not be described by numerical vectors but by other data formats like graphs and sequences. An example of these applications is textual document clustering where using vectorial description of text usually leads to ignore relations between words and consequently leads to reduce the performance of the clustering.

In this thesis, we are concerned with three limits of the existing methods which are the inability to look for nonlinear separations, the inability to regulate the sizes of overlaps and the inability to cluster textual data having non vectorial description. The geometrical model is adopted in all the presented solutions to introduce overlaps without loss of generality.

1.5 Evaluation of overlapping clustering

The evaluation of clustering, also referred to as cluster validity, is a crucial process to assess the performance of the learning method in identifying relevant groups. This process allows the comparison of several clustering methods and allows the analysis of whether one method is superior to another one. Usually, the evaluation of clustering can be categorized into *internal* and *external*. The first type, internal evaluation, is based only on the output of clustering by measuring the closeness of observations from one cluster and the distinctions of observations from different clusters. External evaluation, on the other hand, is based on comparisons between the output of the clustering and a dataset with known labels, also referred to as gold standard, usually built using human assessors.

For overlapping clustering, most of the validity measures traditionally used for clustering assessment, including both internal and external evaluations, become obsolete because of the multiple assignments of each observation. Despite this, some works (Banerjee et al. 2005; Amigo et al. 2009; Tsoumakas et al. 2010) propose an extension of well known validation measures to validate overlapping partitioning. In particular, internal evaluation measures, such as purity and entropy-based measures, cannot capture this aspect of the quality of a given clustering solution because they focus on the internal quality of the clusters. However, external validation measures, essentially Precision-Recall measures, were designed for overlapping partitioning. We give in the following three evaluation methods used for computing precision-recall measures for overlapping clustering which are respectively Label based, Pair based and BCubed evaluations.

1.5.1 Label based evaluation

Label based evaluation (Tsoumakas et al. 2010) is usually used in the field of Information Retrieval (IR) where each document can discuss several topics. This evaluation method is based on the evaluation of each class separately. Given a set of observations $X = \{x_1, ..., x_N\}$ and two partitions over X to compare, $C = \{c_1, ..., c_k\}$ a non-exclusive partitioning of X into k classes representing true labels, and $\Pi = \{\pi_1, ..., \pi_k\}$ a non-exclusive partitioning of X into k clusters where Π is defined by the clustering algorithm. Known the following:

- True positive TP_{ij}: the number of observations in π_j that exist in c_i,
- False negative FN_{ij}: the number of observations in c_i that not exist in π_j
- False positive FP_{ij}: the number of observations in π_j that not exist in c_i,

the Precision-Recall validation measures are computed for each class i and cluster j as follows:

$$Precision_{ij} = \frac{TP_{ij}}{TP_{ij}+FP_{ij}}$$
$$Recall_{ij} = \frac{TP_{ij}}{TP_{ij}+FN_{ij}}$$
$$F-measure_{ij} = \frac{(2*Recall_{ij}*Precision_{ij})}{(Recall_{ij}+Precision_{ij})}.$$

The computation of Precision-Recall measures for all labels is archived using macro-averaging technique which is usually used in Information Retrieval tasks to evaluate clustering results when the number of classes is not large (Yang 1999) as follows:

$$\begin{aligned} Recall &= \frac{\sum_{i=1}^{k} \max_j Recall_{ij}}{k} \\ Precision &= \frac{\sum_{i=1}^{k} \max_j Precision_{ij}}{k} \\ Fmeasure &= \frac{\sum_{i=1}^{k} \max_j Fmeasure_{ij}}{k}. \end{aligned} \tag{1.44}$$

1.5.2 Pair based evaluation

The pair based Precision-Recall measures are calculated over pairs of observations (Banerjee et al. 2005). For each pair of observations that share at least one cluster in the overlapping clustering results, Precision-Recall measures evaluate whether the prediction of this pair as being in the same cluster is correct with respect to the underlying true class in the data.

Given a set of observation $X = \{x_1, ..., x_N\}$ and two non-exclusive partitionings over X to compare, $C = \{c_1, ..., c_k\}$ a partition of X into k classes, and $\Pi = \{\pi_1, ..., \pi_{k1}\}$ a partition of X into $k1$ clusters and by Considering the following:

- TP : the number of pairs of observations in X that share at least one class in C and share at least one cluster in Π,
- FN : the number of pairs of observations in X that share at least one class in C and do not share any cluster in Π and

- FP : the number of pairs of observations in X that do not share any class in C and share at least one cluster in Π,

the Precision-Recall measures are computed as follows:

$$\text{Precision} = (TP)/(TP+FP)$$
$$\text{Recall} = (TP)/(TP+FN)$$
$$\text{F-measure} = (2{*}\text{Recall}{*}\text{Precision})/(\text{Recall}{+}\text{Precision}).$$

1.5.3 BCubed evaluation

Given the importance of observation occurrences in clusters and classes in overlapping partitioning, the BCubed evaluation (Amigo et al. 2009) takes into account the multiplicity of classes and clusters which considers the fact that two observations sharing n classes should share n clusters.

BCubed Precision-Recall measures are computed independently for each observation in the partitioning. Let the following:

$$Multiplicity \quad precision(x_i, x_j) = \frac{Min(|\Im(x_i) \cap \Im(x_j)|, |L(x_i) \cap L(x_j)|)}{|\Im(x_i) \cap \Im(x_j)|}$$

$$Multiplicity \quad recall(x_i, x_j) = \frac{Min(|\Im(x_i) \cap \Im(x_j)|, |L(x_i) \cap L(x_j)|)}{|L(x_i) \cap L(x_j)|} \tag{1.45}$$

where x_i and x_j are two observations, $L(x_i)$ the set of classes and $\Im(x_i)$ the set of clusters associated to observation x_i. In fact, Multiplicity Precision is defined only when the pair of observations (x_i, x_j) share at least one cluster, and Multiplicity Recall is defined only when (x_i, x_j) share at least one class. Multiplicity Precision is maximal, equal to 1, when the number of shared clusters is lower or equal than the number of shared classes and it is minimal, equal to 0, when the two observations do not share any class. Reversely, Multiplicity Recall is maximal when the number of shared classes is lower or equal than the number of shared clusters, and it is minimal when the two observations do not share any cluster.

The BCubed precision associated to one observation will be its averaged multiplicity precision over other observations sharing some of its classes; and the overall BCubed precision will be the averaged precision of all observations. The overall BCubed recall is obtained using the same procedure. The overall BCubed Precision-Recall measures can be formally described by:

$$Precision = AVG_i[AVG_{j.C(x_i)\cap C(x_j)\neq\varnothing} Multiplicity \quad precision(x_i, x_j)]\forall i, j \in \{1, .., N\}$$

$$Recall = AVG_i[AVG_{j.L(x_i)\cap L(x_j)\neq\varnothing} Multiplicity \quad recall(x_i, x_j)]\forall i, j \in \{1, .., N\}$$

$$F - measure = (2 * Precision * Recall)/(Precision + Recall)$$

1.5.4 Synthesis of evaluation methods for overlapping clustering

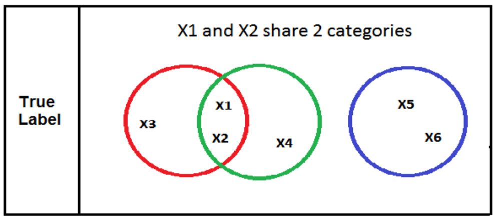
Three evaluation methods for assessing the quality of overlapping clustering were presented. The first evaluation method, label based evaluation, is based on matching between labels while the two others are based on pairs of observations. In fact, the label based evaluation requires to label the obtained clusters by matching between classes and clusters which is not a trivial task for unsupervised learning, especially for datasets with large overlaps. The labeling of clusters could lead to biased matching, and consequently lead to biased validation measures. Moreover, the matching between classes and clusters requires to configure a number of clusters equal to that of classes which limits the process of evaluation. Although these limitations, label based evaluation is usually used in Information Retrieval tasks to evaluate clustering results when the number of classes and the sizes of overlaps are not large (Yang 1999). Consequently, label based evaluation will be used to evaluate the quality of partitionings of documents obtained with the proposed method in Chapter 4.

To address the issue of labeling the obtained clusters and to make possible the comparison of partitionings with different number of clusters, the pair based Precision-Recall measures are calculated over pairs of observations. This evaluation method offers a flexible evaluation of obtained clusters independently from the real number of classes in the labeled dataset. However, the pair based evaluation has the issue that obtained Recall could be biased as the built overlap in the partitioning decreases and the actual overlap in the dataset increases. The biased Recall is induced by considering only a *binary* function for assessing the relation between a pair of observations and ignoring the *multiplicity* of shared clusters between pairs of observations. For instance, if two observations share three classes in the actual dataset and just share two clusters in the partitioning, the obtained Recall is 1 which is not correct. This problem also occurs for the Precision when actual overlap in the dataset is large.

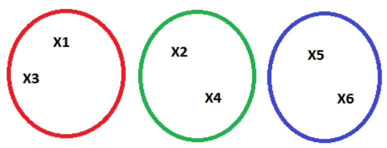
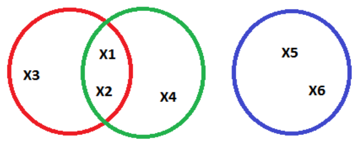
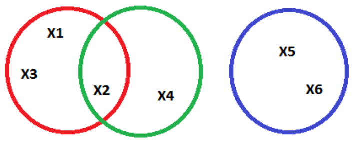
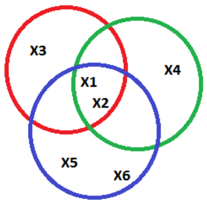
Given the importance of observation occurrences in clusters and classes in overlapping partitioning, the relation between two observations can not be represented as a binary function. If two observations share two classes and share just one cluster, then the clustering is not capturing completely the relation between both observations as presented in case 3 in Figure 1.9. On the other hand, if two observations share three clusters but just two classes, then the clustering is introducing more information than necessary as shown in case 3 in Figure 1.9. This relation is considered for BCubed validation measures by extending the pair based evaluation to take into account the multiplicity of clusters and classes.

Figure 1.9 shows an illustrative example comparing Precision and Recall computed for a pair of observations (x_1, x_2) using BCubed and pair based evaluations by considering different case-studies. We notice that Precision and Recall using the label based evaluation are not reported because they can not be computed for a pair of observations. This comparison shows that multiplicity Recall is reduced compared to Recall computed with the pair based evaluation if the partitioning gives less shared clusters than needed as described in case 3. In contrast, if the partitioning gives more shared clusters than actual labels as described in case 4, the multiplicity Precision is reduced compared to the one obtained with pair based evaluation.

As a summary, we can conclude that assessing the quality of overlapping clustering using the BCubed evaluation is more suitable than the pair based evaluation while it



(a)

Different cases		Bcubed Evaluation	Pair Evaluation
Case 1	X1 and X2 don't share any cluster 	Multiplicity Recall(X1,X2)=min(0,2)/2=0 Multiplicity Precision(X1,X2)=Undefined	Recall(X1,X2)= 0 Precision(X1,X2)= Undefined
Case 2	X1 and X2 share two clusters 	MultiplicityRecall(X1,X2)=min(2,2)/2=1 Multiplicity Precision(X1,X2)=min(2,2)/2=1	Recall(X1,X2)= 1 Precision(X1,X2)= 1
Case 3	X1 and X2 share one cluster 	MultiplicityRecall(X1,X2)=min(1,2)/2=0.5 MultiplicityPrecision(X1,X2)= min(1,2)/1=1	Recall(X1,X2)= 1 Precision(X1,X2)= 1
Case4	X1 and X2 share three clusters 	MultiplicityRecall(X1,X2) =min(3,2)/2=1 MultiplicityPrecision(X1,X2) =min(3,2)/3=0.66	Recall(X1,X2) = 1 Precision(X1,X2) = 1

(b)

Figure 1.9: Comparison of Recall and Precision measures computed using the BCubed and the Pair based evaluations: (a) true labels and (b) different cases of the clustering.

takes into account the multiplicity of shared clusters and classes between the pair of observations. The BCubed method will be used to evaluate the performance of the proposed methods, in Chapter 2 and Chapter 3, against the existing ones.

1.6 Conclusion

We introduced in this chapter clustering as an unsupervised learning method requiring data format and similarity or dissimilarity measure definition. Then, we introduced the domain of overlapping clustering which aims to build non-disjoint groups from unlabeled data. We have shown that enabling observations to belong to more than one cluster would be recommended for several applications to better fit the existing structures in data. We reviewed existing overlapping clustering methods, especially those based on the partitional approach. Furthermore, we detailed different evaluation methods designed to assess the quality of overlapping clusters and we have showed that BCubed evaluation offers the more suitable assessment.

In the next chapter, we will focus on the issue of identifying overlapping clusters with nonlinear separations. We will show how can we make non-disjoint partitioning with linear and nonlinear separations.

Chapter 2

Identification of overlapping clusters with nonlinear boundaries

Contents

2.1 Introduction

This chapter deals with the issue of identifying overlapping clusters with nonlinear and non spherical separations. We focus on kernel k-means, a nonlinear variant of k-means, which is generalized to look for overlapping clusters. Two variants are proposed (BenN'Cir and Essoussi 2011; BenN'Cir and Essoussi 2012b) in this work, Kernel Overlapping K-Means I (KOKMI) and Kernel Overlapping K-Means II (KOKMII) to produce clusters in a high, possibly infinite, dimensional space based on Mercer Kernel technique. Data are implicitly mapped to a higher dimensional space where separability of input patterns is improved.

This chapter is organized as follows: Section 2.2 describes the motivation of identifying non-disjoint clusters with nonlinear boundaries. Then Section 2.3 describes the existing Kernel k-means algorithm used to perform nonlinear separations between clusters. After that Section 2.4 describes the two proposed solutions, KOKMI and KOKMII, which generalize kernel k-means for overlapping clustering while section 2.5 presents the experiments that we performed to check their effectiveness. Finally Section 2.6 presents the conclusion.

2.2 Motivation: overlapping clustering with nonlinear separations

In real life applications of clustering data usually have complex shapes. The identification of separations between groups is usually hard and require looking for non spherical and nonlinear separations. Therefore, overlapping clustering methods need to perform nonlinear and non spherical separations between clusters to fit the true structure of data.

In order to check this important characteristic, we study patterns produced by existing overlapping methods based on additive and geometrical models. We visualize partitioning of OKM, a *geometrical* based method, and ALS, an *additive* based method, through Voronoï cells obtained for three clusters over a two dimensional space as defined by the objective criterion optimized by these methods. Figure 2.1 shows an example of these Voronoï cells: the representation space is divided into several cells where each possible combination of clusters is associated to one cell. For OKM, Figure 2.1.(a) shows seven cells, all possible combinations of clusters except the empty set, where each cell is centered on a prototype or a combination of prototypes. For ALS, Figure 2.1.(b) shows that overlaps between clusters are not recovered, except $cluster1 \cap cluster2$ and $cluster2 \cap cluster3$. We notice that the Gray cell is the resulting combination (sum) of representatives of cluster 1 (Red cell) and cluster 2 (Green cell).

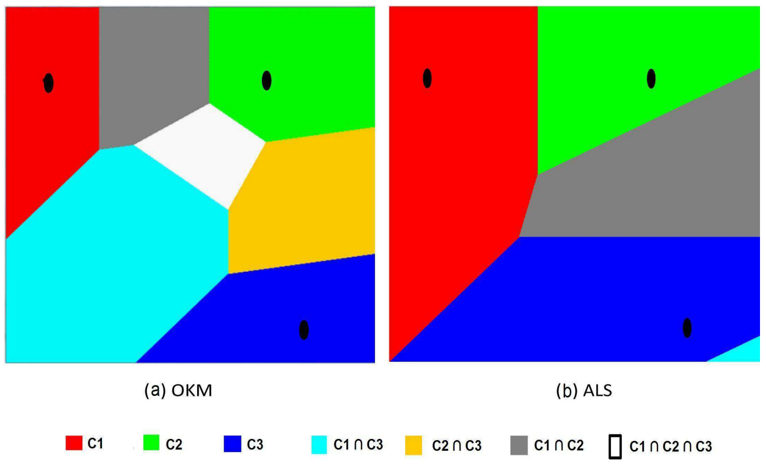


Figure 2.1: Voronoï cells obtained with OKM and ALS for three clusters.

Hence, OKM and ALS have the ability to detect overlapping clusters even though separations between resulting clusters are linear. This fact makes these methods not well adapted to real life applications where separations between clusters are usually nonlinear with complex shapes.

Since complex and nonlinear separations between clusters are expected depending on the target application, we focus our study on the nonlinear method kernel k-means (Girolami 2002). We show how it can be generalized to detect overlapping clusters with both linear and nonlinear boundaries.

2.3 Learning with nonlinear separations using Kernel Methods

2.3.1 Kernel Methods

To solve the problem of non-linearly-separable clusters, many methods have been modified incorporating kernels such as SVM (Cortes and Vapnik 1995) which performs better than other classification algorithms in many applications (Cristianini et al. 2002). The success of SVM has extended the use of kernels to other learning algorithms (e.g., Kernel PCA (Schölkopf et al. 1998), kernel k-means (Girolami 2002)). These methods use positive-definite kernel, also referred to as Mercer kernel, to implicitly map data from original space called input space into a high dimensional space called feature space. Computing a partition with linear boundaries in the feature space results in a partition with nonlinear boundaries 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 condition of positiveness holds:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} c_i c_j K_{ij} \geq 0 \quad \forall N \geq 2 \quad and \quad \forall c_i, c_j \in \mathbb{R}, \tag{2.1}$$

where K_{ij} is the dot product of mapped data in the feature space defined as follows:

$$K_{ij} = K(x_i, x_j) = \phi(x_i)\phi(x_j), \tag{2.2}$$

where $\phi : X \longrightarrow F$ is a mapping from the input space X to a high dimensional feature space F. Some widely used kernels given in Table 2.1 are the Linear, Polynomial, Gaussian, Exponential, Laplace, Quadratic, Inverse Multi Quadratic and Sigmoid kernels.

The advantage of kernel consists in the possibility of computing distance measure between observations in the feature space F without explicitly knowing ϕ. The kernel induced distance measure (Schölkopf et al. 1998), also referred to as Kernel Trick, can be

Table 2.1: 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	$K(x_i, x_j) = \frac{1}{\sqrt{\| x_i - x_j \|^2 + c^2}}$

computed as follows:

$$\begin{aligned} \| \phi(x_i) - \phi(x_j) \|^2 &= (\phi(x_i) - \phi(x_j))((\phi(x_i) - \phi(x_j)) \\ &= \phi(x_i)\phi(x_i) - 2\phi(x_i)\phi(x_j) + \phi(x_j)\phi(x_j) \\ &= K_{ii} - 2K_{ij} + K_{jj}. \end{aligned} \tag{2.3}$$

In fact, the use of Mercer kernel in clustering can be divided into three categories (Filippone et al. 2008). The first category includes methods based on kernelization of the metric (Zhang and Chen 2002; Zhang and Chen 2003; Zhang and Chen 2004; Wu et al. 2003) which look for centroids in input space and distances between patterns and centroids are computed by means of kernels. Second, methods based on clustering in the feature space (Graepel and Obermayer 1998; Girolami 2002; Inokuchi and Miyamoto 2004; Qinand and Suganthan 2004) which map data into a higher feature space and then compute centroids using the Kernel Trick. Third, methods based on support vectors (Ben-Hur et al. 2001; Camastra and Verri 2005) which use One Class SVM to find a minimum enclosing sphere in the feature space able to include almost all data excluding outliers.

In this work, Mercer kernel is used for clustering in the feature space (the second category of the use of Mercer kernel in clustering) for KOKMI and KOKMII where the whole learning process is performed in a high dimensional space.

2.3.2 Kernel k-means method

Kernel k-means (Girolami 2002) is an extension of k-means to solve the problem of non-linearly separable clusters. By an implicit mapping of the data from an input space to a higher feature space, kernel k-means looks for separations in feature space and solves the problem of clustering non-linearly-separable data. For a finite data sample X, the kernel function yields a symmetric $N \times N$ positive definite matrix K, where each K_{ij} entry is the dot product between the representations in feature space, $\phi(x_i)$ and $\phi(x_j)$, of observations x_i and x_j as measured by the kernel function.

Kernel k-means aims to minimize the sum of squared Euclidean errors in feature space given by:

$$J(\Pi) = \sum_{i=1}^{N} \sum_{c=1}^{k} P_{ic} \|\phi(x_i) - m_c^{\phi}\|^2, \tag{2.4}$$

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 in the feature space as the gravity center 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:

$$\begin{aligned} \|\phi(x_i) - m_c^{\phi}\|^2 &= \|\phi(x_i) - \frac{1}{W_c} \sum_{j=1}^{N} P_{jc}\phi(x_j)\|^2 \\ &= K_{ii} - \frac{2}{W_c} \sum_{j=1}^{N} P_{jc} K_{ij} + \frac{1}{(W_c)^2} \sum_{j=1}^{N} \sum_{g=1}^{N} P_{jc} P_{gc} K_{jg}, \end{aligned} \tag{2.5}$$

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 $P_{gc} \in \{0,1\}$ denote the memberships of observations x_j and x_g to cluster c. Then, the clustering error function in kernel k-means can be written as follows:

$$J(\Pi) = \sum_{i=1}^{N}\sum_{c=1}^{k} P_{ic}[K_{ii} - \frac{2}{W_c}\sum_{j=1}^{N} P_{jc}K_{ij} + \frac{1}{(W_c)^2}\sum_{j=1}^{N}\sum_{g=1}^{N} P_{jc}P_{gc}K_{jg}]. \tag{2.6}$$

To minimize the clustering error (2.6), kernel k-means performs two principal steps: the determination of the nearest cluster from each observation in the feature space and the update of memberships matrix. The stopping rule is defined by the maximal number of iterations and the minimal improvement of the objective function between two iterations.

2.4 Proposed overlapping clustering with nonlinear boundaries

In order to look for overlapping clusters with linear and nonlinear boundaries, we propose two methods, KOKMI and KOKMII, where all steps of the clustering algorithm are performed in a high dimensional feature space implicitly computed based on kernels. For both methods the *geometrical* model is adopted to introduce overlaps in the objective criterion.

2.4.1 KOKMI: Kernel Overlapping K-Means I

The first method generalizes kernel k-means to produce overlapping clusters by introducing overlaps between clusters in the objective function. Given a set of observations $X = \{x_i\}_{i=1}^{N}$ with $x_i \in \mathbb{R}^d$ and N the number of observations and given an implicit nonlinear mapping function ϕ, the aim of KOKMI is to find a set $\Pi = \{\pi_1, ..., \pi_k\}$ of k overlapping clusters such that the following objective function is minimized:

$$J(\Pi) \quad = \quad \sum_{i=1}^{N} \|\phi(x_i) - \overline{\phi(x_i)}\|^2. \tag{2.7}$$

The objective function iteratively minimizes the distance between each observation and its representative in a new feature space F obtained by the nonlinear mapping function ϕ. The representative $\overline{\phi(x_i)}$ of observation x_i is defined by the *average* of clusters prototypes which x_i belongs to. The representative is performed in feature space and is described by:

$$\overline{\phi(x_i)} = \frac{\sum_{c=1}^{k} P_{ic} m_c^{\phi}}{\sum_{c=1}^{k} P_{ic}}, \tag{2.8}$$

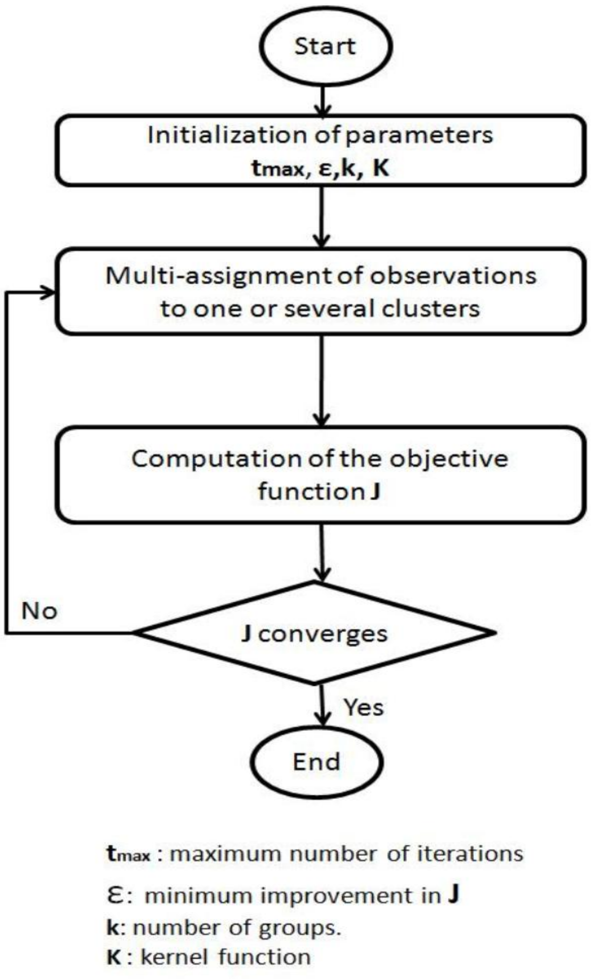

Start
Initialization of parameters
tmax, ε,k, K
Multi-assignment of observations to one or several clusters
Computation of the objective function J
J converges
No
Yes
End
tmax : maximum number of iterations
ε: minimum improvement in J
k: number of groups.
K : kernel function

Figure 2.2: Main steps of KOKMI

where $P_{ic} \in \{0, 1\}$ is a binary variable that indicates membership of x_i to cluster c and m_c^{ϕ} is the representative (prototype) of cluster c in the feature space. To minimize the objective function J, KOKMI assigns, at each iteration, observations to one or several clusters and then computes the new value of J. If J shows improvement, the same steps are repeated until optimization of this function. The function J is considered optimized if the maximum number of iterations is reached or the minimum improvement of this function is no longer significant. The main steps of KOKMI are described in Figure 2.2. A pseudo code of KOKMI is presented in Algorithm 2.

Algorithm 2 $KOKMI(X, t_{max}, \varepsilon, k) \rightarrow \Pi = \{\pi_c\}_{c=1}^{k}$

```
INPUT $X$: set of observations in $\mathbb{R}^d$.
OUTPUT $\Pi$: memberships over $k$ clusters.
1: Choose the kernel function and its corresponding parameters.
2: Initialize representatives of clusters with random clusters prototypes, derive clusters
   memberships using "$ASSIGN$" and compute value of the objective function $J_{t=0}(\Pi)$
   in iteration 0 using Equation 2.11.
3: Assign each observation $x_i$ to one or several clusters using function"$ASSIGN$".
4: Compute objective function $J_t(\Pi)$ using Equation 2.11.
5: if ($t < t_{max}$ and $J_{t-1}(\Pi) - J_t(\Pi) > \varepsilon$) then
6:    go to step 3.
7: else
8:    return the distribution of clusters' memberships.
9: end if
```

Cluster prototype computation in feature space

To evaluate the objective function J we need to compute at each iteration clusters prototypes in the feature space. Each prototype is defined by the average of observations that belong to the corresponding cluster weighted according to the number of assignments of each observation as follows:

$$m_c^{\phi} = \frac{\sum_{j=1}^{N} P_{jc} w_j \phi(x_j)}{W_c}, \tag{2.9}$$

where w_j is the single weight assigned to observation x_j defined by $w_j = 1/(\sum_{c=1}^{k} P_{jc})^2$ and $W_c = \sum_{j=1}^{N} P_{jc} w_j$ is the sum of single weights of the observations which belong to cluster

c. The single weight w_j decreases as well as assignments of x_j increase in order to reduce the impact of overlapping observations in determining the representative of each cluster.

Since the mapping function ϕ is not explicitly known, it is impossible to compute clusters prototypes in feature space using Equation 2.9. Nevertheless, it is always possible to compute distances between observations and prototypes in feature space using the Kernel Trick. Therefore, using this technique, the objective function of KOKMI becomes:

$$\begin{aligned} J(\Pi) &= \sum_{i=1}^{N} \|\phi(x_i) - \frac{1}{L_i}\sum_{c=1}^{k} P_{ic}\frac{1}{W_c}\sum_{j=1}^{N} P_{jc}w_j\phi(x_j)\|^2 \\ &= \sum_{i=1}^{N}\{\phi(x_i)\phi(x_i) - \frac{2}{L_i}\sum_{c=1}^{k}\sum_{j=1}^{N} P_{ic}\frac{1}{W_c}P_{jc}w_j\phi(x_i)\phi(x_j) + \\ &\quad \frac{1}{L_i^2}\sum_{c=1}^{k}\sum_{j=1}^{N}\sum_{t=1}^{k}\sum_{g=1}^{N} P_{ic}\frac{1}{W_c}P_{jc}P_{it}\frac{1}{W_t}P_{gt}w_jw_g\phi(x_j)\phi(x_g)\}, \end{aligned} \tag{2.10}$$

where $L_i = \sum_{c=1}^{k} P_{ic}$. If each dot product in feature space is replaced by the Mercer kernel, the objective function J can be performed in feature space as follows:

$$J(\Pi) = \sum_{i=1}^{N} d[\phi(x_i), \overline{\phi(x_i)}], \tag{2.11}$$

where:

$$\begin{aligned} d[\phi(x_i), \overline{\phi(x_i)}] &= K_{ii} - \frac{2}{L_i}\sum_{c=1}^{k}\sum_{j=1}^{N} P_{ic}\frac{1}{W_c}P_{jc}w_jK_{ij} + \\ &\quad \frac{1}{L_i^2}\sum_{c=1}^{k}\sum_{j=1}^{N}\sum_{t=1}^{k}\sum_{g=1}^{N} P_{ic}\frac{1}{W_c}P_{jc}P_{it}\frac{1}{W_t}P_{gt}w_jw_gK_{jg}. \end{aligned} \tag{2.12}$$

If observations are assigned to only one cluster as in hard clustering methods, the objective function in KOKMI is reduced to:

$$\begin{aligned}
J(\Pi) &= \sum_{i=1}^{N}[K_{ii} - \frac{2}{1}\sum_{c=1}^{k}\sum_{j=1}^{N} P_{ic}\frac{1}{W_c}P_{jc}K_{ij} + \\
&\qquad \frac{1}{1^2}\sum_{c=1}^{k}\sum_{j=1}^{N}\sum_{t=1}^{k}\sum_{g=1}^{N} P_{ic}\frac{1}{W_c}P_{jc}P_{it}\frac{1}{W_t}P_{gt}K_{jg}] \\
J(\Pi) &= \sum_{i=1}^{N}\sum_{c=t=1}^{k} P_{ic}[K_{ii} - \frac{2}{W_c}\sum_{j=1}^{N}P_{jc}K_{ij} + \frac{1}{(W_c)^2}\sum_{j=1}^{N}\sum_{g=1}^{N}P_{jc}P_{gc}K_{jg}].
\end{aligned} \tag{2.13}$$

This reduced objective function exactly matches with the objective function of kernel k-means. This fact shows that KOKMI is a *generalization* of kernel k-means for overlapping clustering.

Multi-assignments of observations in feature space

The assignment step is a combinatorial discrete optimization problem. The latter is solved through searching for optimal cluster memberships by evaluating all possible 2^k combinations of clusters for each observation. However, it becomes computationally infeasible in real life applications. In the following, a heuristic solution is introduced to minimize the objective function and to explore a sub space of possible assignments. Given an observation x_i, given a set of k clusters and possibly an old assignments of x_i, new optimal assignments of x_i are determined using the function ASSIGN. Firstly, this function assigns x_i to the closest cluster, updates representative $\overline{\phi(x_i)}$ and then evaluates the distance between the observation and its representative using Equation 2.12. Next, this function looks for the next nearest cluster which is not included in the set of assignments. This cluster is added to the assignments of observation x_i, then the representative $\overline{\phi(x_i)}$ is updated and the distance between the observation and its new representative is reevaluated. If the distance is minimized, the function continues the assignment of the observation to the next nearest cluster. Otherwise, the function compares obtained assignments in the last step with those given before performing the function ASSIGN and returns the optimal assignments which minimize the distance between the observation and its representative.

The old assignments are evaluated to ensure the minimization of the objective function after each assignment step. The pseudo code of the function ASSIGN is described by Algorithm 3.

Algorithm 3 $ASSIGN(x_i, \{c = 1, ..., c = k\}, \Pi_i^{old}) \rightarrow \Pi_i$

```
INPUT $x_i$: Observation in $\mathbb{R}^d$.
      $\{c = 1, ..., c = k\}$: Set of $k$ clusters.
      $\Pi_i^{old}$: Old assignment of observation $x_i$.
OUTPUT $\Pi_i$: New assignment of $x_i$.
 1: set $\Pi_i = \{c^\star\}$ using Equation 2.14 , evaluate the distance $d[\phi(x_i), \overline{\phi(x_i)}]$ with assignments $\Pi_i$ using Equation 2.12.
 2: Look for the next nearest cluster $c^\star$ which is not included in $\Pi_i$ such that $c^\star = \min_{\{c=1,...,c=k\}/\Pi_i} \|\phi(x_i) - m_c^\phi\|^2$ using Equation 2.14
 3: Evaluate the distance $d[\phi(x_i), \overline{\phi(x_i)'}]$ with assignments $\Pi_i' = \Pi_i \cup c^\star$
 4: if $d[\phi(x_i), \overline{\phi(x_i)'}] \leq d[\phi(x_i), \overline{\phi(x_i)}]$ then
 5:   $\Pi_i \leftarrow \Pi_i \cup \{c^\star\}$, and go to step 2.
 6: else
 7:   if $d[\phi(x_i), \overline{\phi(x_i)}] \leq d[\phi(x_i), \overline{\phi(x_i)^{old}}]$ then
 8:     return $\Pi_i$.
 9:   else
10:     return $\Pi_i^{old}$.
11:   end if
12: end if
```

For the determination of the closest cluster from an observation, it could be performed in feature space using the Kernel Trick as follows:

$$\begin{aligned} c^\star &= \min_{\{c=1,...,c=k\}} \|\phi(x_i) - m_c^\phi\|^2 \qquad (2.14) \\ &= \min_{\{c=1,...,c=k\}} \{K_{ii} - \frac{2}{W_c} \sum_{j=1}^{N} P_{jc} w_j K_{ij} + \frac{1}{(W_c)^2} \sum_{j=1}^{N} \sum_{g=1}^{N} P_{jc} P_{gc} w_j w_g K_{jg}\}. \end{aligned}$$

Evaluation of the computational complexity of KOKMI

We describe in the following the evaluation of the computational complexity of KOKMI method. We considered that the computational complexity of Mercer Kernel $K(x_i, x_j) = K_{ij}$ between each pair of observations is equal to $O(1)$ because we use an online implementation of the Kernel function. Given N the number of observations and k the number of expected clusters, the main algorithm of KOKMI iterates two independent steps which are respectively: 1) update of assignments for each observation and 2) evaluation of the objective function with new assignments. Therefore, the hole computational complexity of KOKMI can be determined by the sum of the computational complexity of this two steps:

1. **Computational complexity of the step of update of assignments:** the assignment of clusters is determined for each observation in the dataset by iterating two independent sub-steps: 1) looking for the nearest cluster and 2) evaluation of the local error. The first sub-step is determined by Equation 2.14 which is evaluated by the sum of the computational complexity of its three terms $O(1 + N + N^2) \simeq O(N^2)$. The second sub-step is determined by Equation 2.12 which is evaluated by $O(1 + N.k + N^2.k^2) \simeq O(N^2.k^2)$. As a result, the computational complexity of the assignment step, for each observation, is evaluated by $O(N^2.k^2 + N^2) \simeq O(N^2.k^2)$. Given that this step is repeated for each observation, the evaluation of the computational complexity of the assignment step for all observations is evaluated by $O(N.[N^2.k^2]) \simeq O(N^3k^2)$.

2. **Computational complexity of the step of the evaluation of the objective function:** the step of the evaluation of the objective function is determined by Equation 2.11 which is evaluated by $O(N.[1 + N.k + N^2.k^2]) \simeq O(N^3.k^2)$.

Given the computational complexity of each step of the main algorithm of KOKMI, the computational complexity of this method is evaluated by $O(N^3.k^2 + N^3.k^2) \simeq O(N^3.k^2)$.

2.4.2 KOKMII: Kernel Overlapping K-means II

In order to detect overlapping clusters with linear and nonlinear separations between clusters we propose a second alternative, referred to as KOKMII, where cluster centroids are replaced by clusters medoids in order to improve the efficiency of clustering and the computational complexity of KOKMI.

Computation of cluster's representative in KOKMII

The new definition of cluster's representative of KOKMII in the feature space is based on cluster medoid. Each cluster's representative is defined as the observation that minimizes all distances over all other observations of the cluster. Using Mercer kernel, the prototype is performed as follows:

$$\begin{aligned} m_c = & \min_{i \in N_c}(x_i) \frac{\sum_{j=1, j \neq i}^{N_c} \frac{1}{w_j} \|\phi(x_i) - \phi(x_j)\|^2}{N_c \sum_{j=1, j \neq i}^{N_c} \frac{1}{w_j}} \\ = & \min_{i \in N_c}(x_i) \frac{\sum_{j=1, j \neq i}^{N_c} \frac{1}{w_j} [K_{ii} - 2K_{ij} + K_{jj}]}{N_c \sum_{j=1, j \neq i}^{N_c} \frac{1}{w_j}}, \end{aligned} \quad (2.15)$$

where N_c is the number of observations in cluster c. In this way, the prototype is determined in the feature space F and is a member of the initial set of observations.

Clustering algorithm of KOKMII

Given the new way of determining prototypes in the feature space, the computational complexity of the objective function J is reduced to $O(N.k^2)$ compared to KOKMI. The objective function is performed as shown in Equation 2.16.

$$\begin{aligned} J(\Pi) &= \sum_{i=1}^{N} \|\phi(x_i) - \overline{\phi(x_i)}\|^2 \\ &= \sum_{i=1}^{N} \|\phi(x_i) - \frac{\sum_{c=1}^{k} P_{ic}\phi(m_c)}{L_i}\|^2 \\ &= \sum_{i=1}^{N} d'[\phi(x_i), \overline{\phi(x_i)}], \end{aligned} \quad (2.16)$$

where $d'[\phi(x_i), \overline{\phi(x_i)}]$ is defined by:

$$\begin{aligned} &= \phi(x_i).\phi(x_i) - \frac{2}{L_i}\sum_{c=1}^{k} P_{ic}\phi(m_c)\phi(x_i) + \frac{1}{(L_i)^2}\sum_{c=1}^{k}\sum_{l=1}^{k} P_{ic}P_{il}\phi(m_c)\phi(m_l) \\ &= K_{ii} - \frac{2}{L_i}\sum_{c=1}^{k} P_{ic}K_{im_c} + \frac{1}{(L_i)^2}\sum_{c=1}^{k}\sum_{l=1}^{k} P_{ic}P_{i_l}K_{m_cm_l}. \end{aligned} \tag{2.17}$$

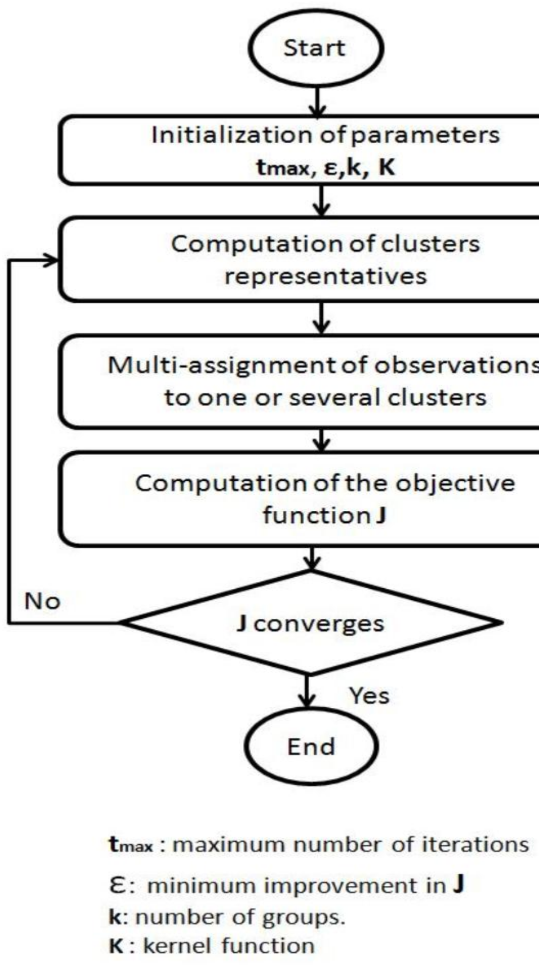

Start
Initialization of parameters
tmax, ε,k, K
Computation of clusters representatives
Multi-assignment of observations to one or several clusters
Computation of the objective function J
J converges
No
Yes
End
tmax : maximum number of iterations
ε: minimum improvement in J
k: number of groups.
K : kernel function

Figure 2.3: Main steps of KOKMII

However, KOKMII adds a new step for the determination of cluster prototypes independently from the evaluation of the objective function. At each iteration, cluster prototypes are computed, then observations are assigned to one or several clusters, and finally the objective function J is evaluated. These steps are repeated until improvement of J is no longer significant or the maximum number of iterations is reached. The main algorithm of KOKMII is given in Algorithm 4 and is schematized in Figure 2.3.

Algorithm 4 $KOKMII(X, t_{max}, \varepsilon, k) \rightarrow \Pi = \{\pi_c\}_{c=1}^{k}$

```
INPUT  $X$: set of observations in $\mathbb{R}^d$.
       $t_{max}$: maximum number of iterations.
       $\varepsilon$: minimal improvement in the objective function.
       $k$: number of clusters.
OUTPUT $\Pi$: memberships over $k$ clusters.
 1: Choose the kernel function and its corresponding parameters.
 2: Initialize representatives of clusters with random cluster prototypes, derive clusters
    memberships using "ASSIGN" and compute value of the objective function $J_{t=0}(\Pi)$
    in iteration 0 using Equation 2.16.
 3: Compute cluster prototypes using Equation 2.15.
 4: Assign each observation $x_i$ to one or several clusters using "ASSIGN".
 5: Compute objective function $J_t(\Pi)$ using Equation 2.16.
 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
```

Assignments of each observation to one or several clusters in KOKMII are performed using the same function ASSIGN. The new definition of clusters prototypes has made computation of the distance between each observation and its representative easier as described in Equation 2.17. Also, finding the closest cluster from an observation x_i in the feature space is computationally easier and is given by:

$$c^\star = \min_{\{m_c\}_{c=1}^{k}} \|\phi(x_i) - \phi(m_c)\|^2 = \min_{\{m_c\}_{c=1}^{k}} K_{ii} - 2K_{im_c} + K_{m_c m_c}. \tag{2.18}$$

Evaluation of the computational complexity of KOKMII

Let us recall that the main algorithm of KOKMII iterates three independent steps which are respectively: 1) update of clusters' representatives 2) update of assignments for each observation and 3) evaluation of the objective function. Therefore, the hole computational complexity of KOKMII can be determined by the sum of the computational complexity of this three steps:

1. **Computational complexity of the step of update of clusters' representatives:** the update of clusters' representatives is performed independently for each cluster using Equation 2.15 which is evaluated by $O(N_c^2)$ where N_c the maximum number of observations per cluster. Therefore, the computational complexity for updating clusters' representatives for k clusters is evaluated by $O(N_c^2.k)$.

2. **Computational complexity of the step of update of assignments:** the assignment step is determined for each observation in the dataset by iterating two independent sub-steps: 1) looking for the nearest cluster and 2) evaluation of the local error. The first sub-step is determined by Equation 2.18 which is evaluated by $O(k.[1+1+1]) \simeq O(k)$. The second sub-step is performed using Equation 2.17 which is evaluated by $O(1+k+k^2) \simeq O(k^2)$. As a result, the evaluation of the computational complexity of the assignment step, for each observation, is determined by $O(k+k^2) \simeq O(k^2)$. Given that this step is repeated for each observation, the evaluation of the computational complexity of the assignment step for all observations is evaluated by $O(N).O(k^2) \simeq O(N.k^2)$.

3. **Computational complexity of the step of the evaluation of the objective function:** the step of the evaluation of the objective function is determined by Equation 2.16 which is evaluated by $O(N.[1+k+k^2]) \simeq O(N.k^2)$.

As a summary, the computational complexity of KOKMII method is evaluated by: $O(N_c^2.k+N.k^2+N.k^2) \simeq O(N_c^2.k)$.

2.5 Experiments and discussions

In this section, through an experimental study we evaluate the performance of the two proposed methods in detecting overlapping groups with complex and nonlinear separations between clusters.

2.5.1 Evaluation methodology

We begin by studying patterns induced by the proposed objective criterion through the report of Voronoï cells for the example described in Figure 2.1. After that, we perform two experimental studies. First, experiments are conducted on datasets with complex, non spherical and nonlinear separations to check the effectiveness of proposed methods in detecting these clusters. Second, experiments are conducted on real overlapping datasets to check the effectiveness in identifying overlapping groups. For each dataset, the number of clusters is set by the number of underlying true categories. Although many methods and techniques were proposed to select the best parameter of the kernel function, it is still a challenging issue. Since designing the best parameter of a kernel function is not the objective of this work, it is determined empirically over different executions of proposed methods. However, we report results using different kernels with different initialization of parameters to show the sensitivity of proposed methods to the selection of a kernel.

Experiments are performed on a computer, with 4 GB RAM and 2.1 GHZ Intel Core 2 duo processor. Results are compared using four validation measures: Precision, Recall, F-measure and Overlap size. The first three validation measures are computed using the BCubed technique as described in Chapter 1. The fourth measure, Overlap size, evaluates the size of overlaps yielded by the learning method. This measure can be determined by the average number of clusters of each observation in the dataset as follows:

$$Overlap = \frac{\sum_{i=1}^{N} |\Pi_i|}{N}. \tag{2.19}$$

This important measure influences the performance of different overlapping clustering methods.

2.5.2 Evaluation of the nonlinearity of separations

To evaluate the performance of proposed methods in detecting complex and nonlinear separations between clusters, we visualize Voronoï cells (for 3 clusters) obtained with

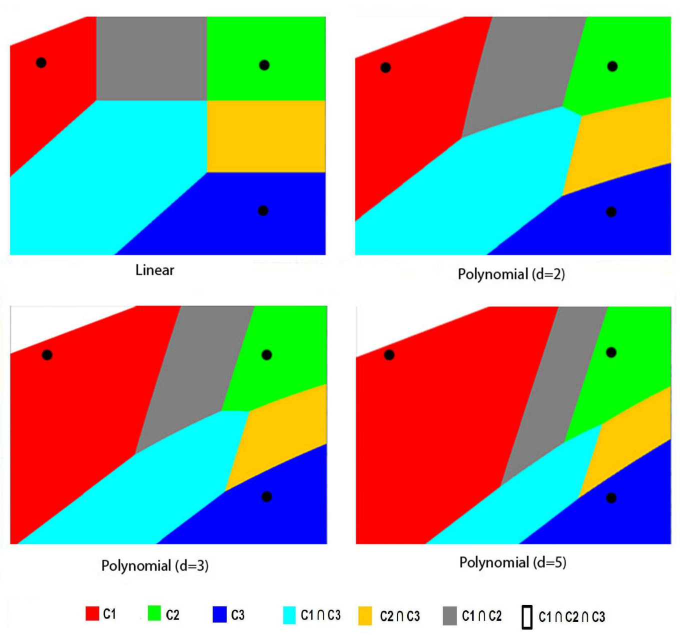


Figure 2.4: Voronoï cells obtained with KOKMII using Linear and Polynomial kernels

KOKMII using different types of kernels with different parameters. Figures 2.4 to 2.8 show the ability of KOKMII in detecting overlapping clusters with both linear and nonlinear boundaries depending of the type of the kernel function. For example, the Polynomial Kernel performs linear separations as showed in Figure 2.4, the Gaussian, Exponential and Laplace kernels perform nonlinear separations as showed in Figures 2.5, 2.6 and 2.7, while Sigmoid performs a more complex and nonlinear separations as showed in Figure 2.7. In fact, the choice of the kernel function and its parameters affects the structure of obtained patterns and influences the type of separations. For example, using Gaussian kernel, as the parameter σ increases overlapping regions between clusters are reduced. This result is also confirmed in real datasets where obtained overlap decreases as σ becomes larger.

Some kernels perform only linear separations between clusters, such as the case for Linear and Polynomial Kernels. However, other kernels can perform both linear and nonlinear separations depending on the value of their parameters. Example of these kernels is the Sigmoid which can build linear boundaries when $\theta = -10^{-9}$ and $c = 10$ as shown in Figure 2.8.

Moreover, Voronoï cells show that some kernels have a similar behavior and can detect

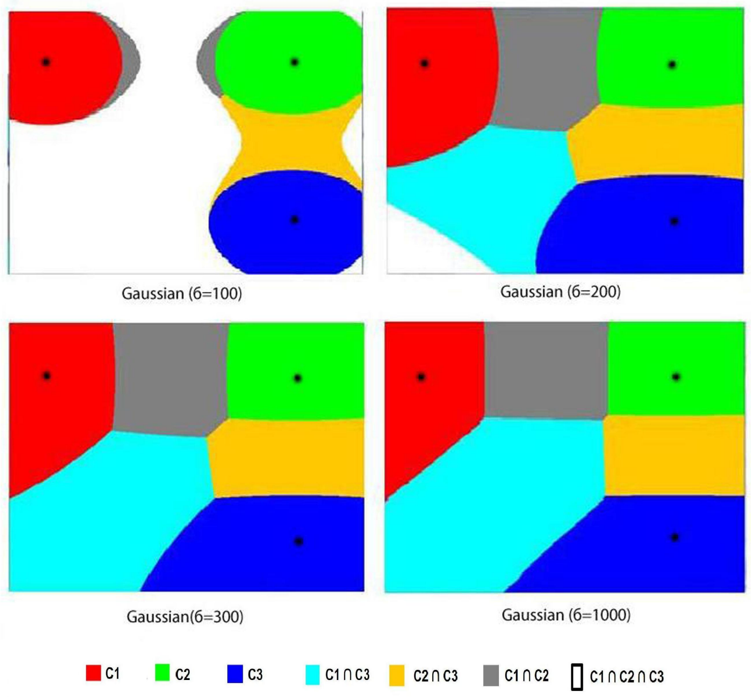


Figure 2.5: Voronoï cells obtained with KOKMII using Gaussian RBF kernel

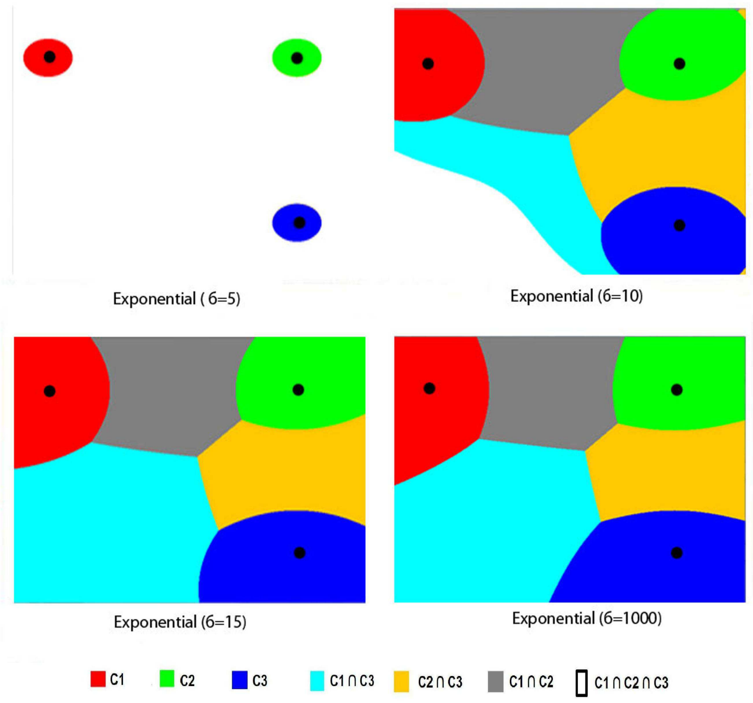


Figure 2.6: Voronoï cells obtained with KOKMII using Exponential kernel

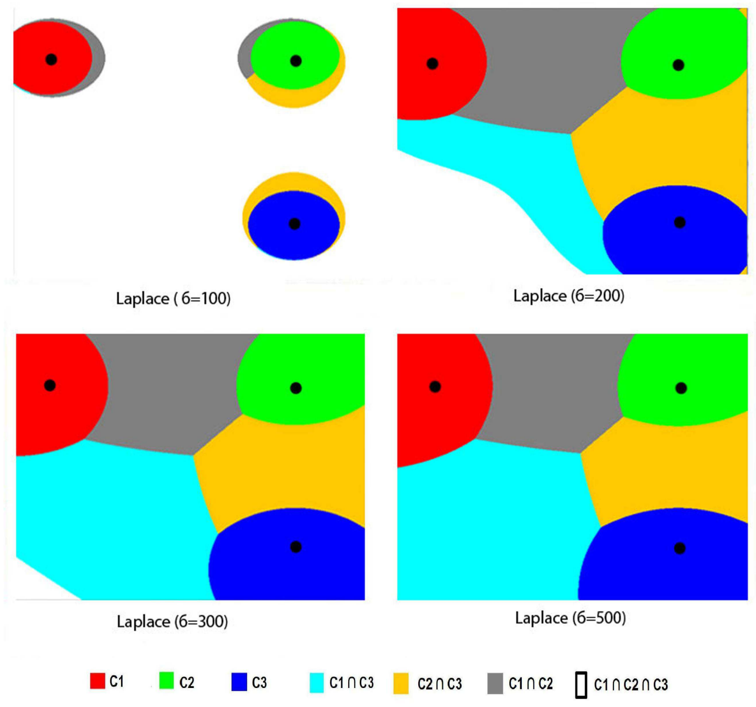


Figure 2.7: Voronoï cells obtained with KOKMII using Laplace kernel

the same patterns: for example the Gaussian Kernel (with $\sigma = 1000$) and the Sigmoid Kernel (with $\theta = -10^{-9}$ and $c = 10$) build identical clusters shapes.

Compared to Voronoï cells obtained with OKM and ALS, those obtained with KOKMII using different kernels prove the ability of the proposed methods to detect overlapping clusters with both linear and nonlinear separations. Hence, the ability of proposed methods to detect more relevant groups, when clusters have a complex shapes, is demonstrated. In the next, these results are checked over artificial and real overlapping datasets.

2.5.3 Empirical results on datasets with non-spherical and nonlinear separations

Datasets description

Experiments are performed on three artificial datasets which are Iris dataset, Test dataset and Ionosphere dataset. Iris[1] dataset is traditionally used as a basis test for evaluation.

[1]cf. http://archive.ics.uci.edu/ml/datasets/Iris.

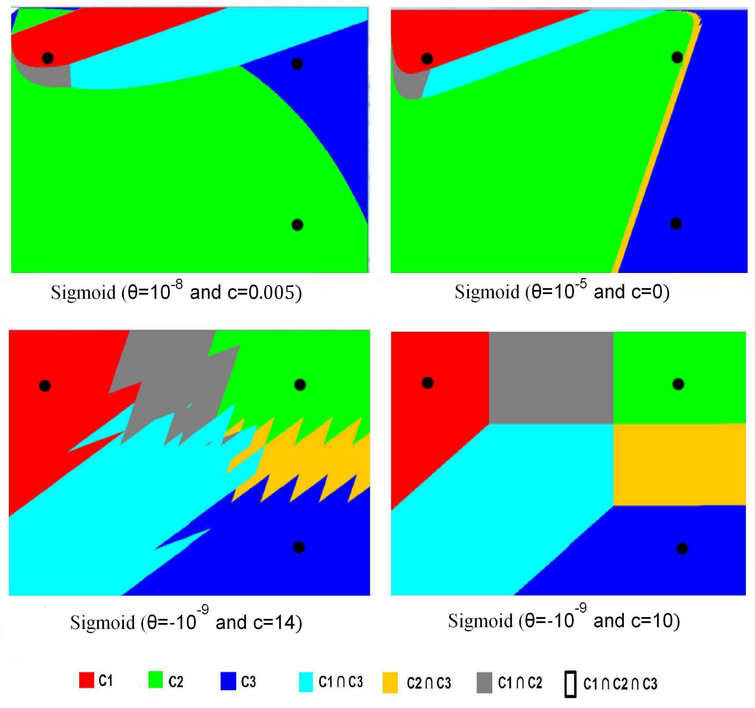

Sigmoid (θ=10^-8 and c=0.005)
Sigmoid (θ=10^-5 and c=0)
Sigmoid (θ=-10^-9 and c=14)
Sigmoid (θ=-10^-9 and c=10)
C1
C2
C3
C1∩C3
C2∩C3
C1∩C2
C1∩C2∩C3

Figure 2.8: Voronoï cells obtained with KOKMII using Sigmoid Kernel

It is composed of 150 data in $\mathbb{R}^4$ tagged according to three non-overlapping clusters with 50 observations per class. One of these clusters "setosa" is known to be clearly separated from the two others which are difficult to learn.

The second dataset is Test dataset which has a simple structure with two non-spherical clusters. All observations (14 observations) are randomly chosen from unit square $[0, 1] \times [0, 1]$ in a two dimensional space with the uniform distribution. Observations that fall in the kernel area with center $(0.5, 0.5)$ and radius 0.4 are assigned label $+1$. Those in the complement are assigned label -1. The particularity of this dataset is that corresponding clusters are non-linearly-separable with non-spherical shapes. Algorithms based k-means fail to determine these clusters. When these data are mapped to a higher feature space using RBF kernel, the clusters lose their circular shapes and become easier to separate as shown in Figure 2.9(a). If data are shown from another angle, the circular shapes of clusters are there as reported in Figure 2.9(b).

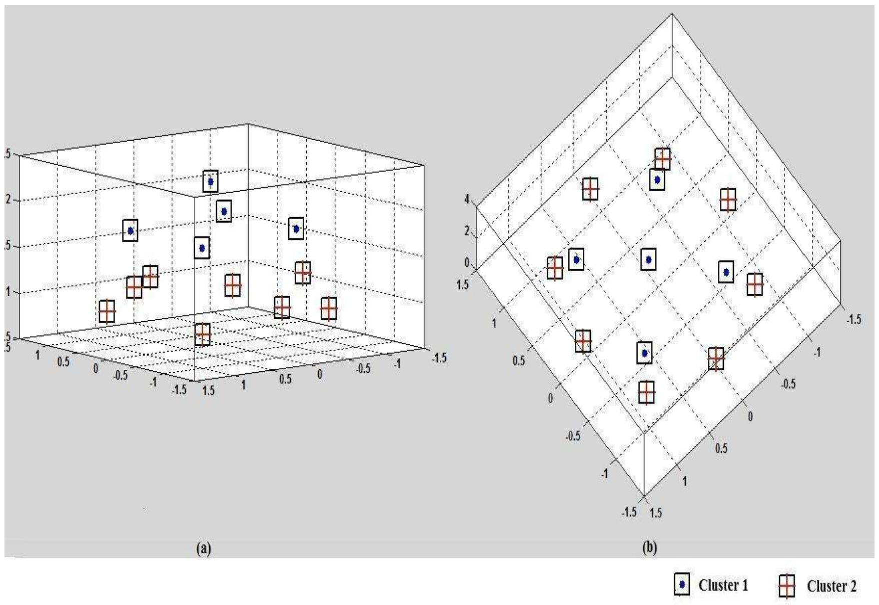


Figure 2.9: 3D plot of Test dataset using the three principal axis obtained with PCA method: data in feature space using RBF kernel

The third dataset is Ionosphere[2], built by a radar system in Goose Bay Labrador. This system analyzes the electrons in the ionosphere, where some electrons show a certain type of structure. These electrons determine the first class in the dataset that is labeled “good”. Other electrons, with no structure in ionosphere, define the second class in the dataset that is labeled “bad”. Electrons are transmitted from antennas via a signal, which is described by 34 attributes that will constitute the size of the dataset Ionosphere. The total number of signals in the dataset is 351 signals. The characteristic of this dataset is that the two classes have circular shapes that are difficult to separate by linear clustering algorithms as shown in Fig. 2.10. When these data are mapped to higher dimensional space using RBF kernel, it will be easier to find separations between good and bad electrons. The two classes lose their circular shapes and become linearly separable.

Empirical results

We compared the performance of the proposed methods with 5 existing methods: k-means, kernel k-means, fuzzy c-means, OKM and ALS. Table 2.2 reports the average of

[2]cf. http://archive.ics.uci.edu/ml/datasets/Ionosphere.

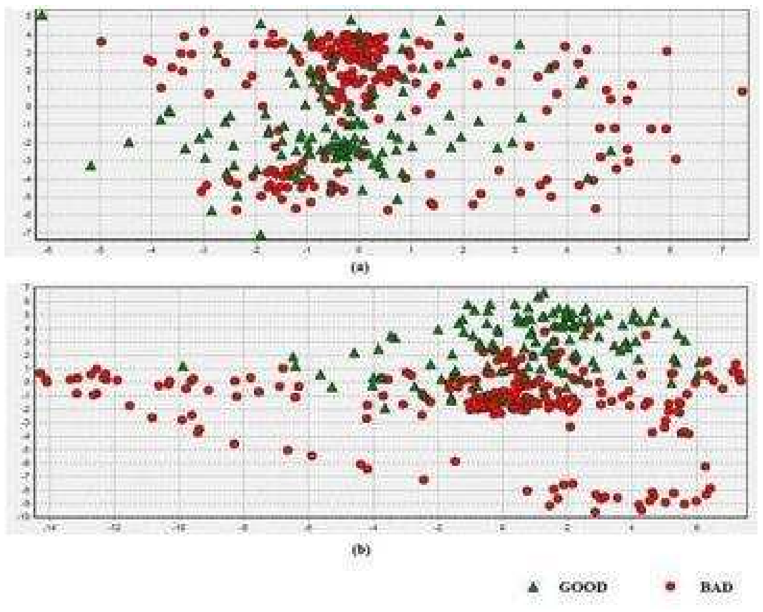


Figure 2.10: 2D plot of Ionosphere dataset using the first and second principal axis obtained with PCA method: (a) data in input space (b) data in feature space

Precision (P), Recall (R) and F-measure (F) on twenty runs while Table 2.3 reports the average size of overlaps obtained using the same methods. For each run, all methods have the same initialization of prototypes.

Compared to existing overlapping methods, F-measures obtained with KOKMI and KOKMII outperform F-measures obtained with OKM and ALS using Euclidean distance for all datasets. The improvement in classification results is achieved in terms of Precision and Recall. The improvement is important in Test dataset where OKM and ALS fail

Table 2.2: Comparison of the performance of KOKMI and KOKMII versus other existing methods for non overlapping datasets

Dataset Label	**Iris dataset**			**Test dataset**			**Ionosphere dataset**		
	P	R	F	P	R	F	P	R	F
k-means	0.794	0.790	**0.790**	0.540	0.580	0.559	0.621	0.591	0.605
Kernel K-means	0.822	0.824	**0.822**	1	1	**1**	0.710	0.715	**0.712**
Fuzzy c-means	0.800	0.807	**0.800**	0.499	0.550	0.523	0.633	0.602	0.613
OKM	0.594	0.973	0.738	0.307	0.500	0.381	0.459	0.791	0.581
ALS	0.520	0.997	0.680	0.562	0.692	0.620	0.420	0.902	0.531
KOKMI with RBF kernel	0.704	0.920	**0.792**	0.940	1	**0.969**	0.662	0.745	**0.701**
KOKMII with RBF kernel	0.704	0.956	**0.807**	0.940	1	**0.969**	0.557	0.702	0.621

to determine clusters with concentric shapes. Figure 2.11 shows structures of patterns obtained by k-means, fuzzy c-means, OKM, kernel k-means, KOKMI and KOKMII in Test dataset. Figures 2.11(b,e,f) show that methods incorporating kernel can detect clusters with circular shapes. However, k-means and OKM fail to detect these clusters as shown in Figures 2.11(a,d). These results show the ability of KOKMI and KOKMII to achieve nonlinear separations between clusters and then their performance in detecting clusters with complex separations.

Table 2.3: Size of overlaps obtained with KOKMI, KOKMII and other methods for non overlapping datasets

	Size of Overlap		
	Iris dataset	Test dataset	Ionosphere dataset
Real overlap size	(1)	(1)	(1)
k-means	1	1	1
Kernel K-means	1	1	1
Fuzzy c-means	1	1	1
ALS	1.37	1.23	1.45
OKM	1.34	1	1.39
KOKMI with RBF kernel	1.15	1.07	1.28
KOKMII with RBF kernel	1.22	1.07	1.25

In fact, methods using kernel approach yield better results than traditional non kernel methods. For hard methods, Kernel k-means outperforms k-means and fuzzy c-means. For overlapping methods, KOKMI and KOKMII outperform OKM and ALS. These empirical results prove the theoretical finding that looking for separations between clusters in a high dimensional space is better than looking for separations in an Euclidean input space either for hard or for overlapping methods.

Table 2.2 shows that hard methods, specifically kernel k-means, give better classification results than overlapping methods. This usefulness of hard methods is explained by the type of data where all datasets are non overlapping (overlap size= 1). Classification results obtained with overlapping methods are characterized by a low Precision because observations are assigned to more than one cluster. Although all datasets do not contain overlapping observations, all overlapping methods build an overlap size greater than 1 as reported in Table 2.3. In fact, the size of overlaps affects the value of the F-measure. As the obtained size of overlaps increases, the value of Precision decreases inducing the decrease of F-measure.

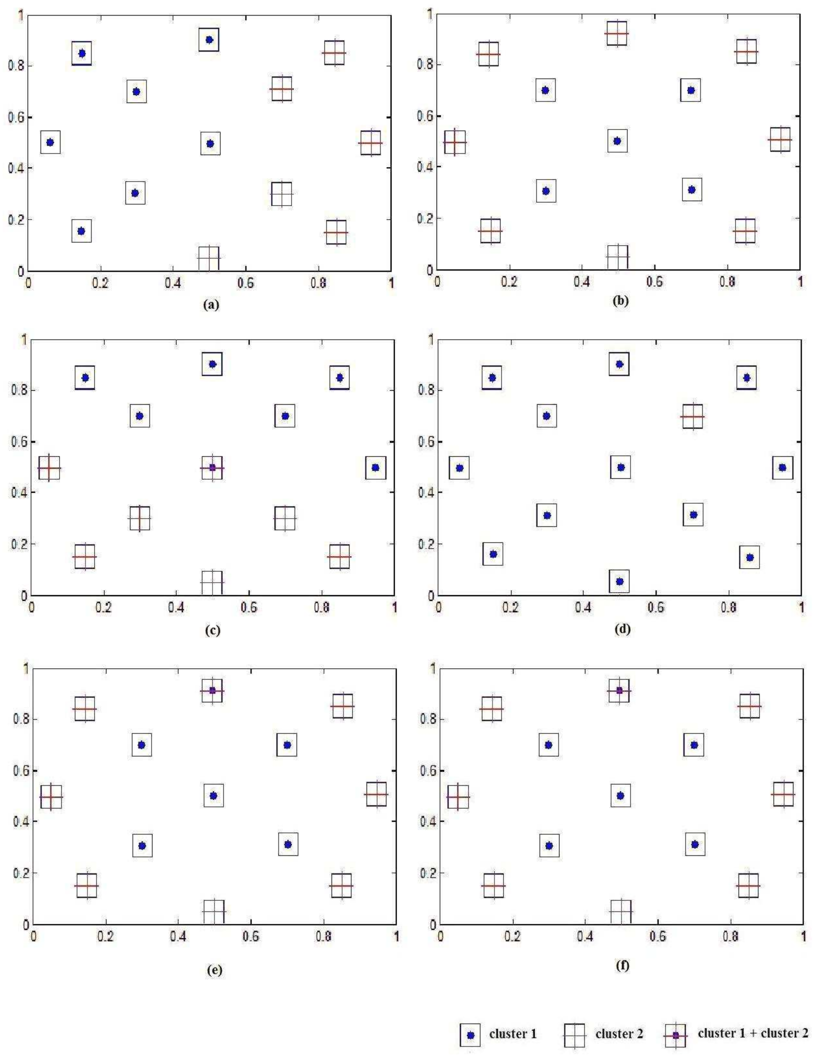


Figure 2.11: Clusters obtained with different methods on Test dataset : (a) Clusters obtained with k-means (Euclidean distance), (b) Clusters obtained with kernel k-means (RBF kernel), (c) Clusters obtained with fuzzy c-means (threshold=0.4), (d) Clusters obtained with OKM (Euclidean distance), (e) Clusters obtained with KOKMI (RBF kernel), (f) Clusters obtained with KOKMII (RBF kernel)

In the next section, we study the effectiveness of these methods over real multi-labeled datasets having different degrees of natural overlaps.

2.5.4 Empirical results on real multi-labeled datasets

We conducted experiments on real overlapping datasets from three domains that strongly motivate overlapping clustering researches: video classification, detection of emotions in music and image classification.

Overlapping datasets description

The first overlapping dataset is EachMovie[3] dataset, containing user ratings for each movie in the collection. Users give ratings on a scale of 1 to 5, with 1 indicating extreme dislike and 5 indicating strong approval. For each movie, the corresponding genre information is extracted from the Internet Movie Database (IMDB) collection. If each genre is considered as a separate category or cluster, then this dataset has naturally overlapping clusters since many movies are annotated in IMDB as belonging to multiple genres (Banerjee et al. 2005). For example, Aliens movie belongs to three genres: action, horror and science fiction.

From the EachMovie dataset, we extracted a subset of 75 movies scattered over three overlapping clusters as follows: "action" = 21 movies; "comedy" = 26 movies; "crime"= 17 movies; "action+crime"= 11 movies. Based on age, sex and rate of users we try to find categories for each video. Figure 2.12 shows the initial distribution of these movies where overlapping movies belong to both action and crime genres. When mapping the same movies into a higher feature space, overlapping movies are easily detected while there are geometrically laying in the extremity surface between action and crime movies. In addition, separations between "crime" and "comedy" movies become easier to detect compared to their first distribution.

The second overlapping dataset is Music emotion [4]dataset. Emotion detection in music can be realized by analyzing music signals. The emotion labels are not usually disjoint in the sense that a single music sound may be classified simultaneously into multiple emotional categories e.g. both "happy" and "relaxing" (Trohidis et al. 2008). This

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

Table 2.4: Distribution of songs by six principal class labels in Music dataset: Amazed Surprised, Happy Pleased, Relaxing Calm, Quite Still, Sad Lonely and Angry Aggressive

Label	number of songs
Amazed Surprised	173
Happy Pleased	166
Relaxing Calm	264
Quite Still	148
Sad Lonely	168
Angry Aggressive	189
Total	**593**

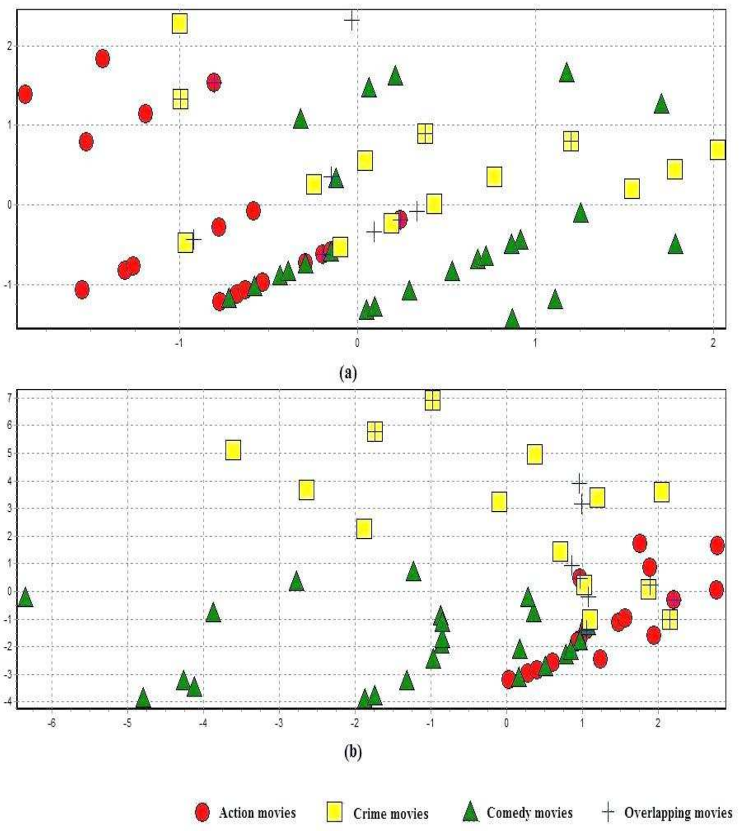


Figure 2.12: 2D plot of EachMovie dataset using the first and the second principal axes obtained with PCA: (a) data in input space (b) data in feature space

stipulation seems to make the Music dataset naturally overlapping. The Music dataset contains sound clips described by 72 real attributes and annotated by three male experts. This process led to a final annotated dataset of 593 songs with 6 main emotional clusters as described in Table 3.3. The overlap between clusters is important in Music dataset. For example, all music sounds that evoke “Quite Still” emotion also evoke “Relaxing Calm” or/and “Sad Lonely” emotions.

The third overlapping dataset is the Scene [5] dataset that contains 2407 natural scene images. Each image is transformed into a $49 \times 3 \times 2 = 294$ dimensional features vectors. Table 2.5 gives the detailed description of the number of images associated with different label sets, where all the possible class labels are Beach, Sunset, Fall foliage, Field, Mountain and Urban. Over 8% of images belong to multiple classes simultaneously such as images belonging to beach and mountain as illustrated in Fig. 2.13. On average, each image is associated with 1.08 class labels.

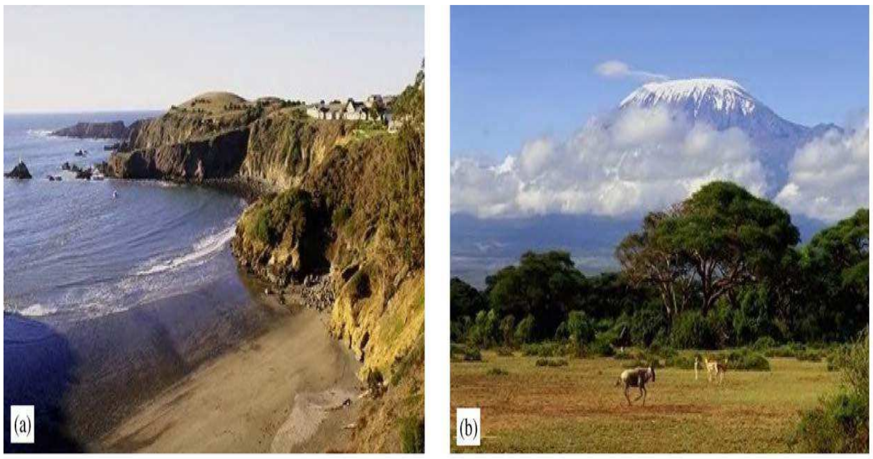


Figure 2.13: Example of overlapping images in Scene dataset: **(a)** image associated to Beach and Mountain **(b)** image associated to Field and Mountain (Rokach 2004)

Empirical results

For datasets described in Section 2.5.3, Table 2.7 presents results obtained with KOKMI and KOKMII versus k-means, kernel k-means, fuzzy c-means, OKM and ALS methods in terms of Precision, Recall and F-measure. Each reported result is an average over twenty runs of each algorithm. Results of KOKMI in Music and Scene datasets are not reported due to a time difficulty where execution needs more than 24 hours, however it is easily

[5]cf.http://mulan.sourceforge.net/datasets.html

Table 2.5: Distribution of images by six principal class labels in Scene dataset: Beach, Sunset, Fall foliage, Field, Mountain and Urban

Label	**number of images**
Beach	369
Sunset	364
Fall foliage	360
Field	327
Beach+Field	1
Fall foliage+Field	23
Mountain	405
Beach+Mountain	38
Fall foliage+Mountain	13
Field+Mountain	75
Field+Fall foliage+Mountain	1
Urban	405
Beach+Urban	19
Field+Urban	6
Mountain+Urban	1
Total	**2407**

solved with KOKMII. A Comparison of runtime required for KOKMI and KOKMII to return the final clusters in different datasets are given in Table 2.6. The improvement with KOKMII is shown in terms of computational complexity and in terms of classification results.

Table 2.6: Comparison of the run time of KOKMI and KOKMII methods

Dataset	**#Observations**	**#Labels**	**KOKMI**	**KOKMII**
Iris	150	3	17.52 Seconds	**0.83** Seconds
Ionosphere	351	2	2380.31 Seconds	**3.45** Seconds
Eachmovie	75	3	3.46 Seconds	**0.52** Seconds
Music	593	6	>24 Hours	**8.41** Seconds
Scene	2407	6	>24 Hours	**300.23** Seconds

Table 2.7 shows that, for all datasets, F-measures obtained with the proposed methods are better than those obtained with existing ones. Compared to overlapping methods, the improvement of the F-measure is induced by the improvement of Precision. There is no large difference between overlapping methods in terms of Recall because all of them assign observations to many clusters. However, in terms of Precision, KOKMII has the best values with a large margin compared to other overlapping methods. These results prove

Table 2.7: Comparison of the performance of KOKMI and KOKMII versus other existing methods for overlapping datasets

Dataset Label	**Eachmovie**			**Music**			**Scene**		
	P	R	F	P	R	F	P	R	F
K-means	0.546	0.538	0.542	0.515	0.177	0.263	0.399	0.522	0.452
Kernel k-means	0.556	0.542	0.549	0.520	0.181	0.270	0.401	0.533	**0.571**
Fuzzy c-means ($\theta = 1/k$)	0.605	0.683	0.623	0.480	0.354	0.408	0.102	0.946	0.185
OKM	0.399	0.912	0.555	0.353	0.544	**0.428**	0.192	0.926	0.216
ALS	0.515	0.779	0.620	0.307	0.940	**0.436**	-	-	-
KOKMI with RBF kernel	0.650	0.722	**0.671**	-	-	-	-	-	-
KOKMII with RBF kernel	0.698	0.731	**0.714**	0.410	0.496	**0.449**	0.390	0.791	**0.510**

that our proposed methods can detect more relevant and complex separations between clusters leading to the improvement of the classification precision.

Table 2.8: Size of overlaps obtained with KOKMI, KOKMII and other methods for overlapping datasets

	Size of Overlap		
	Eachmovie dataset	Music dataset	Scene dataset
Actual overlap size	(1.14)	(1.81)	(1.08)
k-means	1	1	**1**
Kernel K-means	1	1	**1**
Hard Fuzzy c-means	1	1	**1**
OKM	1.40	2.35	2.85
ALS	1.73	3.46	-
Fuzzy c-means ($\theta = 1/k$)	**1.26**	1.43	3.34
Fuzzy c-means ($\theta = 0.3$)	**1.26**	1.22	0.00
Fuzzy c-means ($\theta = 0.4$)	0.93	0.97	0.00
KOKMI with RBF kernel	**1.26**	-	-
KOKMII with RBF kernel	1.17	**1.98**	**1.99**

In addition, reported results show that the size of actual overlaps in data affects the performance of overlapping methods. As the size of overlaps in the dataset increases, the performance of overlapping methods compared to hard methods becomes more noticeable. For example, in Music dataset, where actual overlaps is 1.81, all F-measures obtained with OKM, ALS and KOKMII outperform results of k-means and kernel k-means. However, when the size of overlaps nearly reaches 1, such as in Scene dataset, F-measures obtained with hard methods outperform, or at at least equal, those of overlapping methods. For example, F-measure obtained with kernel k-means in Scene dataset is 0.570 which outperforms F-measures obtained with OKM and ALS. In fact, high values of F-measure obtained with hard methods are induced by high values of Precision, while for overlapping methods, high values of F-measure are induced by high values of Recall.

Table 2.9: Validation measures obtained with KOKMII using different types of kernels in Eachmovie Dataset

Kernel	**Value**	**Precision**	**Recall**	**F-measure**	**Overlap**
Gaussian	$\sigma = 1$	0.426 ± 0.06	0.788 ± 0.15	0.553 ± 0.11	1.53 ± 0.10
	$\sigma = 10$	0.698 ± 0.05	0.731 ± 0.15	0.714 ± 0.08	1.17 ± 0.05
RBF kernel	$\sigma = 15$	0.698 ± 0.05	0.731 ± 0.15	0.714 ± 0.08	1.17 ± 0.05
	$\sigma = 100$	0.447 ± 0.05	0.676 ± 0.15	0.559 ± 0.08	1.24 ± 0.05
Exponential	$\sigma = 1$	0.277 ± 0.03	0.928 ± 0.11	0.419 ± 0.09	1.96 ± 0.12
	$\sigma = 5$	0.338 ± 0.06	0.846 ± 0.21	0.483 ± 0.15	1.46 ± 0.15
RBF kernel	$\sigma = 10$	0.338 ± 0.06	0.846 ± 0.21	0.483 ± 0.15	1.46 ± 0.15
	$\sigma = 100$	0.338 ± 0.06	0.846 ± 0.21	0.483 ± 0.15	1.46 ± 0.15
	$\sigma = 10000$	0.338 ± 0.06	0.846 ± 0.21	0.483 ± 0.15	1.46 ± 0.15
Laplace	$\sigma = 1$	0.208 ± 0.08	0.938 ± 0.14	0.340 ± 0.09	2.33 ± 0.03
	$\sigma = 10$	0.490 ± 0.12	0.820 ± 0.11	0.613 ± 0.07	1.50 ± 0.13
kernel	$\sigma = 20$	0.388 ± 0.13	0.846 ± 0.11	0.483 ± 0.08	1.46 ± 0.13
	$\sigma = 100$	0.388 ± 0.13	0.846 ± 0.11	0.483 ± 0.08	1.46 ± 0.13
	$\sigma = 10000$	0.388 ± 0.13	0.846 ± 0.11	0.483 ± 0.08	1.46 ± 0.13
	$d = 1$	0.423 ± 0.09	0.690 ± 0.11	0.524 ± 0.10	1.42 ± 0.08
Polynomial kernel	$d = 2$	0.389 ± 0.08	0.756 ± 0.14	0.514 ± 0.10	1.49 ± 0.02
	$d = 3$	0.418 ± 0.02	0.667 ± 0.13	0.517 ± 0.07	1.36 ± 0.11
	$d = 4$	0.418 ± 0.02	0.667 ± 0.13	0.517 ± 0.07	1.36 ± 0.11

Table 2.10: Validation measures obtained with KOKMII using different types of kernels in Music dataset

Kernel	**Value**	**Precision**	**Recall**	**F-measure**	**Overlap**
Gaussian	$\sigma = 10$	0.370 ± 0.03	0.571 ± 0.00	0.449 ± 0.00	2.64 ± 0.00
	$\sigma = 15$	0.423 ± 0.00	0.445 ± 0.02	0.434 ± 0.02	1.96 ± 0.03
RBF kernel	$\sigma = 100$	0.445 ± 0.02	0.312 ± 0.02	0.370 ± 0.03	1.43 ± 0.01
	$\sigma = 1000$	0.469 ± 0.08	0.288 ± 0.02	0.357 ± 0.02	1.48 ± 0.02
Exponential	$\sigma = 5$	0.388 ± 0.02	0.475 ± 0.02	0.427 ± 0.02	1.96 ± 0.00
	$\sigma = 10$	0.396 ± 0.02	0.393 ± 0.02	0.395 ± 0.02	1.88 ± 0.00
RBF kernel	$\sigma = 15$	0.398 ± 0.00	0.387 ± 0.02	0.392 ± 0.02	1.86 ± 0.01
	$\sigma = 100$	0.402 ± 0.01	0.383 ± 0.02	0.392 ± 0.02	1.84 ± 0.03
Laplace	$\sigma = 10$	0.167 ± 0.04	0.763 ± 0.03	0.274 ± 0.04	3.36 ± 0.01
	$\sigma = 20$	0.279 ± 0.02	0.646 ± 0.02	0.389 ± 0.02	2.76 ± 0.01
kernel	$\sigma = 100$	0.394 ± 0.02	0.385 ± 0.01	0.389 ± 0.01	1.85 ± 0.05
	$\sigma = 500$	0.415 ± 0.02	0.401 ± 0.01	0.408 ± 0.01	1.71 ± 0.05
	$\sigma = 10000$	0.413 ± 0.02	0.362 ± 0.02	0.386 ± 0.01	1.77 ± 0.05
	$d = 1$	0.437 ± 0.01	0.323 ± 0.02	0.372 ± 0.01	1.55 ± 0.01
Polynomial kernel	$d = 2$	0.437 ± 0.00	0.319 ± 0.01	0.369 ± 0.01	1.54 ± 0.02
	$d = 3$	0.441 ± 0.00	0.337 ± 0.01	0.382 ± 0.01	1.49 ± 0.01
	$d = 4$	0.441 ± 0.00	0.337 ± 0.01	0.382 ± 0.01	1.49 ± 0.01

Therefore, knowing the actual overlaps in each dataset, the sizes of overlaps built by each method are discussed. Table 2.8 summarizes overlaps obtained with KOKMI and

Table 2.11: Validation measures obtained with KOKMII using different types of kernels in Scene dataset

Kernel	**Value**	**Precision**	**Recall**	**F-measure**	**Overlap**
Gaussian	$\sigma = 10$	0.419 ± 0.04	0.663 ± 0.05	0.513 ± 0.04	1.75 ± 0.04
	$\sigma = 15$	0.418 ± 0.04	0.655 ± 0.05	0.510 ± 0.05	1.75 ± 0.04
RBF kernel	$\sigma = 100$	0.423 ± 0.05	0.657 ± 0.05	0.511 ± 0.05	1.75 ± 0.04
	$\sigma = 10000$	0.423 ± 0.05	0.657 ± 0.05	0.511 ± 0.05	1.75 ± 0.04
Exponential	$\sigma = 10$	0.425 ± 0.01	0.750 ± 0.04	0.548 ± 0.02	1.99 ± 0.00
	$\sigma = 15$	0.426 ± 0.01	0.753 ± 0.05	0.544 ± 0.02	1.99 ± 0.00
RBF kernel	$\sigma = 100$	0.425 ± 0.01	0.753 ± 0.04	0.544 ± 0.02	1.99 ± 0.00
	$\sigma = 10000$	0.426 ± 0.01	0.753 ± 0.05	0.544 ± 0.02	1.99 ± 0.00
Laplace	$\sigma = 1$	0.208 ± 0.08	0.938 ± 0.14	0.340 ± 0.09	2.33 ± 0.03
	$\sigma = 10$	0.490 ± 0.12	0.820 ± 0.11	0.613 ± 0.07	1.50 ± 0.13
kernel	$\sigma = 20$	0.388 ± 0.13	0.846 ± 0.11	0.483 ± 0.08	1.46 ± 0.13
	$\sigma = 100$	0.388 ± 0.13	0.846 ± 0.11	0.483 ± 0.08	1.46 ± 0.13
	$\sigma = 10000$	0.388 ± 0.13	0.846 ± 0.11	0.483 ± 0.08	1.46 ± 0.13
	$d = 2$	0.435 ± 0.04	0.661 ± 0.06	0.525 ± 0.05	1.75 ± 0.05
Polynomial kernel	$d = 3$	0.455 ± 0.05	0.656 ± 0.05	0.537 ± 0.05	1.74 ± 0.05
	$d = 4$	0.481 ± 0.03	0.593 ± 0.02	0.530 ± 0.02	1.63 ± 0.14

KOKMII compared to overlaps obtained with existing overlapping and non overlapping methods. For hard methods, all sizes of overlaps are equal to 1 since these methods build non-disjoint clusters and ignore the possibility that an observation belongs to more than one cluster which is the case for all these datasets. Fuzzy c-means builds acceptable overlaps if the threshold is well determined, elsewhere we can obtain an overlap size less than 1. For overlapping methods, we notice that OKM and ALS build large overlap sizes. For example, in Music dataset, the size of overlaps obtained with OKM and ALS are 2.35 and 3.46 respectively, while actual overlaps in this dataset is 1.81. However, for the same dataset, KOKMII builds an acceptable overlap size of 1.98. These results prove the capacity of the proposed methods to build acceptable sizes of overlaps given actual overlaps in each dataset.

As described in Section 2.5.2, the structure of separations between overlapping clusters depends of the choice of the kernel function ant its parameters. Therefore, we study the sensitivity of proposed methods to these parameters by analyzing the performance of KOKMII in real overlapping datasets using Gaussian, Exponential and Polynomial kernels with different values of kernel parameters.

Table 2.9, 2.10 and 2.11 report average results and standard deviations, on ten runs, obtained with KOKMII using different kernels in Eachmovie, Music and Scene datasets. From these results we notice that kernels performing nonlinear separations, such as Exponential and Gaussian, give better results than those performing linear separations like the Polynomial kernel. These results state that in real life applications where separations between clusters may be complex, it would be better to perform a learning process with nonlinear separations. We also notice that kernel-based methods are highly sensitive to the choice of the kernel function. For example, in Scene dataset, the best obtained F-measure with Exponential kernel is 0.548; whereas this value does not exceed 0.513 using Gaussian kernel. Moreover, reported results show that initializing the parameters of the kernel function can considerably affect the performance of clustering. Some kernel parameters have identical behaviors in all datasets. For example, using Gaussian or Exponential kernels, the size of overlaps increases, Precision decreases and Recall increases when σ becomes large. Finally, the reported results of standards deviations on ten runs with different initializations of clusters representatives, prove that KOKMII is not very sensitive to these initializations and converges to nearly the same partitions.

2.6 Conclusion

In order to better look for overlapping clusters with nonlinear and non spherical separations two clustering methods, based Mercer kernel, are proposed. The first method, based on centroids, generalizes kernel k-means for overlapping clustering. The second method, based on medoids, improves the clustering accuracy of the previous method and adapts it to large datasets. Experiments on artificial and real overlapping datasets show the ability of these methods to detect clusters with complex separations and their efficiency compared to other overlapping clustering methods.

Chapter 3

Generic model for non-disjoint clustering with overlap regulation

Contents

3.1 Introduction

Existing overlapping clustering methods are able to produce a non-disjoint partitioning of data. However, most of these methods produce clusters with a fixe and large sizes of overlaps which lead to produce more information than needed. This fact reduces considerably the precision of classification of existing methods. To deal with this issue, we present in this chapter a new clustering model (BenN'Cir et al. 2014b) that generalizes k-means to detect overlapping clusters with parameterizable and auto-adjusted sizes of overlaps. We propose different instantiations (BenN'Cir et al. 2014a) of the proposed model which produce different layouts for the overlapping boundaries between clusters. These instantiations are well adapted for the regulation of the sizes of the overlaps.

This chapter is organized as follows: Section 3.2 describes the motivation of this work by presenting the requirement of detecting clusters with regularized overlaps. Section 3.3 and Section 3.4 respectively present a first approach that we propose and the generalization of this approach for detecting overlapping clusters with possibility of control of the overlaps. Experiments on different datasets are described and discussed in Section 3.5. Finally, Section 3.6 presents conclusions

3.2 Motivation: control of overlapping boundaries between clusters

Into a knowledge discovery process, the user or expert is central and should have the means to interact with the system; as well as he examines several possibilities on the number of clusters, the metric to use or the fuzziness of the solution in a clustering process, he must be able to regulate the size of the overlaps when such an overlapping structuring is expected. Usually, the existing overlapping clustering methods produce a large overlaps between clusters. Although the overlapping clustering task reconsiders the "well separated clusters" property, clusters with too large overlaps are not appropriate for most of the target applications. The overlapping clustering hypothesis tolerates overlaps between clusters, but the size of the overlaps should be controlled, depending on the requirements of the application.

Therefore, we study patterns and overlaps sizes produced by the existing OKM method, we build Voronoï cells induced by the objective function of OKM for three clusters. Figure 3.1 shows an example of these Voronoï cells. The representation space is divided into several areas where each possible combination of clusters is associated to one area except the empty set. Each area is centered on a prototype or a combination of prototypes. We notice the large overlapping boundaries which is not appropriate to detect clusters with small overlaps. For example, the size of the area "cluster1∩3" is very large compared to areas "cluster1" and "cluster3". The produced patterns do not match with the overlapping clustering hypothesis where overlaps between clusters should not be as important as the *single* clusters.

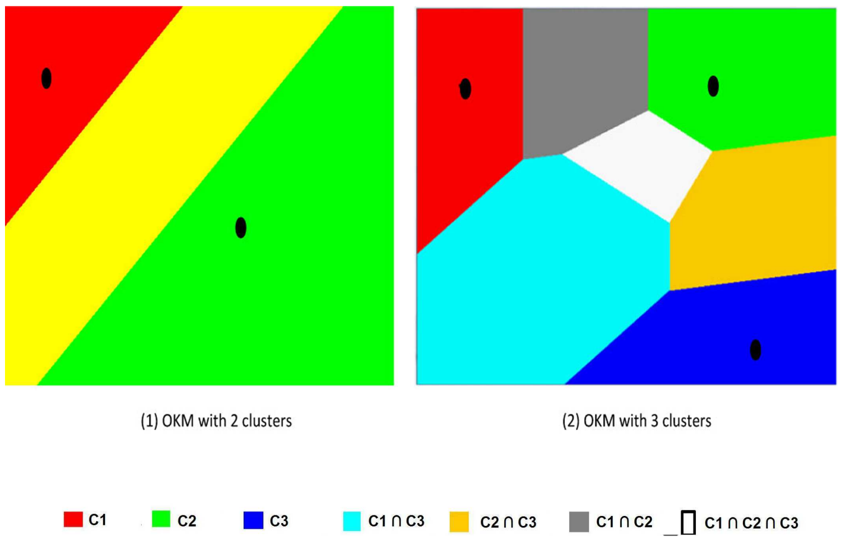


Figure 3.1: Voronoï cells obtained with OKM for two and three clusters.

Because the expected clusterings can have different levels of overlaps depending on the requirement of the application, we focused our study on k-means and on the geometrical model which will be generalized in order to produce overlaps with fixed (but limited), parameterizable or auto-adjusted sizes. The problem of overlap regulation for clustering approaches is considered for the geometrical model, without loss of generality since similar regulation principles could be performed on additive models.

3.3 Proposed Restricted OKM method

To make the overlapping regulation feasible for geometrical model, we firstly propose the R-OKM method (BenN'Cir and Essoussi 2012a) which formalizes a regulation principle based on the number of clusters concerned by the data assignment. Instead of considering strong global thresholds that could for example limit or favor for any data the number of its memberships to a given value regardless of the data context, the regulation process we propose is data sensitive ; it aims to parameterize the expected benefit of an overlap on the local errors. In fact, the large overlap boundaries built by OKM method are induced by the model used that consists in dividing the representation space of observations into multiple areas, the Voronoï cells, where each observation is assigned to its nearest area. For example, if we have three clusters, there exists seven areas associated respectively to clusters $\{\pi_1\}$, $\{\pi_2\}$, $\{\pi_3\}$, $\{\pi_1,\pi_2\}$,$\{\pi_1,\pi_3\}$, $\{\pi_2,\pi_3\}$ and $\{\pi_1,\pi_2,\pi_3\}$. All the observations x_i in each area are closer to the prototype or the combination of prototypes of this area. In the actual OKM model, given two assignments Π_i and Π'_i of an observation x_i, the assignment Π_i is preferred if and only if $d^2(x_i, im_{\Pi,C}(x_i)) \leq d^2(x_i, im_{\Pi',C}(x_i))$ with $im_{\Pi,C}(x_i)$ denotes [1] the combination of clusters' representatives to which x_i belongs to.

To produce overlapping clusters with limited sizes of overlaps, the areas corresponding to combinations of clusters (for example areas $\{\pi_1,\pi_2\}$, $\{\pi_1,\pi_3\}$, $\{\pi_2,\pi_3\}$ and $\{\pi_1,\pi_2,\pi_3\}$) should be reduced as well as the number of combined clusters increases. A natural solution to restrict the assignment of observations is to introduce a weight when evaluating distances between an observation and it's image. The weight should be relative to the number of clusters that participate to the considered overlaps. The preference rule when building assignment vector of each observation is transformed to:

$$d^2(x_i, im_{\Pi,C}(x_i)) \leq \eta . d^2(x_i, im_{\Pi',C}(x_i)). \tag{3.1}$$

[1]The notation $im_{\Pi,C}(x_i)$ is equivalent to the notation $\overline{x_i}$ used in chapter 1 when we described the combination of clusters' representatives for OKM.

where the coefficient $\eta \in]0,1[$ for $|\Pi'_i| \leq |\Pi_i|$ is set to $\eta = \frac{|\Pi'_i|}{|\Pi_i|}$.

For example, when evaluating assignment of an observation x_i that belongs to two clusters ($|\Pi_i| = 2$) with another assignment of this observation that belongs to only one cluster ($|\Pi'_i| = 1$), the assignment Π_i is preferred if and only if $d^2(x_i, im_{\Pi,C}(x_i)) \leq \frac{1}{2}.d^2(x_i, im_{\Pi',C}(x_i))$.

The proposed weight to penalize assignment of each observation can be integrated in the objective function by introducing the assignment cardinality of each observation as follows:

$$\begin{aligned} J(\Pi, C) &= \sum_{x_i \in X} |\Pi_i|. \parallel x_i - im_{\Pi,C}(x_i) \parallel^2 \\ &= \sum_{c=1}^{k} \sum_{x_i \in \pi_c} \parallel x_i - im_{\Pi,C}(x_i) \parallel^2 . \end{aligned} \tag{3.2}$$

To give the reader a visual reading of the regulation principle, we illustrate the overlaps with Voronoï cells. Figure 3.2 shows the Voronoï cells obtained with the proposed R-OKM method for the same example that Figure 3.1. The proposed weight penalizes assignment of observations and reduces the size of regions where observations are assigned to more than one cluster. The proposed R-OKM method is considered as a generalization of k-means for detecting overlapping clusters. If observations are assigned to only one cluster, the objective function coincides with the objective function of k-means ($|\Pi| = 1$).

This proposed method builds a fixed, but limited, overlap size according to the number of assignments of each observation. However, in real life applications of overlapping clustering, the size of overlaps is not unique and depends on the requirement of the application. The user must be able to regulate the size of the overlaps when such an overlapping structuring is expected. In contrast, when the expert does not have ay expectations on the size of overlaps, the ideal solution consists in enabling to auto-regulate these sizes based on the existing structures in data. In this way, to make the overlapping regulations feasible for geometrical model, we propose a generic framework that can be instantiated

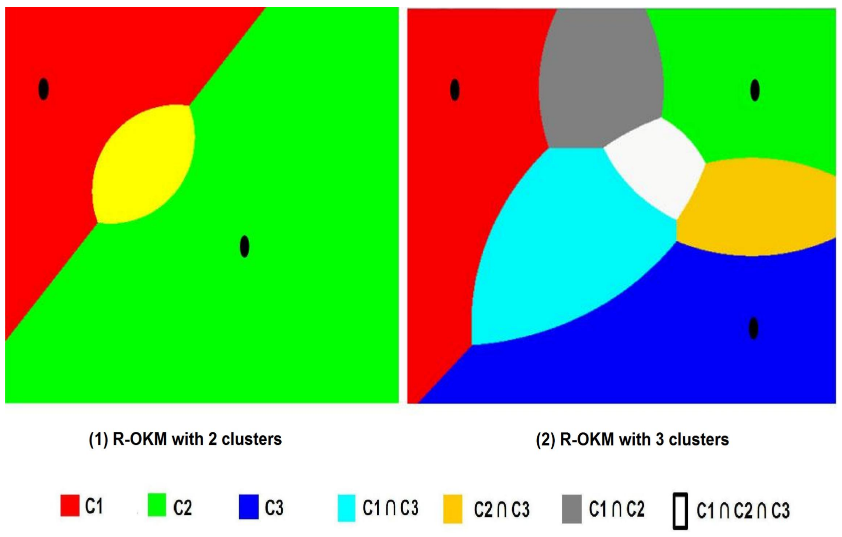


Figure 3.2: Voronoï cells obtained with R-OKM method for two and three clusters.

using different regulation principles.

3.4 Proposed generic model for overlapping clustering with overlap regulation

3.4.1 Objective function considerations

The proposed generic model allows users to detect clusters with non-large overlapping boundaries, to parameterize the size of the overlaps and to auto-regulate overlaps taking into account the real organization of the data. The generic behavior of the model is obtained from the generalization of the weight (ω) associated to each local error :

$$J(\Pi, C, \Omega) = \sum_{x_i \in X} \omega_i . \parallel x_i - im_{\Pi,C}(x_i) \parallel^2 . \tag{3.3}$$

The proposed model can be instantiated in different ways. For example, the model can be instantiated to coincide with OKM when $\omega_i = 1 \quad \forall i$ or to coincide with the proposed R-OKM when $\omega_i = |\Pi_i| \quad \forall i$.

To give the user the possibility to parameterize the size of overlaps between clusters, the model can be instantiated (BenN'Cir et al. 2013a; BenN'Cir et al. 2013b) with

$\omega_i = |\Pi_i|^\alpha$ where α is a parameter to control the size of the overlaps. Depending on the values of α, the model has the following behavior:

- $\alpha = 0$ eliminates the weighting on the combination sizes thus leading to the original OKM model,
- $\alpha > 0$ penalizes the assignments to wide combinations as well as α increases until a non-overlapping model identical to the k-means algorithm ($\alpha \to +\infty$),
- $\alpha < 0$ favors the overlaps as well as α decreases until a trivial clustering scheme with any data assigned to any clusters ($\alpha \to -\infty$).

The proposed generic model can be also instantiated (BenN'Cir et al. 2013b) to take into account the real organization of the data in order to adjust automatically a weight for each cluster. The new weights ω_i can be defined by a combination of weights λ_c, local to each cluster π_c. We define the new weights as $\omega_i = \sum_{c \in \Pi_i} (\lambda_c)^\beta$ under the constraint $\sum_{c=1}^{k} \lambda_c = 1$. The objective function of R-OKM with Adjusted Weights (Adjusted R-OKM) can be described by:

$$J(\Pi, C, \Omega) = \sum_{x_i \in X} \sum_{c \in \Pi_i} (\lambda_c)^\beta . \parallel x_i - im_{\Pi,C}(x_i) \parallel^2, \tag{3.4}$$

where β is a fixed parameter to control the variations in the local weights $\{\lambda_c\}_{c=1}^{k}$. In the following, we consider $\beta \in]0,1[$, noticing that when $\beta \to 0$ any $\lambda_c^\beta \approx \frac{1}{k}$ and the Adjusted R-OKM is equivalent to R-OKM.

3.4.2 Optimization and algorithmic resolutions

The minimization of the objective function of each mentioned instantiation of the model (R-OKM, Parameterized R-OKM and Adjusted R-OKM) is performed by iterating two or three steps according to the number of parameters considered: (1) computation of cluster representatives C, (2) multi assignment (Π) of observations to one or several clusters and (3) computation of weights (Ω). These three steps are independent and the objective function is reduced after each step:

1. **step of the computation of cluster representatives:** considering the assignments (Π) and the local weights (ω) fixed, cluster representatives are successively update by derivation of the objective function as shown in Table 3.1. The notation $\tilde{x_i}^c$ denotes representative m_c according to x_i such that $\parallel x_i - im_{\Pi,C}(x_i) \parallel^2 = 0$ and is computed $\tilde{x_i}^c = |\Pi_i|.x_i - (|\Pi_i| - 1).im_{\Pi\setminus c,C}(x_i)$.

2. **step of the multi-assignments:** the assignment step (considering C and ω fixed) is a combinatorial discrete optimization problem that cannot be solved directly. We present in the following a generic heuristic that makes the objective function minimized.

3. **step of the update of clusters weights:** this step updates the weights considered for each cluster. For R-OKM and Parameterized R-OKM, this step is not required since clusters have equal weights. However, for the Adjusted R-OKM, the local weights (λ) can be obtained independently for each cluster by solving a constraint optimization problem (last column in Table 3.1). We notice that the update of local weights λ_c quantifies the quality of each cluster in terms of optimization of the local errors. Choosing $\beta < 1$ makes the global weight ω_i higher for assignments Π_i composed of clusters with high local errors thus, restricting multi-assignments to these “high quality” clusters.

Algorithm 5[2] and Figure 3.3 describe the generic algorithm of the proposed model for restricted overlapping clustering. The computational complexity of this algorithm is in the order of $O(N.k.\lg k)$ for both R-OKM and Parameterized R-OKM and in the order of $O(N.k^2.\lg k)$ for Adjusted R-OKM. The main algorithm uses the function R_ASSIGN (Regulated Assignments) that defines the assignment strategy. This strategy consists, for each observation x_i, in sorting clusters from closest to farthest with respect to $\parallel x_i - m_c \parallel^2$

[2]Step 5 in Algorithm 1 is used only for Adjusted R-OKM method. For the other methods, while clusters weights are equal this step is not required

Table 3.1: Description of the update rules with different instantiations of the Generic Restricted Overlapping k-means model

Overlapping methods	Instantiations ω_i	Clusters Prototypes m_c^*	Clusters Weights λ_c^*
OKM	1	$\frac{\sum_{x_i\in\pi_c}\frac{1}{\lvert\Pi_i\rvert^2}.\tilde{x_i}^c}{\sum_{x_i\in\pi_c}\frac{1}{\lvert\Pi_i\rvert^2}}$	1
R-OKM	$\lvert\Pi_i\rvert$	$\frac{\sum_{x_i\in\pi_c}\frac{1}{\lvert\Pi_i\rvert}.\tilde{x_i}^c}{\sum_{x_i\in\pi_c}\frac{1}{\lvert\Pi_i\rvert}}$	$\frac{1}{k}$
Parameterized R-OKM	$\lvert\Pi_i\rvert^\alpha$	$\frac{\sum_{x_i\in\pi_c}\frac{1}{\lvert\Pi_i\rvert^{2-\alpha}}.\tilde{x_i}^c}{\sum_{x_i\in\pi_c}\frac{1}{\lvert\Pi_i\rvert^{2-\alpha}}}$	$\frac{1}{k}$
Adjusted R-OKM	$\sum_{c\in\Pi_i}(\lambda_c)^\beta$	$\frac{\sum_{x_i\in\pi_c}\frac{\sum_{c\in\Pi_i}(\lambda_c)^\beta}{\lvert\Pi_i\rvert^2}.\tilde{x_i}^c}{\sum_{x_i\in\pi_c}\frac{1}{\lvert\Pi_i\rvert^2}}$	$\frac{1}{\sum_{l=1}^{k}\left(\frac{\sum_{x_i\in\pi_c}\lVert x_i - im_{\Pi,C}(x_i)\rVert^2}{\sum_{x_j\in\pi_l}\lVert x_j - im_{\Pi,C}(x_j)\rVert^2}\right)^{\frac{1}{\beta-1}}}$

then assigning observations in the order defined while assignment improves the local error therefore, reducing the objective function after each assignment step. A pseudo code for R_ASSIGN is described in Algorithm 6.

The proposed generic model for overlapping clustering is instantiated with three different methods which are R-OKM, Parameterized R-OKM and Adjusted R-OKM. Each instantiation of the generic model takes into account some properties relative to the size of overlaps between clusters and leads to different layouts for the overlapping boundaries between clusters. In practice, users should decide whether instantiation must be used depending on the requirement of the application and depending on the expected size of overlaps. For example, if small overlaps are expected, users can use the R-OKM method; However, to test different alternatives with different sizes of overlaps the Parameterized R-OKM can be used by adjusting the parameter α. This parameter can be bounded by the interval $[\![-1, 10]\!]$ where -1 coincides with the situation of max overlaps where any data is assigned is to any cluster and 10 coincides with the situation of null overlaps. Users can

Algorithm 5 *Regulated overlapping k-means* $(X, t_{max}, \varepsilon, k) \rightarrow \Pi$

```
INPUT  $X$: a dataset described over $\mathbb{R}^d$.
       $t_{max}$: maximum number of iterations.
       $\varepsilon$: minimal improvement in the objective function.
       $k$: number of clusters.
OUTPUT $\Pi$: assignment of observations over $k$ clusters.
 1: Initialize representatives of clusters $C^0$ randomly over $X$, initialize weights $\Omega^0$, initialize clusters memberships $\Pi^0$ using $R\_ASSIGN(x_i, C^0)$ and compute the objective function $J(\Pi^0, C^0, \Omega^0)$ at iteration 0.
 2: $t = t + 1$.
 3: Update clusters representatives $C^t$ (using Table 3.1 column 3).
 4: Compute new assignments $\Pi^t$ using $R\_ASSIGN(x_i, C^t, \Pi_i^{t-1})$ $\forall i$.
 5: Update weights $\Omega^t =$ (using Table 3.1 column 4).
 6: Compute objective function $J(\Pi^t, C^t, \Omega^t)$.
 7: if ($t < t_{max}$ and $J(\Pi^{t-1}, C^{t-1}, \Omega^{t-1}) - J(\Pi^t, C^t, \Omega^t) > \varepsilon$) then
 8:    Go to step 2.
 9: else
10:    Return $\Pi^t$ the final cluster memberships matrix.
11: end if
```

Algorithm 6 $R\_ASSIGN(x_i, \{c = 1, ...c = k\}, \Pi_i^{old}) \rightarrow \Pi_i$

```
INPUT  $x_i$: Vector in $\mathbb{R}^d$.
       $\{c = 1, ...c = k\}$: set of $k$ clusters
       $\Pi_i^{old}$: Old assignment for observation $x_i$.
OUTPUT $\Pi_i$: New assignment for $x_i$.
 1: Initialize $\Pi_i = \{c^\star\}$ the nearest cluster where $c^\star = \arg\min_{m_c} \parallel x_i - m_c \parallel^2$.
 2: Looking for the next nearest cluster $c^\star$ which is not included in $\Pi_i$.
 3: Compute $im_{\Pi',C}(x_i)$ and $\omega'_i$ with assignments $\Pi'_i = \Pi_i \cup \{c^\star\}$.
 4: if $\omega'_i. \parallel x_i - im_{\Pi',C}(x_i) \parallel^2 < \omega_i. \parallel x_i - im_{\Pi,C}(x_i) \parallel^2$ then
 5:    $\Pi_i \leftarrow \Pi'_i$ and go to step 2.
 6:    if $\omega_i. \parallel x_i - im_{\Pi,C}(x_i) \parallel^2 \leq \omega_i^{old}. \parallel x_i - im_{\Pi^{old},C}(x_i) \parallel^2$ then
 7:       Return $\Pi_i$.
 8:    else
 9:       Return $\Pi_i^{old}$.
10:    end if
11: end if
```

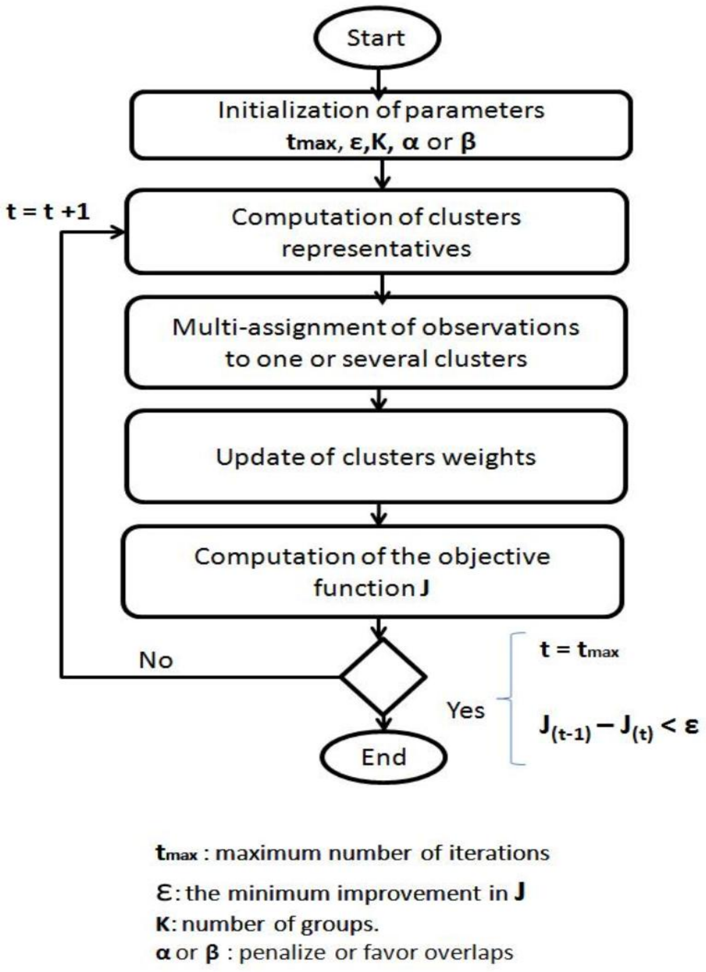

Start
Initialization of parameters
tmax, ε,K, α or β
t = t +1
Computation of clusters representatives
Multi-assignment of observations to one or several clusters
Update of clusters weights
Computation of the objective function J
No
Yes
t = tmax
J(t-1) – J(t) < ε
End
tmax : maximum number of iterations
ε: the minimum improvement in J
K: number of groups.
α or β : penalize or favor overlaps

Figure 3.3: Main steps of the generic model for overlapping clustering with overlap regulation

detect by dichotomy on this interval the smallest value α_{min} leading to a non-overlapping clustering. Then, they can test several values uniformly distributed over $[-1, \alpha_{min}]$ and keep the salient values corresponding to strong jumps in the overlap rates obtained from two consecutive values of α. On the other side, if users have no prior knowledge about the size of overlaps of the data to be clustered, the Adjusted R-OKM should be used to obtain an automatic customization of the overlaps.

All the proposed instantiations require to configure the number of clusters in prior. However, in practice this number is unknown while data to be clustered are unlabeled.

As a solution, users can test different clusterings with increasingly number of clusters and then, take the clustering having the best balance between the minimization of the objective function and the number of clusters.

In the next section, we look to empirically confirm the performance of the proposed methods, and therefore the performance of the proposed generic model.

3.5 Experiments and discussions

This section describes the details of the experiments that we performed in order to show the ability of the proposed methods to produce clusters with relevant overlaps.

3.5.1 Evaluation methodology

We propose a two fold assessment: firstly, as an "internal" point of view we give visual and quantitative information about the overlaps produced on a small example and on multi-labeled benchmarks; then we perform a standard "external" evaluation by comparing clusterings obtained with the benchmark basis.

We conducted experiments on different domains that motivate the overlapping clustering researches. In addition to the datasets described in Section 2.5.4, we add experiments on the yeast dataset which contains a numeric descriptions of genes where each gene can participate on different metabolic processes. Table 3.2 summarizes the statistics of each dataset. "Overlap" rate is the average number of labels per data. This rate will be compared with the overlap rate obtained with the clustering method. All described datasets are multi-labeled except the "Iris" dataset which is used only as a comparative example. These datasets are characterized by their diversity of application domains, their diversity of sizes (75 $\rightarrow$ 2417), their diversity of dimensions (3 $\rightarrow$ 192) and their diversity of overlap rates (1 $\rightarrow$ 4.23).

Table 3.2: Statistics of used benchmarks

Dataset	Domain	Observation	Dimension	Labels	Overlap
Iris	Plants	150	4	3	1
EachMovie	Video	75	3	3	1.14
Music emotion	Music	593	72	6	1.86
Scene	Images	2407	294	6	1.07
Yeast	Biology	2417	103	14	4.23

3.5.2 Evaluation of the size of overlaps

To study the overlaps obtained by the new model, particulary the Parameterized R-OKM method, we build Voronoï cells on the three-clusters example previously used. Figure 3.4 shows the evolution in the overlaps when the parameter α varies from $\alpha = 0$ (matching with OKM) to $\alpha \to +\infty$ (matching with k-means) via $\alpha = 1$ (matching with R-OKM). This figure shows that Parameterized R-OKM considerably reduces the overlapping area between clusters 1,2 and 3 compared to the ones obtained with OKM, and then, its ability to control the sizes of the overlaps.

On a quantitative point of view, Table 3.3 reports the overlap rates reported by the proposed methods compared with OKM and fuzzy-c-means. We produce overlapping clustering with fuzzy-c-means using a threshold post-assignment stage: any data with a membership degree higher than a threshold θ are assigned to the considered cluster. Comparison are performed with different initializations. The reported rates are averages and standard deviations obtained over ten runs. For each run, all the methods are started with the same initialization of cluster representatives to guarantee that all methods have the same experimental conditions.

Table 3.3 shows that the proposed methods generate reasonable overlap sizes with respect to the expected overlaps in each dataset. However, OKM and fuzzy-c-means generate large overlaps as well as the actual overlap size in each dataset increases. For example, using fuzzy c-means, observations in the Yeast dataset are assigned to almost clusters and the overlap rate is near to 14 (12.87). In fact, the size of overlaps obtained with proposed methods is lower than the size of overlaps obtained with OKM over all datasets. These results show the ability of the proposed generic model to produce clusters

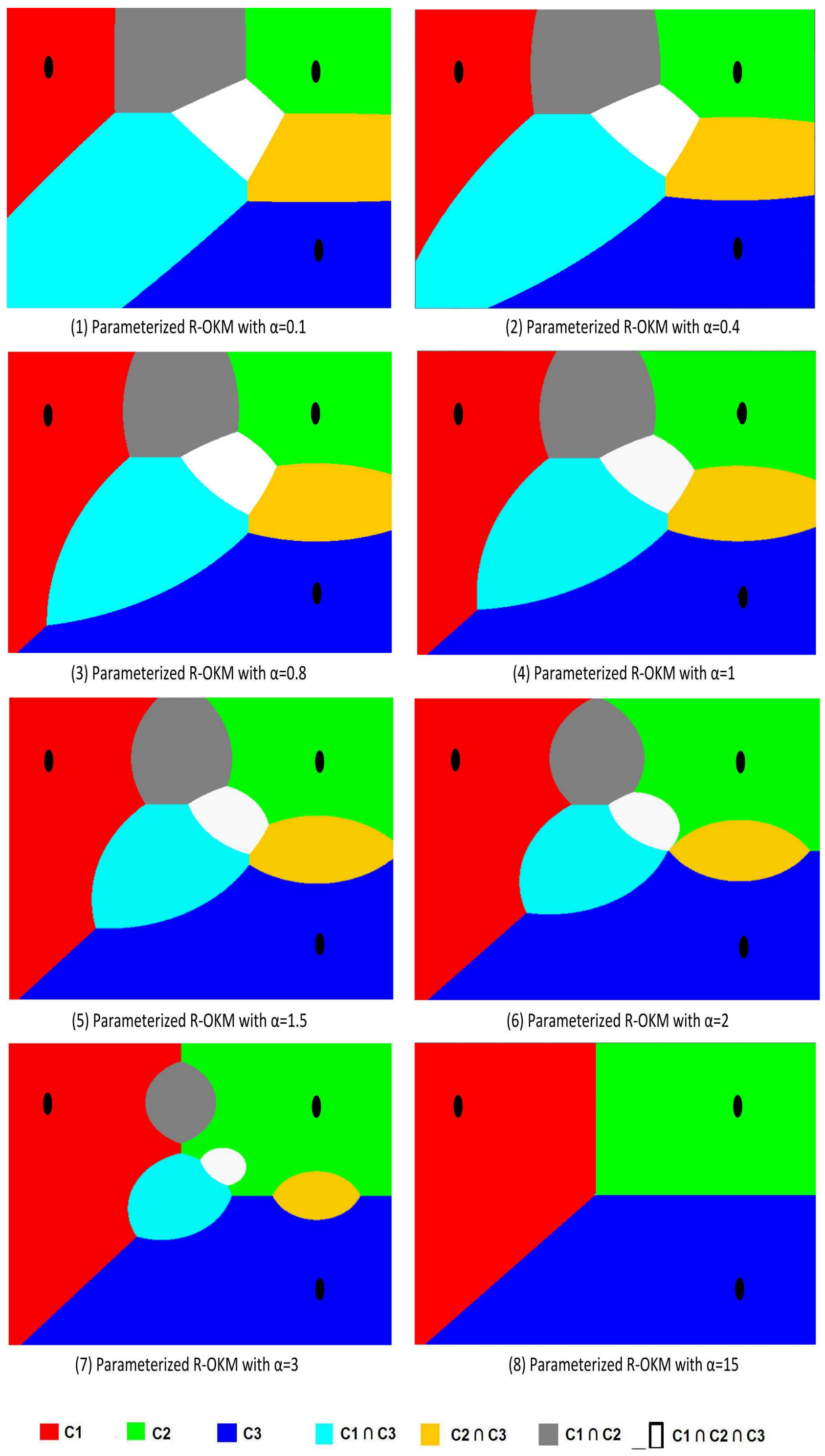

(1) Parameterized R-OKM with α=0.1
(2) Parameterized R-OKM with α=0.4
(3) Parameterized R-OKM with α=0.8
(4) Parameterized R-OKM with α=1
(5) Parameterized R-OKM with α=1.5
(6) Parameterized R-OKM with α=2
(7) Parameterized R-OKM with α=3
(8) Parameterized R-OKM with α=15
C1
C2
C3
C1 ∩ C3
C2 ∩ C3
C1 ∩ C2
C1 ∩ C2 ∩ C3

Figure 3.4: Voronoï cells obtained with Parameterized R-OKM method for three clusters with different values of parameter α: overlapping cells become more important when α approach to 0 (coincides with OKM) and become null when α is larger (coincides with k-means).

Table 3.3: Size of the overlaps on benchmark datasets using the proposed methods and compared with OKM, ALS and fuzzy-c-means.

Dataset	**Iris**	**Eachmovie**	**Music Emotion**	**Scene**	**Yeast**
Overlap rates	**(1)**	**(1.14)**	**(1.81)**	**(1.07)**	**(4.23)**
fuzzy-c-means ($\theta = \frac{1}{k}$)	1.14 ± 0.01	1.24 ± 0.00	1.43 ± 0.01	5.22 ± 0.22	12.87 ± 0.16
OKM	1.34 ± 0.12	1.76 ± 0.22	2.35 ± 0.20	2.33 ± 0.08	**4.75** ± 0.23
ALS	1.57 ± 0.10	1.76 ± 0.11	3.45 ± 0.10	-	-
Parameterized R-OKM ($\alpha = 0.01$)	1.32 ± 0.06	1.70 ± 0.07	2.11 ± 0.05	2.34 ± 0.11	**4.48** ± 0.22
Parameterized R-OKM ($\alpha = 0.1$)	1.31 ± 0.04	1.68 ± 0.07	**1.84** ± 0.05	1.66 ± 0.09	2.75 ± 0.10
Parameterized R-OKM($\alpha = 0.4$)	1.25 ± 0.02	1.49 ± 0.03	1.69 ± 0.02	**1.03** ± 0.02	1.03 ± 0.04
Parameterized R-OKM ($\alpha = 0.6$)	1.24 ± 0.02	1.29 ± 0.02	1.57 ± 0.04	1.01 ± 0.01	1.00 ± 0.00
Parameterized R-OKM ($\alpha = 0.8$)	1.21 ± 0.03	1.28 ± 0.10	1.39 ± 0.08	1.01 ± 0.01	1.00 ± 0.00
Restricted-OKM	1.16 ± 0.03	1.21 ± 0.11	1.31 ± 0.04	**1.03** ± 0.04	1.00 ± 0.00
Parameterized R-OKM ($\alpha = 1.2$)	1.13 ± 0.03	**1.14** ± 0.07	1.22 ± 0.05	1.00 ± 0.00	1.00 ± 0.00
Parameterized R-OKM($\alpha = 1.5$)	1.10 ± 0.02	**1.11** ± 0.08	1.20 ± 0.02	1.00 ± 0.00	1.00 ± 0.00
Parameterized R-OKM ($\alpha = 2$)	1.05 ± 0.01	1.05 ± 0.08	1.14 ± 0.04	1.00 ± 0.00	1.00 ± 0.00
Parameterized R-OKM ($\alpha = 3$)	**1.01** ± 0.00	1.03 ± 0.03	1.05 ± 0.03	1.00 ± 0.00	1.00 ± 0.00
Parameterized R-OKM ($\alpha = 5$)	**1.00** ± 0.00	1.00 ± 0.00	1.00 ± 0.01	1.00 ± 0.00	1.00 ± 0.00
Adjusted R-OKM ($\beta = 0.8$)	1.13 ± 0.07	1.18 ± 0.05	1.29 ± 0.09	1.00 ± 0.00	1.00 ± 0.00
Adjusted R-OKM ($\beta = 0.6$)	1.18 ± 0.01	**1.16** ± 0.08	1.30 ± 0.10	1.00 ± 0.00	1.00 ± 0.00
Adjusted R-OKM ($\beta = 0.4$)	1.17 ± 0.02	**1.18** ± 0.10	1.30 ± 0.08	1.00 ± 0.00	1.00 ± 0.00
Adjusted R-OKM ($\beta = 0.1$)	1.16 ± 0.01	1.21 ± 0.12	1.27 ± 0.07	1.01 ± 0.01	1.00 ± 0.00
Adjusted R-OKM ($\beta = 0.05$)	1.16 ± 0.03	1.21 ± 0.10	1.27 ± 0.03	1.01 ± 0.01	1.00 ± 0.00
Adjusted R-OKM ($\beta = 0.01$)	1.16 ± 0.03	1.20 ± 0.10	1.28 ± 0.03	1.01 ± 0.01	1.00 ± 0.00

with adjusted overlaps.

3.5.3 Empirical results on real multi-labeled datasets

To evaluate the clustering results, external validation measures were calculated based on the BCubed technique. We compare the new generic model using different instantiations with OKM, ALS, fuzzy-c-means and k-means. Table 3.4 reports the average scores and the standard deviation of Precision, Recall and F-measure on ten runs. For each run, all the methods are started with the same initialization (cluster representatives). Values in bold correspond to the best obtained scores.

Results obtained with fuzzy-c-means using different thresholding membership are characterized by low values and are match sensitive to the used threshold: for example, in the Yeast dataset, using a threshold equal to 0.0714 the obtained F-measure is equal to 0.257 while using the same method with a threshold equal to 0.0715 the F-measure decreases to 0.017 because clusters memberships are almost null. These results show the limit of fuzzy c-means to detect overlapping groups.

Table 3.4: Comparison of proposed methods with OKM, ALS, fuzzy c-means and k-means on the benchmark datasets

	Method	Precision	Recall	F-measure
Eachmovie dataset	k-means	0.644 ± 0.100	0.587 ± 0.036	0.610 ± 0.070
	fuzzy-c-means ($\theta = \frac{1}{k}$)	0.610 ± 0.001	0.734 ± 0.001	**0.666** ± 0.001
	OKM	0.402 ± 0.020	**0.912** ± 0.055	0.540 ± 0.030
	ALS	0.366 ± 0.032	0.819 ± 0.050	0.506 ± 0.038
	Restricted-OKM	0.532 ± 0.010	0.764 ± 0.003	0.627 ± 0.006
	Parameterized R-OKM($\alpha = 0.1$)	0.474 ± 0.016	**0.901** ± 0.043	0.621 ± 0.024
	Parameterized R-OKM($\alpha = 0.4$)	0.498 ± 0.029	0.846 ± 0.062	0.627 ± 0.023
	Parameterized R-OKM($\alpha = 0.9$)	0.535 ± 0.013	0.763 ± 0.020	0.629 ± 0.016
	Parameterized R-OKM($\alpha = 1.5$)	0.591 ± 0.071	0.700 ± 0.008	0.639 ± 0.044
	Parameterized R-OKM($\alpha = 3$)	0.627 ± 0.136	0.601 ± 0.063	0.605 ± 0.083
	Adjusted R-OKM ($\beta = 0.4$)	0.534 ± 0.039	0.639 ± 0.134	0.580 ± 0.080
	Adjusted R-OKM ($\beta = 0.1$)	0.573 ± 0.058	0.735 ± 0.038	0.642 ± 0.034
	Adjusted R-OKM ($\beta = 0.05$)	0.568 ± 0.028	0.757 ± 0.015	0.649 ± 0.014
	Adjusted R-OKM ($\beta = 0.01$)	0.564 ± 0.030	0.745 ± 0.027	0.641 ± 0.028
Emotion dataset	k-means	0.507 ± 0.003	0.206 ± 0.012	0.293 ± 0.010
	fuzzy-c-means ($\theta = \frac{1}{k}$)	0.493 ± 0.003	0.357 ± 0.001	0.414 ± 0.002
	OKM	0.483 ± 0.001	0.646 ± 0.031	**0.552** ± 0.012
	ALS	0.307 ± 0.015	**0.970** ± 0.021	0.466 ± 0.018
	Restricted-OKM	**0.505** ± 0.004	0.335 ± 0.058	0.401 ± 0.040
	Parameterized R-OKM($\alpha = 0.1$)	0.480 ± 0.002	0.581 ± 0.087	**0.524** ± 0.036
	Parameterized R-OKM($\alpha = 0.4$)	0.484 ± 0.012	0.525 ± 0.079	0.503 ± 0.041
	Parameterized R-OKM($\alpha = 0.9$)	0.491 ± 0.012	0.378 ± 0.050	0.426 ± 0.031
	Parameterized R-OKM($\alpha = 1.5$)	0.500 ± 0.002	0.286 ± 0.019	0.363 ± 0.015
	Parameterized R-OKM($\alpha = 3$)	0.501 ± 0.007	0.218 ± 0.013	0.304 ± 0.010
	Adjusted R-OKM ($\beta = 0.4$)	0.484 ± 0.004	0.280 ± 0.071	0.352 ± 0.054
	Adjusted R-OKM ($\beta = 0.1$)	0.483 ± 0.016	0.558 ± 0.044	0.479 ± 0.030
	Adjusted R-OKM ($\beta = 0.05$)	0.495 ± 0.022	0.329 ± 0.050	0.393 ± 0.028
	Adjusted R-OKM ($\beta = 0.01$)	**0.503** ± 0.006	0.322 ± 0.019	0.392 ± 0.016
Scene dataset	k-means	0.441 ± 0.010	0.406 ± 0.012	0.429 ± 0.010
	fuzzy-c-means ($\theta = \frac{1}{k}$)	0.324 ± 0.004	0.482 ± 0.022	0.388 ± 0.005
	OKM	0.233 ± 0.006	**0.928** ± 0.013	0.372 ± 0.008
	ALS	-	-	-
	Restricted-OKM	0.448 ± 0.004	0.413 ± 0.058	0.430 ± 0.040
	Parameterized R-OKM($\alpha = 0.1$)	0.290 ± 0.001	0.765 ± 0.027	0.421 ± 0.006
	Parameterized R-OKM($\alpha = 0.4$)	0.444 ± 0.013	0.426 ± 0.014	**0.435** ± 0.013
	Parameterized R-OKM($\alpha = 0.9$)	**0.448** ± 0.005	0.413 ± 0.008	**0.430** ± 0.007
	Parameterized R-OKM($\alpha = 1.5$)	**0.448** ± 0.007	0.413 ± 0.009	**0.430** ± 0.007
	Parameterized R-OKM($\alpha = 3$)	**0.448** ± 0.007	0.413 ± 0.009	**0.430** ± 0.007
	Adjusted R-OKM ($\beta = 0.4$)	0.286 ± 0.093	0.582 ± 0.002	0.344 ± 0.023
	Adjusted R-OKM ($\beta = 0.1$)	0.416 ± 0.039	0.409 ± 0.026	0.412 ± 0.023
	Adjusted R-OKM ($\beta = 0.05$)	**0.454** ± 0.014	0.415 ± 0.012	**0.434** ± 0.012
	Adjusted R-OKM ($\beta = 0.01$)	**0.454** ± 0.010	0.415 ± 0.009	**0.433** ± 0.010
Yeast dataset	k-means	**0.801** ± 0.006	0.075 ± 0.001	0.137 ± 0.002
	fuzzy-c-means ($\theta = \frac{1}{k}$)	0.148 ± 0.000	**1.000** ± 0.000	0.257 ± 0.000
	OKM	0.783 ± 0.001	**0.855** ± 0.051	**0.817** ± 0.024
	ALS	-	-	-
	Restricted-OKM	**0.801** ± 0.006	0.107 ± 0.067	0.137 ± 0.002
	Parameterized R-OKM($\alpha = 0.01$)	0.783 ± 0.002	**0.830** ± 0.050	**0.806** ± 0.024
	Parameterized R-OKM($\alpha = 0.05$)	0.783 ± 0.001	0.710 ± 0.055	0.744 ± 0.031
	Parameterized R-OKM($\alpha = 0.1$)	0.786 ± 0.003	0.493 ± 0.033	0.606 ± 0.027
	Parameterized R-OKM($\alpha = 0.8$)	**0.801** ± 0.005	0.075 ± 0.001	0.137 ± 0.002
	Parameterized R-OKM($\alpha = 1.5$)	**0.801** ± 0.006	0.107 ± 0.067	0.137 ± 0.002
	Parameterized R-OKM($\alpha = 3$)	**0.801** ± 0.006	0.107 ± 0.067	0.137 ± 0.002
	Adjusted R-OKM ($\beta = 0.8$)	0.782 ± 0.002	0.736 ± 0.389	0.735 ± 0.229
	Adjusted R-OKM ($\beta = 0.4$)	0.790 ± 0.011	0.163 ± 0.023	0.270 ± 0.016
	Adjusted R-OKM ($\beta = 0.1$)	**0.804** ± 0.003	0.072 ± 0.001	0.134 ± 0.002

For all experiments, the obtained size of overlaps influences the value of obtained F-measure: as well as the size of overlaps increases, the value of Precision decreases and the value of Recall increases. We notice that OKM has the best values of Recall because OKM builds clusters with large overlapping boundaries and k-means has the best values of Precision because overlaps are null. But, comparing to the overall F-measure, results obtained by proposed methods outperform results obtained by k-means and OKM. This improvement is explained by the reasonable sizes of overlaps compared to the actual overlaps in each dataset.

For the first proposed instantiation R-OKM, we notice the performance of this method to detect clusters with *small overlaps* as shown in Eachmovie and Scene datasets where actual overlaps are less than 1.5. The improvement in overall F-measure is considerably important: for example, in Scene dataset the F-measure obtained is 0.430, while using OKM this measure is 0.372. This improvement is principally induced by the improvement of the Precision. In fact, the average of Precision obtained by R-OKM is characterized by a high value compared to OKM for all datasets. The improvement of Precision is explained by the restriction in the assignments. But in other hand, we notice the limit of R-OKM to detect clusters with *large overlaps* as shown in the Yeast dataset (actual overlaps in Yeast is 4.23) where R-OKM fails to detect these clusters (the obtained F-measure is equal to 0.137).

For the the second proposed instantiation (Parameterized R-OKM), we notice the performance of this method to detect both *small* and *large* overlaps depending on the parameter α: the best obtained F-measure using Parameterized R-OKM outperforms both OKM and R-OKM on *small* overlapping datasets as shown in Scene and Eachmovie, and on *large* overlapping datasets as shown in Music Emotion and Yeast. In fact, the high scores obtained with Parameterized R-OKM matches with reasonable sizes of overlaps compared to the actual overlaps in each dataset as described in Figure 3.5. This figure shows also that OKM usually builds large overlaps in all datasets which confirms the limit of this method to build clusters with a reasonable overlapping boundaries.

The reported results, in Table 3.4, of Parameterized R-OKM with different values of

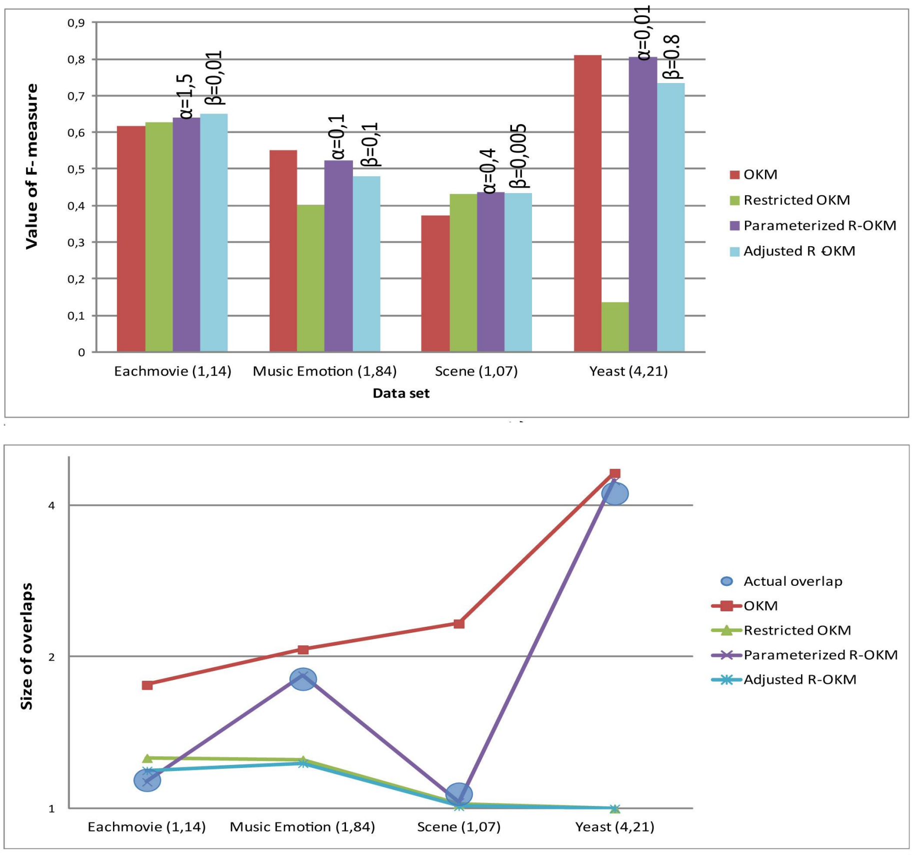


Figure 3.5: Matching between F-measure and Overlap sizes obtained with proposed methods compared to OKM on the benchmark datasets

α show the impact of this parameter used within Parameterized R-OKM: the obtained size of overlaps in all datasets is reduced when α becomes more important. For high values of α, Parameterized R-OKM converges to k-means and builds hard clusters where observations are assigned to only one group. For small values of α, especially those closer to 0, the method converges to OKM and builds large overlaps as shown in Yeast dataset where the size of overlaps is equal to 4.48 when $\alpha = 0.01$.

For the third proposed instantiation Adjusted R-OKM, we notice the ability of this method to auto adjust overlaps based on the quality of obtained clusters as shown in Eachmovie and Scene datasets. The best obtained results outperform results obtained by both OKM and R-OKM methods especially for *small* overlapping datasets. The β parameter, used within Adjusted R-OKM, determines the degree of influence of the considered weight λ_c local to each cluster π_c when building overlapping clusters. In fact, when β approaches to 0, the weight λ_c assigned to each cluster converges to $1/k$ and the method has the characteristics of R-OKM because all clusters have nearly the same cluster weights. In contrast, when β approaches to 1, the weight λ_c assigned to cluster c becomes more important than others weights if observations in this cluster are close to their images. As shown in Eachmovie dataset, when β approaches to 1 the value of precision and the size of overlaps decrease while the value of recall increases.

As a summary of the experiments, histograms in Figure 3.5 show that the new R-OKM method outperforms OKM in the majority of datasets. In addition, the parameterized and the adjusted variants can lead to better results again in almost datasets.

3.6 Conclusion

We proposed in this chapter a generic model for restricted overlapping clustering to produce clusters with control of overlapping boundaries. The proposed generic model can be instantiated in different ways to take into account the ability of the method to regulate sizes of overlaps and the ability to auto adjust overlaps based on the internal variability of data in each obtained cluster. The model can be seen as a new generalization of k-means for overlapping clustering by introducing a new objective function. Empirical results prove the efficiency of the proposed model compared to OKM and k-means methods to produce non large overlapping clusters with a high classification precision.

In real life applications, there are many cases where the input data should not be described by explicit feature vectors but described by strings or trees such as the case of text documents. For such types of data, it would be interesting to investigate the application of an overlapping clustering process.

Chapter 4

Overlapping clustering of textual documents

Contents

4.1 Introduction

Text clustering is a widely used technique in Information Retrieval (IR) to find analogous documents (Saracoglu et al. 2007), to organize large document collection (Aliguliyev 2009; Isa et al. 2009; Liu et al. 2011), to detect duplicate contents and to optimize search engines (Oberreuter and Velsquez 2013). This technique aims to group similar documents in the same group or cluster based on their contents, while dissimilar documents must belong to different groups without using any predefined categories. This definition can be a crucial issue in many real life applications of text clustering where a document needs to be assigned to more than one group (Saracoglu et al. 2008). Another issue in Text clustering is the representation of text within the Vector Space Model (VSM). This text representation is based on the assumption that relative position of tokens are irrelevant leading to the loss of correlation with adjacent words and to the loss of information regarding words positions. This fact affects the quality of obtained clusters.

Therefore, we present in this chapter a clustering process (BenN'Cir et al. 2013) where the correlation between adjacent words in textual documents and the possibility of documents to belong to more than one cluster are not ignored.

This chapter is organized as follows: Section 4.2 presents the motivation of considering overlapping partitioning from textual documents, then Section 4.3 recalls necessary background of document clustering. After that, Section 4.4 describes the proposed solution, KOKM based WSK, which we propose to make overlapping clustering from sequential textual documents while Section 4.5 presents the experiments that we performed to evaluate the effectiveness of the proposed method. Finally, Section 4.6 presents the conclusion.

4.2 Motivation: overlapping clustering of textual documents

Document clustering represents a natural problem where documents need to be assigned to more than one cluster since each document can discuss several topics. For example, a newspaper article concerning the participation of a soccer player in the release of an action film can be grouped with both of the categories Sports and Movies. Although recent methods for overlapping clustering propose theoretical (Banerjee et al. 2005; Heller and Ghahramani 2007; Cleuziou 2008) models to solve this issue, they still understudied in document clustering. The application of these methods to text documents needs to prepare the collection of documents for numerical analysis by using VSM representation. This text representation is based on the assumption that relative position of tokens are irrelevant leading to the loss of correlation with adjacent words and to the loss of information regarding words positions. This fact affects the quality of obtained clusters.

In order to deal simultaneously with the issue of loosing information regarding words positions and the issue of identifying relevant overlapping clusters from a set of documents, we show in the next sections, how we introduce the Word Sequence Kernel (WSK) as similarity measure to detect overlapping groups from sequential textual documents.

4.3 Text document clustering

Document Clustering, which is the process of finding natural groupings in documents, is an important task in information retrieval. It is based on the assumption that documents which discuss the same topic, such as medical, financial, sport and legal topics, tend to be more similar to each other and therefore tend to appear in the same cluster (Jardine and van Rijsbergen 1971).

In fact, building an automated document clustering system consists of two important steps. The first step is text representation which converts the unstructured content of documents into numerical format since unsupervised learning methods receive numerical vectors as their input data. The second step is to learn model of a text classifier which is used for finding groups in unlabeled documents.

4.3.1 Text pre-processing

The process of encoding documents into numerical format begins by pre-processing text in order to extract words as feature candidates from a particular corpus. Figure 4.1

illustrates the process of extracting feature candidates from the corpus where all texts in the corpus are concatenated into a long string.

In the first step, *tokenization*, the long string is segmented into tokens by a white space or a punctuation mark. In the second step, each token is *stemmed* into its root form; verbs in their past form are stemmed into their root form and nouns in their plural form are stemmed into their singular form. For example, the words "computer","computation"and "computer" are all reduced to the stem "compute". After that, in the third step, stop words are removed for processing documents more efficiently. Stop words are words which perform only grammatical functions and are irrelevant to contents such as conjunctions, articles and prepositions. Through the three steps illustrated in Figure 4.1, a list of words and their frequencies are generated as a group of feature candidates.

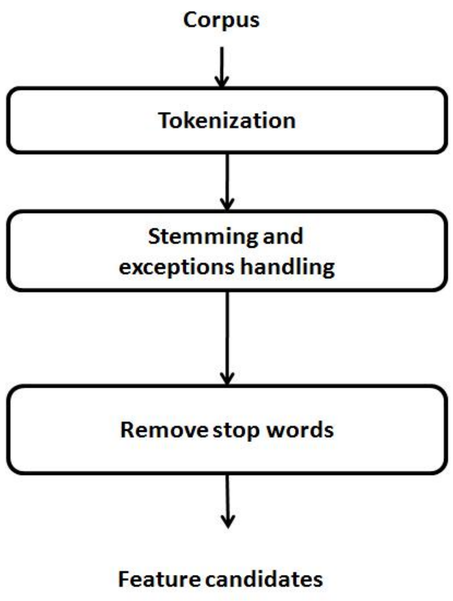


Figure 4.1: The process of extracting feature candidates from a corpus

4.3.2 Document representation: Vector Space Model (VSM)

The VSM representation is usually used in Text Clustering to prepare textual documents for a numerical analysis process. In VSM model, each text document d_j is represented by a vector of features (words or terms) $\overrightarrow{d_j} = (w_{1j}, w_{2j}, ..., w_{|T|_j})$, where T is the whole set

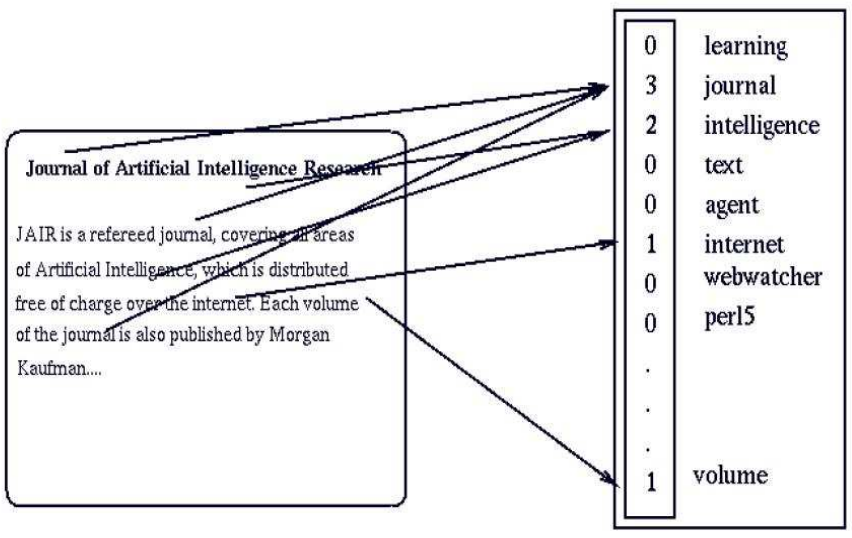

Journal of Artificial Intelligence Research
JAIR is a refereed journal, covering all areas of Artificial Intelligence, which is distributed free of charge over the internet. Each volume of the journal is also published by Morgan Kaufman....
0 learning
3 journal
2 intelligence
0 text
0 agent
1 internet
0 webwatcher
0 perl5
1 volume

Figure 4.2: Document representation under the Vector Space Model (VSM)(Steinbach et al. 2000)

of terms $T = (t_1, ..., t_{|T|})$, $|T|$ is the size of the vocabulary and w_{kj} represents the weight of the term t_k in the document d_j. Documents whose vectors are close to each others are considered to be similar in content. This representation is based on the assumption that relative position of tokens has a little importance leading to the loss of correlation with adjacent words and to the loss of information regarding words positions.

4.3.3 Document representation: n-Grams

The "n-Grams" representation of text (Yannakoudakis et al. 1990), which is a language independent text representation technique, solves the issue of losing information regarding words positions by considering each text document as a sequence of n consecutive characters, syllables or words. The whole set of "n-Grams" is obtained by the extraction of all possible ordered subsequences of consecutive n characters along the text. Similarities between documents are measured based on the number of contiguous and non-contiguous subsequences shared between them. This representation leads to a high dimensional features of subsequences to represent each text document. This problem is solved in information retrieval tasks by using *Kernel Machines* over sequences of text.

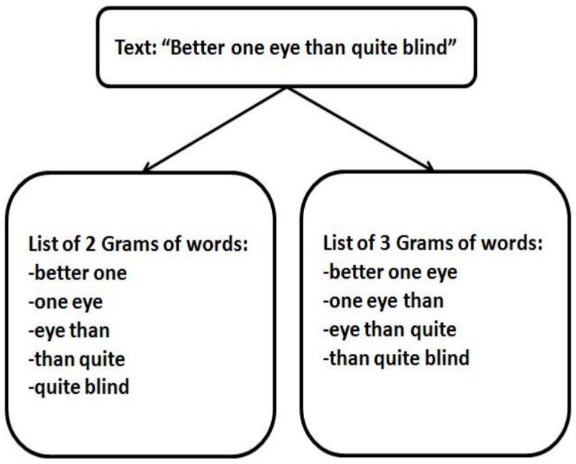


Figure 4.3: n-Grams representation of text: (a) 2-Grams of words (b) 3-Grams of words

4.4 Kernel based sequences similarity

Many kernels known as String Kernel are proposed in the literature to solve the problem of high dimensional features in n-Grams representation of Text. The n-Grams are not explicitly computed for each document, but only dot products between n-Grams of each pair of documents are computed. Examples of these kernels are String Subsequence Kernel (SSK) (Lodhi et al. 2001) and Word Sequence Kernel (WSK) (Cancedda et al. 2003). SSK measures similarity between two documents based on the number of sequences of characters shared between them while WSK measures similarity based on the number of sequences of *words* rather then *characters*. The advantage of using *words* as atomic unit is to keep information regarding words positions in order to maintain linguistic meaning of terms. For example, the terms "son-in-law" have a special meaning that can be lost if they are broken. The WSK has also the advantage of reducing the number of features per document because it uses sequences of words rather than sequences of characters. The time complexity of computing WSK similarity between two documents d_1 and d_2 is evaluated to $O(n.|d_1|.|d_2|)$ (Cancedda et al. 2003) where n is the length of the used subsequence and $|d_i|$ is the number of words in document d_i.

Let Σ the alphabet which consists in the set of words that exist in all documents. Let $S = t_1t_2t_3...t_{|S|}$ a sequence of words where $|S|$ is the length of S. Let $u = s[i]$

a subsequence in S with $s[i] = t_{i_1}..t_{i_j}..t_{i_n}$ where t_{i_1} and t_{i_j} in this subsequence are not necessarily contiguous in S and n is the length of the subsequence u. The feature mapping ϕ for the sentences S in the feature space is given by defining ϕ_u for each $u \in \Sigma^n$ as:

$$\phi_u(S) = \sum_{i:u=s[i]} \lambda^{l(i)}, \tag{4.1}$$

where λ is the decay factor used to penalize non contiguous subsequences and $l(i)$ is the length of subsequence $s[i]$ in S with $l(i) = Index(t_{i_n}) - Index(t_{i_1}) + 1$. These features measure the number of occurrences of subsequence u in the sentences S weighting them according to their lengths. So, given two strings S_1 and S_2, the inner product of the feature vectors is obtained by computing the sum over all common subsequences:

$$\begin{aligned} K_n(S_1, S_2) &= \sum_{u\in\Sigma^n} \phi_u(s_1)\phi_u(s_2) \\ &= \sum_{u\in\Sigma^n} \sum_{i:u=s_1[i]} \sum_{j:u=s_2[j]} \lambda^{l(i)+l(j)}. \end{aligned} \tag{4.2}$$

4.5 Proposed KOKM based WSK method

To detect non-disjoint groups from sequential text document, we propose "KOKM based WSK" using WSK as similarity measure between structured documents. Given a set of N documents $D = \{d_1, d_2, ..., d_N\}$ where each document d_q is defined in the feature space by the Sum of u coordinate $\Phi(d_q) = \sum_u \phi_u(d_q)$ which measures the number of occurrences of subsequence u in the document d_q weighted according to its lengths. The aim of the proposed method is to find the optimal assignments matrix $\Pi(N \times k) = \{\pi_1, \pi_2, ..., \pi_k\}$ of documents over k non-disjoint groups. The proposed method consists in minimizing an objective function defined by the sum of errors E_q local to each document d_q. The sum of local errors over all documents is described by:

$$E_r(\Pi) = \sum_{d_q\in D} E_q = \sum_{d_q\in D} \left\| \Phi(d_q) - \overline{\Phi(d_q)} \right\|^2, \tag{4.3}$$

where $\overline{\Phi(d_q)}$ is the combination of clusters representatives (typical documents) to which document d_q belongs and is defined by:

$$\overline{\Phi(d_q)} = \frac{\sum_{c=1}^{k} P_{qc} \cdot \Phi(d_{m_c})}{\sum_{c=1}^{k} P_{qc}}, \tag{4.4}$$

with P_{qc} is a binary variable indicating membership of document d_q in cluster c and d_{m_c} is the typical document of cluster c. Using the Kernel Trick and based on WSK kernel, the objective function is performed as follows:

$$\begin{aligned}
E_r(\Pi) &= \sum_{d_q \in D} \Big[\Phi(d_q)\Phi(d_q) - \frac{2}{L_q}\sum_{c=1}^{k} P_{qc} \cdot \Phi(d_q)\Phi(d_{m_c}) + \\
&\quad (\tfrac{1}{L_q})^2 \sum_{c=1}^{k}\sum_{l=1}^{k} P_{qc}P_{ql} \cdot \Phi(d_{m_c})\Phi(d_{m_l})\Big] \\
&= \sum_{d_q \in D} \Big[\sum_{u \in \Sigma^n} \phi_u(d_q)\phi_u(d_q) - \frac{2}{L_q}\sum_{c=1}^{k} P_{qc} \cdot \sum_{u \in \Sigma^n} \phi_u(d_q)\phi_u(d_{m_c}) + \\
&\quad (\tfrac{1}{L_q})^2 \sum_{c=1}^{k}\sum_{l=1}^{k} P_{qc}P_{ql} \cdot \sum_{u \in \Sigma^n} \phi_u(d_{m_c})\phi_u(d_{m_l})\Big] \\
&= \sum_{d_q \in D} \Big[K_n(d_q, d_q) - \frac{2}{L_q}\sum_{c=1}^{k} P_{qc} \cdot K_n(d_q, d_{m_c}) + \\
&\quad (\tfrac{1}{L_q})^2 \sum_{c=1}^{k}\sum_{l=1}^{k} P_{qc}P_{ql} \cdot K_n(d_{m_c}, d_{m_l})\Big],
\end{aligned} \tag{4.5}$$

where $L_q = \sum_{c=1}^{k} P_{qc}$ and $K_n(d_i, d_j)$ is the WSK similarity measure between document d_i and document d_j as described in Equation 4.2.

4.5.1 Clustering algorithm of KOKM based WSK

The minimization of the objective function is performed by iterating two independent steps:

1. Update of cluster representatives (d_{m_c}).

2. Multi assignment of documents to one or several clusters (Π).

The stopping rule of KOKM based WSK algorithm is characterized by two criteria: the maximum number of iterations or the minimum improvement of the objective function between two iterations. The main algorithm of KOKM based WSK is described by Algorithm 7.

```
Algorithm 7 KOKM based WSK ($(D, k, t_{max}, \varepsilon, n, \lambda) \rightarrow \Pi$)
INPUT $D$: set of Documents,
    $t_{max}$: maximum number of iterations,
    $\varepsilon$: minimal improvement in $E_r$ between two iterations,
    $k$: number of clusters,
    $n$: length of subsequences
    $\lambda$: decay factor used for WSK.
OUTPUT
 1: Initialize representatives of clusters over $D$, Assign documents using "MULTIASSIGN-
    DOC" and derive value of $E_r(\Pi_0)$ in iteration 0 using Equation 4.5.
 2: $t = t + 1$
 3: Update representatives using Equation 4.6.
 4: Assign documents to one or several clusters using "MULTIASSIGN-DOC" and derive
    $\Pi_t$.
 5: Compute $E_r(\Pi_t)$ using Equation 4.5.
 6: if ($t < t_{max}$ and $E_r(\Pi_{t-1})$ - $E_r(\Pi_t) > \varepsilon$) then
 7:     go to step 2.
 8: else
 9:     Return the assignment matrix $\Pi_t$.
10: end if
```

Considering the assignments (Π) fixed, the update of representatives is performed locally for each cluster. Each representative d_{m_c} is defined by the typical document (medoid) in cluster c which minimizes the sum of distances from all documents belonging to the following cluster weighted according to documents memberships as described in Equation 4.6.

$$
\begin{aligned}
d_{m_c} &= \min_{q \in \pi_c} \frac{\sum\limits_{j \in \pi_c, j \neq q} w_j . \|\Phi(d_q) - \Phi(d_j)\|^2}{\sum\limits_{j \in \pi_c, j \neq q} w_j} \\
&= \min_{q \in \pi_c} \frac{\sum\limits_{j \in \pi_c, j \neq q} w_j [K_n(d_q, d_q) - 2.K_n(d_q, d_j) + K_n(d_j, d_j)]}{\sum\limits_{j \in \pi_c, j \neq q} w_j}, \qquad (4.6)
\end{aligned}
$$

where w_j is a weight assigned to the distance between d_q and d_j depending on the number of clusters to which document d_j belongs. This weight is more important as the assignments of d_j increase in order to reduce its influence in determining the typical document (document which discuss more than one theme has small probability of being and determining the typical document).

The second step concerns the multi assignments of each document to one or several clusters. By considering representatives fixed, we present in the following a heuristic "MULTIASSIGN-DOC" which makes the objective function minimized and explores the combinatorial sets of possible assignments. The heuristic consists, for each document d_q, in sorting representatives of clusters from closest to farthest, then assigning the document in the order defined while assignment minimizes the local error E_q and therefore it minimizes the hole objective function. The heuristic "MULTIASSIGN-DOC" is described by Algorithm 8.

4.5.2 Diagonal dominance problem: Sub Polynomial Kernel as a solution

The proposed method KOKM based WSK uses the Kernel $K_n(d_i, d_j)$ as a similarity measure between documents d_i and d_j. $K_n(d_i, d_j)$ represents the inner product between features of documents d_i and d_j and is evaluated by the sum over all common subsequences weighted according to their frequency of occurrence, lengths and contiguities.

Nevertheless, this definition makes documents having extremely sparse representations

Algorithm 8 MULTIASSIGN-DOC$(d_q, \{d_{m_1}, ..., d_{m_k}\}, \Pi_q^{old}) \to \Pi_q$

```
INPUT $d_q$: Document considered as sequences of words,
    $\{d_{m_1}, ..., d_{m_k}\}$: $k$ cluster representatives ,
    $\Pi_q^{old}$: Old assignments of document $d_q$.
OUTPUT
 1: Initialize $\Pi_q = \{d^{\star}_{m_c}\}$ the nearest representative where $d^{\star}_{m_c} = \min_{d_{m_c}} \|\Phi(d_q) - \Phi(d_{m_c})\|^2$
    and compute $E_q$ with assignment $\Pi_q$.
 2: Find the next nearest representative $d^{\star}_{m_c}$ which is not included in $\Pi_q$ and derive $\Pi_q^{'} =$
    $\Pi_q \cup d^{\star}_{m_c}$
 3: Compute $E_q^{'}$ with assignment $\Pi_q^{'}$
 4: if $E_q^{'} < E_q$ then
 5:   $\Pi_q = \Pi_q^{'}$ and return to step 2.
 6: else
 7:   compute $E_q^{old}$ with assignment $\Pi_q^{old}$.
 8:   if $E_q < E_q^{old}$ then
 9:     return $\Pi_q$
10:   else
11:     return $\Pi_q^{old}$
12:   end if
13: end if
```

in the feature space and leads to the overfitting [1] situation when performing clustering algorithm. The kernel-based similarity $K_n(d_i, d_j)$ between any pair of distinct documents $(d_i \neq d_j)$ will tend to be very small $(K_n(d_i, d_j) \cong 0)$ with respect to the self-similarity of documents $(K_n(d_i, d_i) = 1)$, especially for large values of n. The Gram Matrix tends to be nearly diagonal because the off-diagonal entries are very small and the diagonal entries are equal to 1 meaning that all documents are mapped to nearly orthogonal points.

In order to overcome this problem, we propose to integrate Sub-Polynomial Kernels (Jason et al. 2003) within KOKM based WSK. Sub-Polynomial Kernels are used in Kernel machines to avoid the diagonal dominance in Gram Matrix. Given a positive kernel $K(x_i, x_j)$, the Sub-Polynomial Kernel is defined as:

$$K^{sp}(x_i, x_j) = (K(x_i, x_j))^p = \langle \phi(x_i), \phi(x_j) \rangle^p \tag{4.7}$$

where $p \in [0, 1]$ is the degree of the Sub-Polynomial Kernel. As well as the value of

[1]The Overfitting situation is reached when all documents are mapped to orthogonal points in feature space. For overlapping methods, the overfitting situation consists in assigning each document to all clusters.

p decreases ($p \longmapsto 0$), the ratio of diagonal entries to off-diagonal entries in the Gram Matrix decreases ($ratio \longmapsto 1$). Therefore, to deal with the diagonal dominance problem, we define a new Kernel based similarity $K_n^{sp}(d_i, d_j)$ between any pair of documents d_i and d_j defined by:

$$K_n^{sp}(d_i, d_j) = (K'_n(d_i, d_j))^p \tag{4.8}$$

where $K'_n(d_i, d_j)$ is the normalization of $K_n(d_i, d_j)$ to prevent influence of weighting according to the length of subsequences and is described by:

$$K'_n(d_i, d_j) = \frac{K_n(d_i, d_j)}{\sqrt{K_n(d_i, d_i) K_n(d_j, d_j)}}. \tag{4.9}$$

4.6 Experiments and discussions

4.6.1 Evaluation methodology

To evaluate quality of obtained groups of documents, we used an extension of external validation measures (Precision, Recall and F-measure) for multi labeled data based on Label Based Evaluation methodology (matching) (Tsoumakas et al. 2010) as described in Section 1.5. These validation measures attempt to estimate whether the prediction of categories is correct with respect to the underlying true categories in the data. The computation of external validation measure for all labels is archived using macro-averaging technique which is usually used in Information Retrieval tasks to evaluate clustering results when the number of classes is not large (Yang 1999).

Experiments are conducted on two overlapping textual datasets which are respectively Reuters [2] and Ohsumed [3] datasets. The first dataset contains a collection of documents initially composed of 21578 English newspaper articles. Each document is labeled by one or several labels from a set of 114 categories. We used different subsets [4] which are composed of 100, 500, 800 and 2000 documents, each document belongs to one or

[2]cf.http://kdd.ics.uci.edu/databases/reuters-transcribed/reuters-transcribed.html

[3]cf.http://disi.unitn.it/moschitti/corpora/ohsumed-first-20000-docs.tar.gz

[4]Subsets are built manually over 10 categories where documents which belong to more than one category were kept. The rates of overlaps in these subsets lie between 1.2 and 1.5.

several categories from a set of 10 categories. The second dataset is a set of different references from the On-Line Medical Information database (MEDLINE), consisting of titles and abstracts from 270 medical journals over five-year period. We extract a subset of Ohsumed composed of 200 documents where each document is labeled by one or several labels from a set of 5 categories.

4.6.2 Empirical results on real multi-labeled corpus

Experiments are performed on computer with 4 GB RAM and 2.1 GHZ Intel Core 2 duo processor. Data are preprocessed by removing a stop words. The VSM representation of each dataset is built using the “Weka text processing” module where frequency of occurrence of words is computed using the $TF-IDF$ technique. Table 4.1 and Table 4.2 report average scores and standard deviation variation of Precision, Recall and F-measure on ten runs using k-means, OKM and KOKMII methods based on VSM representation compared to the proposed method KOKM based WSK. For KOKMII method, we studied its performance using different types of kernels (linear, polynomial and RBF) while for KOKM based WSK, we used $n = 2$ and $n = 3$ the length of word sequences, $\lambda = 0.9$ the value of the decay factor and $p = 0.05$ the power of the sub polynomial kernel. For each run, all methods are computed with same initialization of seeds to guarantee that all methods have the same experimental conditions. Values in bold correspond to the best obtained scores.

As reported in Table 4.1 and Table 4.2, KOKM based WSK builds in almost all collections, higher quality clusterings than all the other methods, according to the F-measure. For summarizing the above results, we report the statistical significance matrix for the F-measure values obtained by each method as shown in Table 4.3. In this table, the symbols “>>” (“<<”) indicates that the F-measure values obtained by the method of the row are significantly better (worse) than the values obtained by the method of the column; the symbol “>” (“<”) indicates that the relation is not significant. For testing the statistical significance we used the MannWhitney U-test (Mann and Whitney 1947) with a 90% of confidence. This non parametric test is usually used to test whether one

Table 4.1: Comparison of the performance of KOKM based WSK versus other existing methods based on VSM representation in different collections of Reuters dataset.

Dataset	**Methods**	**Precision**	**Recall**	**F-measure**
Reuters-100	k-means	**0,570±0,02**	0,262±0,03	0,351±0,02
	Fuzzy-c-means	**0,562±0,01**	0,283±0,04	0,359±0,02
	OKM	0,275±0,01	**0,968±0,03**	0,429±0,01
	KOKMII (Linear)	0,275±0,01	0,955±0,04	0,427±0,02
	KOKMII (Polynomial)	0,275±0,01	0,955±0,04	0,427±0,01
	KOKMII (RBF σ=10^{10})	0,275±0,01	0,955±0,05	0,427±0,02
	KOKM based WSK (n=2)	**0,425±0,04**	0,717±0,12	**0,534±0,06**
	KOKM based WSK (n=3)	**0,436±0,06**	0,721±0,13	**0,540±0,09**
Reuters-500	k-means	0,427±0,03	0,132±0,10	0,201±0,07
	Fuzzy-c-means	0,432±0,04	0,142±0,10	0,223±0,07
	OKM	0,308±0,01	0,136±0,03	0,188±0,01
	KOKMII (Linear)	0,162±0,01	0,314±0,04	0,214±0,02
	KOKMII (Polynomial)	0,438±0,01	0,273±0,04	0,336±0,01
	KOKMII (RBF σ=10^{10})	0,388±0,01	0,141±0,05	0,207±0,02
	KOKM based WSK (n=2)	0,283±0,04	0,759±0,12	**0,410±0,04**
	KOKM based WSK (n=3)	0,200±0,04	0,466±0,12	0,280±0,04
Reuters-800	k-means	0,470±0,03	0,135±0,36	0,214±0,04
	Fuzzy-c-means	0,470±0,03	0,135±0,36	0,214±0,04
	OKM	0,122±0,01	0,583±0,03	0,202±0,01
	KOKMII (Linear)	0,170±0,01	**0,613±0,04**	0,267±0,02
	KOKMII (Polynomial)	0,122±0,01	0,583±0,04	0,202±0,01
	KOKMII (RBF σ=10^{10})	0,457±0,01	0,144±0,05	0,219±0,02
	KOKM based WSK (n=2)	0,334±0,04	**0,620±0,12**	**0,434±0,04**
	KOKM based WSK (n=3)	0,324±0,04	0,530±0,12	**0,402±0,04**
Reuters-2000	k-means	**0,520±0,04**	0,132±0,09	0,216±0,06
	Fuzzy-c-means	**0,520±0,04**	0,132±0,09	0,216±0,06
	OKM	0,183±0,01	0,316±0,03	0,232±0,01
	KOKMII (Linear)	0,128±0,01	0,391±0,04	0,193±0,02
	KOKMII (Polynomial)	0,150±0,01	0,321±0,04	0,205±0,01
	KOKMII (RBF σ=10^{10})	0,112±0,01	0,362±0,05	0,171±0,02
	KOKM based WSK (n=2)	0,345±0,04	**0,487±0,12**	**0,404±0,04**
	KOKM based WSK (n=3)	0,257±0,04	**0,568±0,12**	**0,354±0,04**

of two random variables is stochastically larger than the other (Fay and Proschan 2010).

As it can be seen from Table 4.3, KOKM based WSK significantly wins the other algorithms used in the comparison, in terms of the quality of the clusters. In fact, obtained F-measure with overlapping methods outperforms F-measure obtained with hard k-means. The improvement is induced by a remarkable improvement of Recall. Results obtained with hard k-means are characterized by high value of Precision and low value of Recall. Hard k-means fails to detect groups of document as well as the dimensionality increases which explains the low value of Recall when the number of documents increases as shown in Reuters-2000 (Recall obtained with k-means is 0.132).

Table 4.2: Comparison of the performance of KOKM based WSK versus other existing methods based on VSM representation in a collection of Ohsumed dataset.

Dataset	Methods	Precision	Recall	F-measure
Ohsumed-200	k-means	**0,450±0,03**	0,180±0,36	0,252±0,04
	Fuzzy-c-means	**0,455±0,02**	0,188±0,28	0,258±0,13
	OKM	0,274±0,03	0,799±0,36	0,396±0,04
	KOKMII (Linear)	0,297±0,10	0,798±0,36	0,417 ±0,11
	KOKMII (Polynomial)	0,297±0,10	0,798±0,35	0,417±0,10
	KOKMII (RBF σ=10^8)	0,262±0,04	0,835±0,40	0,385±0,03
	KOKM based WSK (n=2)	**0,324±0,03**	0,694±0,08	**0,441±0,02**
	KOKM based WSK (n=3)	**0,319±0,03**	0,709±0,08	**0,440±0,02**

The F-measure obtained with KOKM based WSK is characterized by a high value compared to overlapping methods based VSM representation. The improvement of F-measure is induced by the improvement of Precision. For example, in Reuters-100 dataset, the obtained Precision using KOKM based WSK is 0.425 while using KOKMII and OKM methods the max obtained Precision is 0.275. Recalls obtained with KOKMII and OKM methods are characterized by high values (Recall obtained with OKM in Reuters-100 dataset is 0.968). These high values of recall is explained by the way that OKM and KOKMII assign documents to all clusters because of the high dimensionality of data. For example, in Reuters-100 dataset, where the dimensionality of the VSM matrix is very sparse (1482 words), OKM and KOKMII assign each document to practically all clusters. This problem is solved when using KOKM based WSK which explains the important improvement in terms of Precision and the reduction of Recall (obtained Recall is 0.717).

Obtained results prove the theoretical finding that maintaining order in text improves clustering accuracy compared to VSM representation. The frequent word sequences can provide compact and valuable information about documents structures. In fact, methods based kernel function (Bag of Word Kernel or Word Sequence Kernel) outperform non

Table 4.3: Statistical significance matrix for F-measure values.

Method	k-means	Fuzzy-c-means	OKM	KOKMII (RBF)	KOKMII (Polynomial)	KOKM based WSK
k-means	-	<	<	<<	<	<<
Fuzzy-c-means	>	-	<	<	<<	<<
OKM	<	<	-	<	<	<<
KOKMII (RBF)	>	>	>	-	<	<<
KOKMII (Polynomial)	>>	>>	>>	>	-	<<
KOKM based WSK	>>	>>	>>	>>	>>	-

kernel methods. These results prove that looking for separation between clusters in a feature space is better than looking for separation in input space. Separability between documents can be improved when documents are implicitly mapped to feature space.

4.6.3 Sensitivity of KOKM based WSK to predefined parameters

Table 4.4: The impact of varying the decay factor λ when using KOKM based WSK.

Dataset		**Precision**	**Recall**	**F-measure**
	λ=0,1	0,458±0,045	0,698±0,023	**0,553±0,037**
	λ=0,3	0,443±0,027	0,664±0,096	0,531±0,050
Reuters	λ=0,5	0,434±0,060	0,705±0,093	0,536±0,058
	λ=0,7	0,431±0,042	0,708±0,103	0,535±0,063
	λ=0,9	0,440±0,065	0,705±0,054	0,540±0,031
	λ=0,1	0,320±0,021	0,587±0,070	0,413±0,011
	λ=0,3	0,325±0,022	0,616±0,042	**0,433±0,018**
Ohsumed	λ=0,5	0,323±0,021	0,587±0,035	0,416±0,017
	λ=0,7	0,310±0,014	0,614±0,080	0,411±0,028
	λ=0,9	0,307±0,014	0,695±0,050	**0,426±0,014**

We study in this section the sensitivity of KOKM based WSK to the value of the decay factor λ, the length of the subsequences n and the power of the sub polynomial kernel p. All these parameters should be initialized before performing KOKM based WSK. The first parameter $\lambda \in [0, 1]$ is used to penalize non contiguous subsequences. As well as λ is near to 0, the non contiguous word subsequences are considered *dissimilar* and then, the gap is more penalized. Table 4.4 reports obtained results using different values of λ used within KOKM based WSK in Reuters and Ohsumed datasets. Obtained F-measures are little sensitive to λ. These results prove that locality doesn't have an important impact

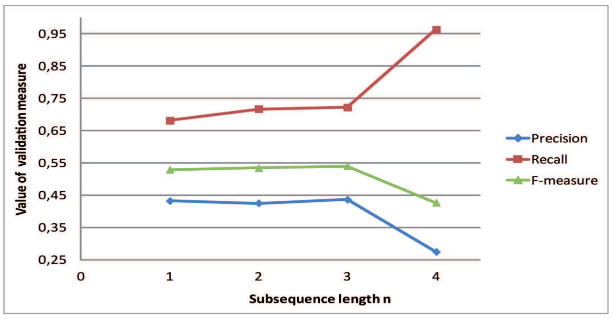
Table 4.5: The impact of varying the subsequence length n and the power p on the performance of KOKM-based-WSK.

Dataset	**Length**	**Power**	**Precision**	**Recall**	**F-measure**
	n=1	p=0,05	0,432±0,107	0,681±0,150	0,529±0,126
	n=1	p=0,20	0,401±0,103	0,689±0,129	0,503±0,114
	n=1	p=0,40	0,410±0,028	0,750±0,013	0,530±0,023
	n=1	p=0,60	0,383±0,029	0,769±0,018	0,511±0,027
	n=2	p=0,05	0,425±0,032	0,717±0,045	0,534±0,037
	n=2	p=0,10	**0,440±0,064**	0,705±0,054	**0,540±0,030**
	n=2	p=0,20	0,388±0,042	0,774±0,033	0,516±0,043
Reuters dataset	n=2	p=0,40	0,331±0,061	0,884±0,037	0,480±0,066
	n=2	p=0,60	0,277±0,006	0,787±0,509	0,391±0,113
	n=3	p=0,05	0,436±0,086	0,722±0,039	**0,540±0,080**
	n=3	p=0,10	0,426±0,072	0,729±0,058	0,537±0,065
	n=3	p=0,20	0,358±0,014	0,811±0,021	0,497±0,014
	n=3	p=0,40	0,274±0,008	0,963±0,037	0,427±0,012
	n=3	p=0,60	0,274±0,010	**0,968±0,032**	0,427±0,015
	n=1	p=0,05	0,323±0,027	0,710±0,041	**0,444±0,018**
	n=1	p=0,10	0,319±0,027	0,694±0,023	0,437±0,029
	n=1	p=0,20	0,304±0,022	0,725±0,040	0,428±0,014
	n=1	p=0,40	0,291±0,015	0,763±0,020	0,421±0,012
	n=1	p=0,60	0,280±0,001	0,860±0,012	0,423±0,000
	n=2	p=0,05	0,324±0,014	0,696±0,070	0,442±0,018
	n=2	p=0,10	0,308±0,012	0,696±0,038	0,426±0,015
Ohsumed dataset	n=2	p=0,20	0,279±0,012	0,760±0,046	0,408±0,014
	n=2	p=0,40	0,253±0,004	0,978±0,038	0,402±0,007
	n=2	p=0,60	0,249±0,006	**0,985±0,025**	0,397±0,010
	n=3	p=0,05	**0,330±0,017**	0,656±0,036	0,437±0,023
	n=3	p=0,10	0,319±0,014	0,709±0,004	0,440±0,012
	n=3	p=0,20	0,265±0,003	0,923±0,033	0,412±0,000
	n=3	p=0,40	0,249±0,006	**0,985±0,025**	0,397±0,010
	n=3	p=0,60	0,249±0,006	**0,985±0,025**	0,397±0,010

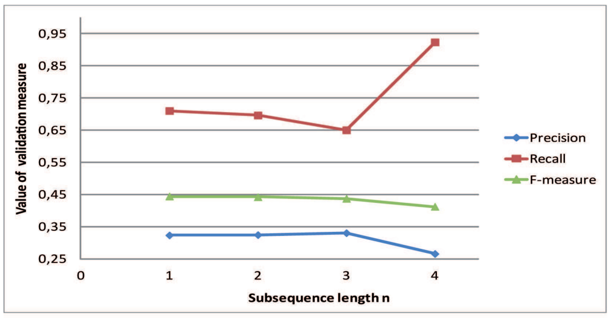
on the performance of the proposed method when applied to text clustering.

To study the sensitivity of KOKM based WSK to both parameters n and p, we fix $\lambda = 0.8$ and we perform experiments with different values of these parameters. Table 4.5 reports the average of obtained results in Reuters and Ohsumed datasets over ten runs with same initialization of cluster representatives. We notice a high sensitivity of KOKM based WSK to parameters n and p. For example, in Reuters dataset, obtained values of Precision lie between 0.274 and 0.440 and obtained values of Recalls lie between 0.681

and 0.968. Concerning the impact of n, as well as this parameter increases, obtained F-measures are slightly improved until $n = 4$ where F-measures are reduced as shown in Figure 4.4. These results can be explained by the difficulty to find similar subsequences of words between textual documents since the length of subsequences becomes larger than 3. In this case, documents are considered all different which explains the high value of Recall and the low value of Precision.



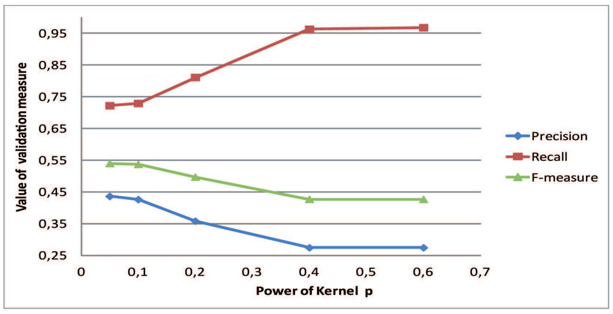
a) Reuters data set



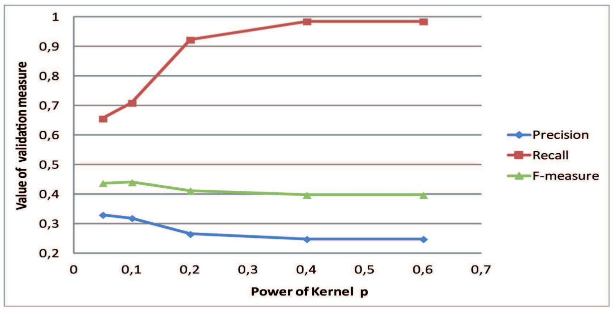
b) Ohsumed data set

Figure 4.4: The impact of varying subsequences length n on the performance of KOKM based WSK

Concerning the impact of p, as well as p decreases, the obtained Precision increases and obtained Recall decreases leading to the improvement of over all F-measure in both Reuters and Ohsumed datasets as illustrated in Figure 4.5. The high values of Recall (Recall $\longmapsto$ 1) which coincides with high values of p ($p \longmapsto 1$), are induced by the off-diagonal problem in the Gram Matrix. This problem is solved when using small values of p ($p \longmapsto 0$). For example, Recall is reduced from 0.968 to 0.722 when the value of p varies from 0.6 to 0.05.



a) Reuters data set



b) Ohsumed data set

Figure 4.5: The impact of varying power of sub polynomial kernel p on the performance of KOKM based WSK

4.7 Conclusion

We dealt in this chapter with the issue of identifying overlapping clusters for textual documents. We proposed the KOKM based WSK method which is able to detect non-disjoint groups from sequential textual documents based on WSK as similarity measure. Detecting overlapping groups by considering text as a sequence of words improves quality of obtained groups compared to VSM representation of text. Obtained results on Reuters and Ohsumed datasets show the efficiency of KOKM based WSK compared to overlapping methods using VSM.

This proposed method can be applied for many others application domains where textual data need to be assigned to more than one cluster. Some of these applications considers text document as a structured data such as Trees and Graphs. For such representation of text, it would be interesting to look for relevant overlapping groups.

Chapter 5

Extending Weka for Overlapping Clustering

Contents

5.1 Introduction

Most of the overlapping clustering methods presented in this thesis were developed, in an integrated toolbox, with the aim of being accessible and useful to researchers and developers working on a diverse range of overlapping clustering applications. The proposed toolbox, referred to as Weka4OC, is a Graphical User Interface (GUI) extending the well known data mining tool "Weka". The Toolbox is accessible as a java application or as a web based application (Applet) giving the possibility to make an overlapping clustering process on different types of data such as vectorial, structured and unstructured data. It can be integrated in any java application and can be executed on different operating system such as Windows or Linux.

This chapter is organized as the following: in the first section we present the main functionalities and characteristics of Weka4OC, then in Section 2 we present the loading and preprocessing process supported by the proposed software. In Section 3 we describe the different learning methods that can be used to perform non-disjoint partitioning while in Section 4 and Section 5 we respectively present the different outputs resulting from the learning process and the supported visualization options that can be performed within Weka4OC. Finally, we present in Section 6 the functionality of exporting results of clustering.

5.2 Main functionalities and characteristics of Weka4OC

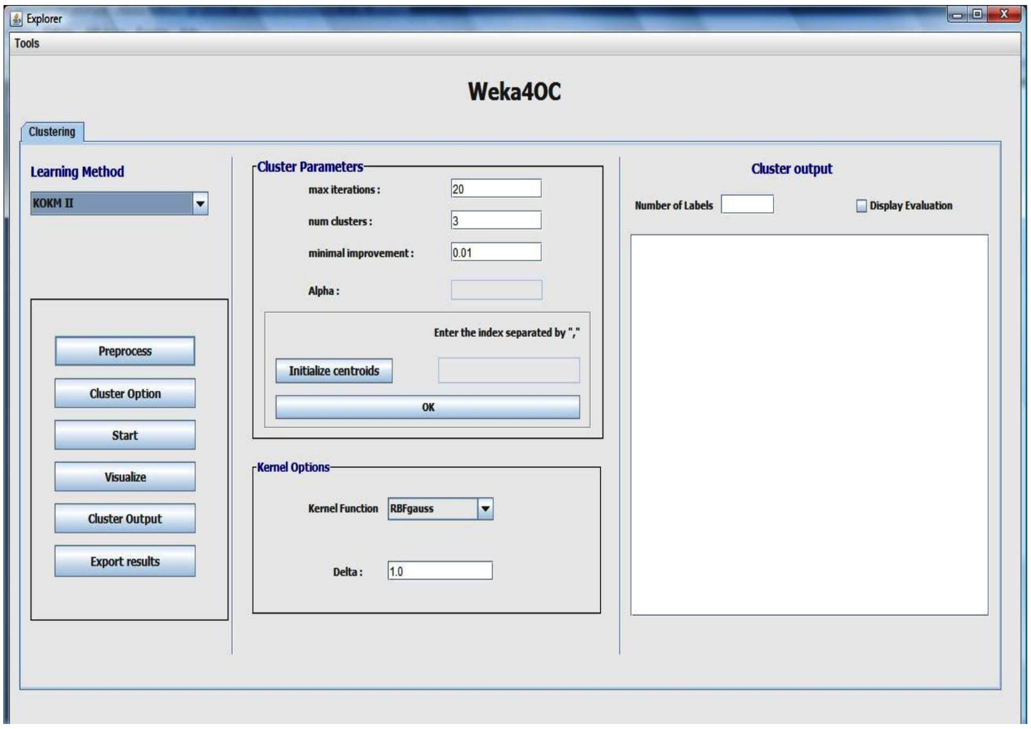
Weka4OC, formally called Weka for Overlapping Clustering, is a software that has been developed for the purpose of identifying non-disjoint groups from different types of data. Weka4OC has been developed within Weka data mining toolkit. This choice was made in order to take advantages of the vast resources of Weka on preprocessing data and the familiarity of many machine learning and data mining practitioners with this tool.

Weka4OC is freely available [1] under the GNU general public license agreement. It has been programmed in Java and is compatible with almost every computing platform.

[1]Weka4OC is available at: https://sourceforge.net/projects/overlappingclus

It can be easily installed and run (as given in Appendix B) and allows for quick set up and operation through a user friendly graphical interface (a use case of using Weka4OC is described in Appendix A).

The main user-related advantages of Weka4OC are: (a) it can be downloaded freely from the internet, (b) it can easily be functional by end-users that are not familiar with software programming, (c) it is easy and flexible in use in that it extends Weka tool and allows the user to specify different options for the analysis and (d) the results of the clustering can be evaluated, visualized and exported to different formats.



Learning Method: KOKM II

max iterations : 20
num clusters : 3
minimal improvement : 0.01
Alpha :

Kernel Function: RBFgauss
Delta : 1.0

Figure 5.1: Main interface of Weka4OC

Weka4OC supports many standard data mining tasks such as data preprocessing, clustering, evaluation of clustering and visualization. The main interface of the program, as reported in Figure 5.1, shows the main functionalities and the different parameterizable options which can be summarized as in the following:

- "Preprocess": enables to load many types of data saved in different data formats and enables many preprocessing techniques for customizing the data.

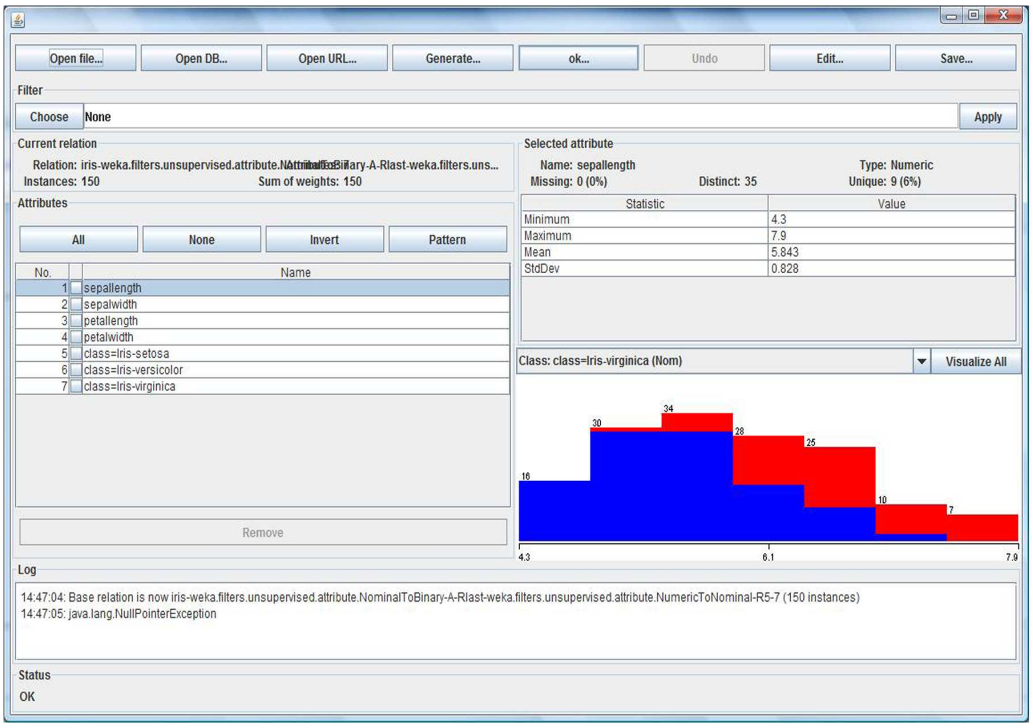
- "Cluster option": enables to ignore attributes that must be leave out from the analysis.
- "Clustering": enables to look for non-disjoint grouping of the loaded data.
- "Cluster output" : reports detailed descriptions concerning the data, the configuration of the clustering, the evaluation and the visualization of the obtained grouping.
- "Visualize": enables to visualize the loaded data.
- "Export": enables to export the obtained partitioning of data into a text file.

In the remaining sections, we detail each functionality supported by Weka4OC by describing the different parameterizable options enabling to perform non-disjoint clustering of data.

5.3 Loading and preprocessing data in Weka4OC

For loading data, Weka4OC offers several standard loaders that can be used by activating the functionality "Preprocess" as described in Figure 5.2. All data format supported by Weka could be used in Weka4OC including:

- Files: many standard files are supported by Weka4OC such as ARFF files, C4.5 Files, CSV files, JSON files and libsvm files.
- Databases: data could be loaded from relational data saved in commercial and open sources databases management systems (DBMS) such as Oracle, SqlServer, BD2, Mysql, etc. A specific ".jar" file must be added to the path of the application in order to perform the connection with the used DBMS. Data which must be loaded and used by Weka4OC can be specified using a customized SQL query.
- URL: If data are saved in a specified server, the Uniform Resource Locator (URL) of the server can be specified to load the data.



Current relation

Relation: iris-weka.filters.unsupervised.attribute.NominalToBinary-A-Rlast-weka.filters.uns... Attributes: 7
Instances: 150 Sum of weights: 150

Attributes

No.		Name
1		sepallength
2		sepalwidth
3		petallength
4		petalwidth
5		class=Iris-setosa
6		class=Iris-versicolor
7		class=Iris-virginica

Selected attribute

Name: sepallength Type: Numeric
Missing: 0 (0%) Distinct: 35 Unique: 9 (6%)

Statistic	Value
Minimum	4.3
Maximum	7.9
Mean	5.843
StdDev	0.828

Class: class=Iris-virginica (Nom)

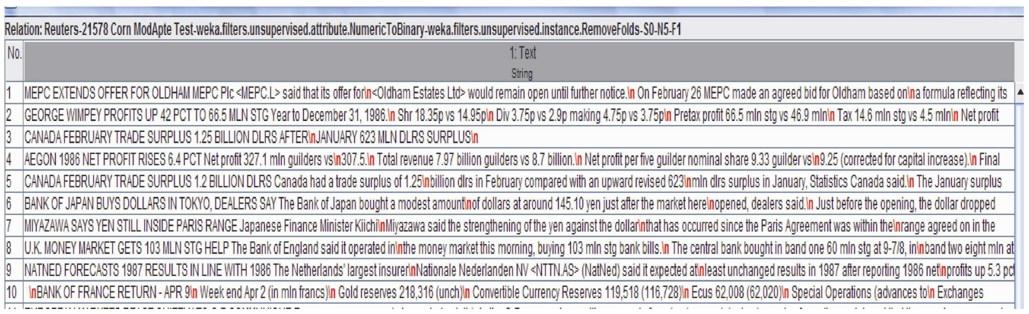
Log

```
14:47:04: Base relation is now iris-weka.filters.unsupervised.attribute.NominalToBinary-A-Rlast-weka.filters.unsupervised.attribute.NumericToNominal-R5-7 (150 instances)
14:47:05: java.lang.NullPointerException
```

Status
OK

Figure 5.2: Loading and Preprocessing data with Weka4OC

We notice that specific loader, such as text directory loader, Matlab loader and XRFF loader can be used to import these data formats. For example, textual files (".txt" or ".html") saved in different folders can be loaded as a string attribute and transformed to vector space model data. We also notice that using the word sequence approach, described in Chapter 4, requires data to be loaded as string attribute containing all the textual description of the document. Figure 5.3 shows an example of a set of documents from Reuters dataset which can be learned using the word sequence approach.



Relation: Reuters-21578 Corn ModApte Test-weka.filters.unsupervised.attribute.NumericToBinary-weka.filters.unsupervised.instance.RemoveFolds-S0-N5-F1

No.	1: Text String
1	MEPC EXTENDS OFFER FOR OLDHAM MEPC Plc <MEPC.L> said that its offer for\n<Oldham Estates Ltd> would remain open until further notice.\n On February 26 MEPC made an agreed bid for Oldham based on\na formula reflecting its
2	GEORGE WIMPEY PROFITS UP 42 PCT TO 66.5 MLN STG Year to December 31, 1986.\n Shr 18.35p vs 14.95p\n Div 3.75p vs 2.9p making 4.75p vs 3.75p\n Pretax profit 66.5 mln stg vs 46.9 mln\n Tax 14.6 mln stg vs 4.5 mln\n Net profit
3	CANADA FEBRUARY TRADE SURPLUS 1.25 BILLION DLRS AFTER\nJANUARY 623 MLN DLRS SURPLUS\n
4	AEGON 1986 NET PROFIT RISES 6.4 PCT Net profit 327.1 mln guilders vs\n307.5.\n Total revenue 7.97 billion guilders vs 8.7 billion.\n Net profit per five guilder nominal share 9.33 guilder vs\n9.25 (corrected for capital increase).\n Final
5	CANADA FEBRUARY TRADE SURPLUS 1.2 BILLION DLRS Canada had a trade surplus of 1.25\nbillion dlrs in February compared with an upward revised 623\nmln dlrs surplus in January, Statistics Canada said.\n The January surplus
6	BANK OF JAPAN BUYS DOLLARS IN TOKYO, DEALERS SAY The Bank of Japan bought a modest amount\nof dollars at around 145.10 yen just after the market here\nopened, dealers said.\n Just before the opening, the dollar dropped
7	MIYAZAWA SAYS YEN STILL INSIDE PARIS RANGE Japanese Finance Minister Kiichi\nMiyazawa said the strengthening of the yen against the dollar\nthat has occurred since the Paris Agreement was within the\nrange agreed on in the
8	U.K. MONEY MARKET GETS 103 MLN STG HELP The Bank of England said it operated in\nthe money market this morning, buying 103 mln stg bank bills.\n The central bank bought in band one 60 mln stg at 9-7/8, in\nband two eight mln at
9	NATNED FORECASTS 1987 RESULTS IN LINE WITH 1986 The Netherlands' largest insurer\nNationale Nederlanden NV <NTTN.AS> (NatNed) said it expected at\nleast unchanged results in 1987 after reporting 1986 net\nprofits up 5.3 pct
10	\nBANK OF FRANCE RETURN - APR 9\n Week end Apr 2 (in mln francs)\n Gold reserves 218,316 (unch)\n Convertible Currency Reserves 119,518 (116,728)\n Ecus 62,008 (62,020)\n Special Operations (advances to\n Exchanges

Figure 5.3: Loading textual data as a String attribute in Weka4OC for learning groups using the word sequence approach

Once data loaded, Weka4OC offers different preprocessing techniques to customize the data such as attribute transformation, class transformation, instances filtering, attributes filtering, etc. In fact, all the preprocessing techniques implemented in Weka are inherited and can be used to configure the loaded data. We recommend to save the preprocessed data in a separate file, like ".arff" file, then reload these data to make the learning process. We notice that loading "Class" attributes within data is optional. However, "Class" attributes are required for evaluating the robustness of the learning method using outputs and visualization functionalities. In that case, each class must be loaded as a separate binary attribute indicating membership of each observation to the following class. The validation of all the loading and preprocessing tasks can be done using the button "OK" which switches to the main interface of Weka4OC.

5.4 Learning methods in Weka4OC

Different learning methods are available in Weka4OC and can be selected using the listbox "Learning Method" which allows to choose a method to be performed for grouping the set of observations. Five methods can be chosen: OKM, KOKMII, KOKM based WSK, Parameterized R-OKM and k-means. We note that all these learning methods lead to non-disjoint partitioning, except k-means which is given as baseline. If "Class" attributes are loaded within the collection of data or any other attributes which should not be included in the analysis, the GUI allows to ignore some attributes when learning clusters by using the functionality "Cluster option". Once parameters and attributes are configured, the learning process can be performed with the specified parameters using the button "Start". We give in the following a description of parameters and options that should be configured, for each method, before performing the learning.

5.4.1 OKM

The OKM method has 3 parameters that should be configured which are the maximal number of iterations, the number of clusters, and the minimal improvement in the objective function. The parameters of OKM can be configured in the tab “Cluster Parameters” and is described by:

- Max iterations: the maximal number of iterations of the main algorithm.
- Num cluster: the number of clusters to be considered.
- Minimal improvement: the minimal improvement in the objective function between two iterations which is used for the convergence of the method. If the improvement is less than the specified value as minimal improvement, the used learning method returns partitioning obtained at the latest iteration. By default, this value is initialized to 0.01.
- Initialize centroids (optional): this functionality can be used to manually configure the positions of observations which will be used as initial representative of clusters, known as seeds. The position of observations to be considered as seeds should be given separated by comma and must exactly match with the number of clusters specified in “Num cluster”. For example, for initializing representative of 3 clusters using the first three observations in the dataset we use “1,2,3”. If this functionality is not activated, observations used as initial clusters’ representatives are chosen randomly.

5.4.2 Parameterized R-OKM

In addition to the parameters described for OKM, Parameterized R-OKM needs to configure the parameter α used to control the size of overlaps. The value of $\alpha \in \mathbb{R}$ can be given in the text field “Alpha” in the tab “Cluster Parameters”. When using Parameterized R-OKM with $\alpha = 0$, obtained results exactly coincide with results of OKM. Instead, when using Parameterized R-OKM with $\alpha = 1$, the obtained results are equivalent to R-OKM as described in Chapter 3. If α is not given, an error message indicates that Parameterized R-OKM needs to configure the value of α. In fact, the value of α controls the overlaps between the clusters. As α increases the size of overlaps are reduced until leading to non-overlapping clustering.

5.4.3 KOKMII and KOKM based WSK

For KOKMII, in addition to parameters used for OKM, kernel options must be configured before performing the learning.

- Kernel function: this parameter must be configured for methods which incorporate kernels. Many standard Kernel functions can be chosen in the listbox responding to data representation model. “Linear”, “Polynomial”, “Laplace”, “RBF Gaussian” and “RBF Exponential” kernels could be configured within KOKMII if data have a numeric vector representation. However, if data are loaded as a string attribute (for example textual data), “String” kernel should be configured to perform the KOKM based WSK method as described in Chapter 4. We used an online implementation of all the kernel function to avoid the pre-computing of the kernel matrix. For the implementation of WSK, we used a recursive definition proposed by Lodhi et al. (2001) and based on the dynamic programming technique. The advantage of this implementation is to reduce the time complexity and to perform WSK without explicitly extracting word sequences.
- Value of the kernel parameter: this option indicates the value of the parameter of the considered kernel function. For example, for Polynomial kernel the parameter “d” must be configured, for RBF kernels the parameter “σ” is required and for WSK the length of the sequence of words “n” is required.

5.5 Outputs and evaluation of clustering in Weka4OC

Once the learning achieved, Weka4OC gives several outputs about the performed learning process and the resulting clusters. These outputs are reported in the tab “Cluster output”. Two alternatives can be configured: *Basic outputs* or *outputs with evaluation of clustering*. The basic outputs are reported without the evaluation of clusters while the second alternative report this information. The alternative of basic outputs is configured by default, however, the second alternative can be activated by checking the option “Display Evaluation” in the tab “Cluster output”. The evaluation of the resulted clusters is optional because it can be time-consuming for data having a huge number of observations and classes or it can not be performed if data are unlabeled which is the case of most of real life applications.

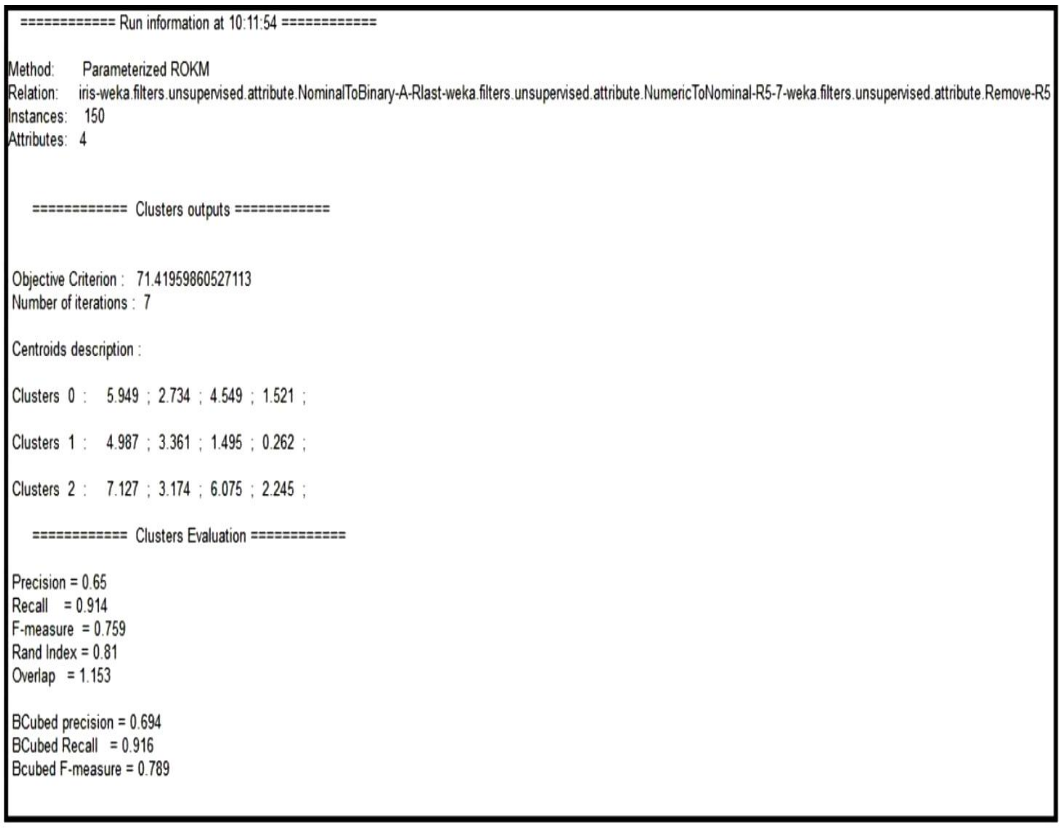
5.5.1 Basic Outputs

Weka4OC reports a summary of the different configurations of the learning process, the considered data and the resulted clusters. Figure 5.4 and Figure 5.5 show an example of clustering outputs obtained using Parameterized R-OKM for 3 clusters on Iris dataset. Reported information are as the following:

1. the learning method considered in learning process
2. the number of observations (instances) considered in the learning process
3. the number of attributes considered in the learning process
4. the value of the optimized objective criterion
5. the number of iterations of the main algorithm
6. the final clusters' representatives
7. the description of the final binary assignment matrix (described in Figure 5.5) where rows represent observations, columns represent clusters and the internal value "1" indicates memberships of observation i to the respective clusters.

5.5.2 Evaluation of the resulting clustering

The evaluation functionality gives users the possibility to check the effectiveness and the robustness of the learning methods among others existing ones. As described in Section 1.5, the evaluation of overlapping clustering is based on comparing scores for Precision-Recall measures on datasets with known labels. These measures check if groups obtained with the overlapping clustering method are similar to the actual groups in the dataset. The



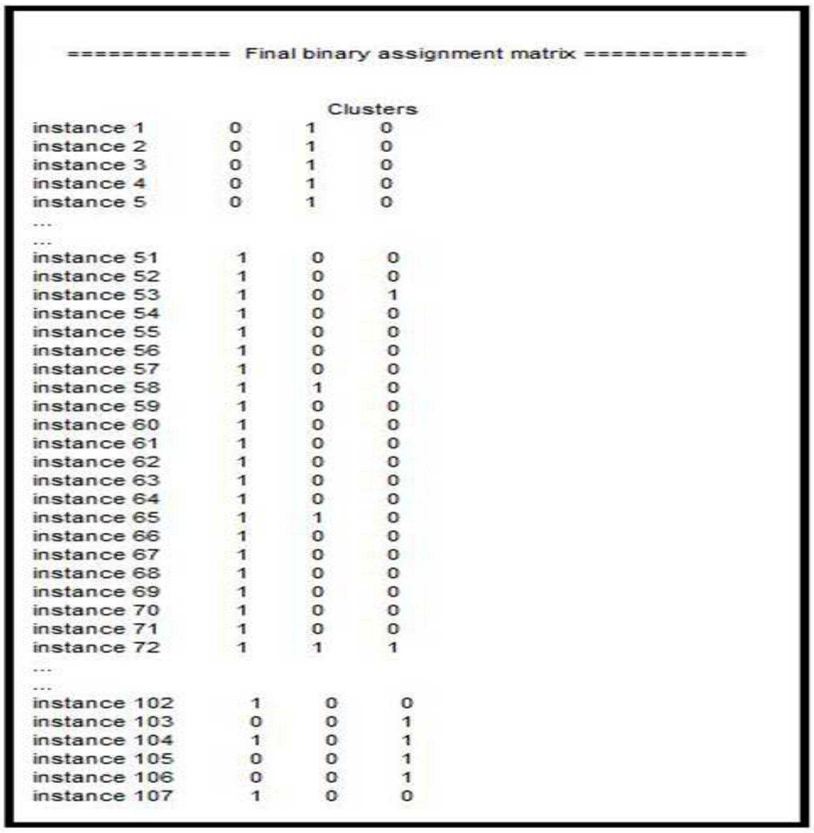
```
 ============ Run information at 10:11:54 ============

Method:      Parameterized ROKM
Relation:    iris-weka.filters.unsupervised.attribute.NominalToBinary-A-Rlast-weka.filters.unsupervised.attribute.NumericToNominal-R5-7-weka.filters.unsupervised.attribute.Remove-R5
Instances:   150
Attributes:  4


   ============ Clusters outputs ============


Objective Criterion :  71.41959860527113
Number of iterations :  7

Centroids description :

Clusters  0 :    5.949 ; 2.734 ; 4.549 ; 1.521 ;

Clusters  1 :    4.987 ; 3.361 ; 1.495 ; 0.262 ;

Clusters  2 :    7.127 ; 3.174 ; 6.075 ; 2.245 ;

   ============ Clusters Evaluation ============

Precision = 0.65
Recall    = 0.914
F-measure  = 0.759
Rand Index = 0.81
Overlap   = 1.153

BCubed precision = 0.694
BCubed Recall   = 0.916
Bcubed F-measure = 0.789
```

Figure 5.4: Example of clusters output obtained using Parameterized R-OKM with $\alpha = 1$ for 3 clusters on Iris dataset.

evaluation functionality in Weka4OC requires the following data format and parameters to be configured:

- Data format: data must be loaded with “Class” attributes. “Class” attributes must be formatted as a binary attribute (one column) for each label. All “Class” attributes must be put after all the exploratory attributes (columns at the end represent the “Class” attributes).
- “Display evaluation”: this option must be checked
- Number of Labels: the number of “Class” attributes to be considered for performing the evaluation process. (number of columns to be considered as “Class” attributes).

We notice that the functionality “Cluster option” must be used to ignore all “Class” attributes before performing the learning. This option must be configured at each start



```
============ Final binary assignment matrix ============

                          Clusters
instance 1       0      1      0
instance 2       0      1      0
instance 3       0      1      0
instance 4       0      1      0
instance 5       0      1      0
...
...
instance 51      1      0      0
instance 52      1      0      0
instance 53      1      0      1
instance 54      1      0      0
instance 55      1      0      0
instance 56      1      0      0
instance 57      1      0      0
instance 58      1      1      0
instance 59      1      0      0
instance 60      1      0      0
instance 61      1      0      0
instance 62      1      0      0
instance 63      1      0      0
instance 64      1      0      0
instance 65      1      1      0
instance 66      1      0      0
instance 67      1      0      0
instance 68      1      0      0
instance 69      1      0      0
instance 70      1      0      0
instance 71      1      0      0
instance 72      1      1      1
...
...
instance 102      1      0      0
instance 103      0      0      1
instance 104      1      0      1
instance 105      0      0      1
instance 106      0      0      1
instance 107      1      0      0
```

Figure 5.5: Final binary assignment matrix obtained using Parameterized R-OKM with $\alpha = 1$ for 3 clusters on Iris dataset.

of learning. The reported evaluation measures includes two evaluation techniques: Pair-based and BCubed-based. For each evaluation technique, Precision, Recall and F-measure are reported. Additionally, the size of overlaps build by the learning method is also reported. Figure 5.4 shows an example of the evaluation process of Parameterized R-OKM on iris dataset.

5.6 Visualization of data and clusters in Weka4OC

Weka4OC offers a visualization tool to discover both the data and the resulting clusters. For data visualization, we integrate the same panel used in Weka that displays a scatter plot matrix for the current dataset as reported in Figure 5.6. This panel can be activated using the functionality "Visualize". If no dataset is loaded, a message box indicates that users must load data before using the visualization functionality. The size of the individual

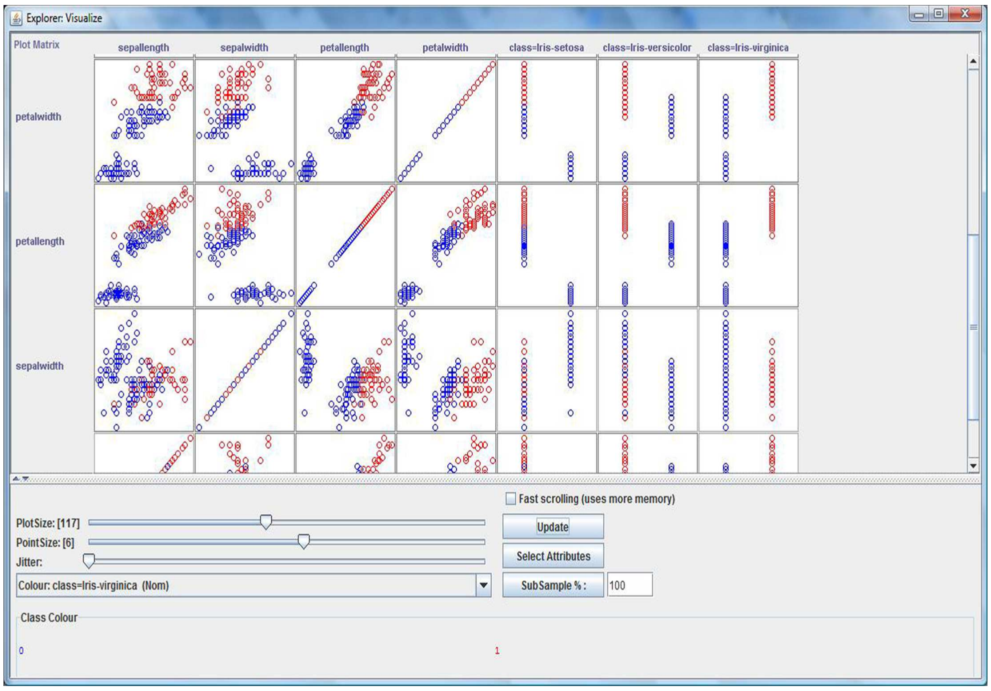


Figure 5.6: Visualization of data using Weka4OC

cells and the size of the points displayed can be adjusted using the slider controls at the bottom of the panel. A zooming functionality can be used by clicking on a cell in the matrix to pops up a larger plot panel that displays the content of the selected cell. This panel allows users to visualize the current dataset in one and two dimensions.

However, for visualizing the resulted clusters, Weka4OC offers an interactive panel where the non-disjoint patterns can be easily showed and analyzed. All the performed learning are saved in the results buffer for a further interactive visual comparison of the

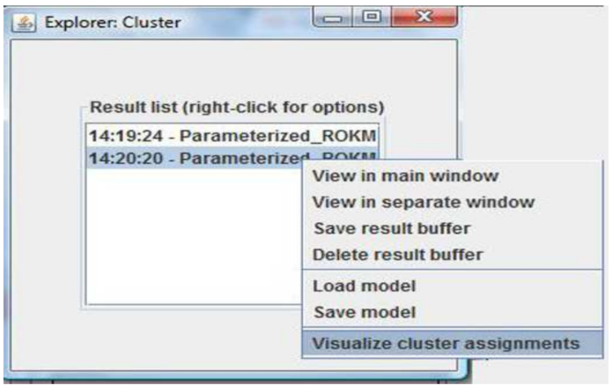

Explorer: Cluster
Result list (right-click for options)
14:19:24 - Parameterized_ROKM
14:20:20 - Parameterized_ROKM
View in main window
View in separate window
Save result buffer
Delete result buffer
Load model
Save model
Visualize cluster assignments

Figure 5.7: Partitioning saved in the buffer of results for visualization.

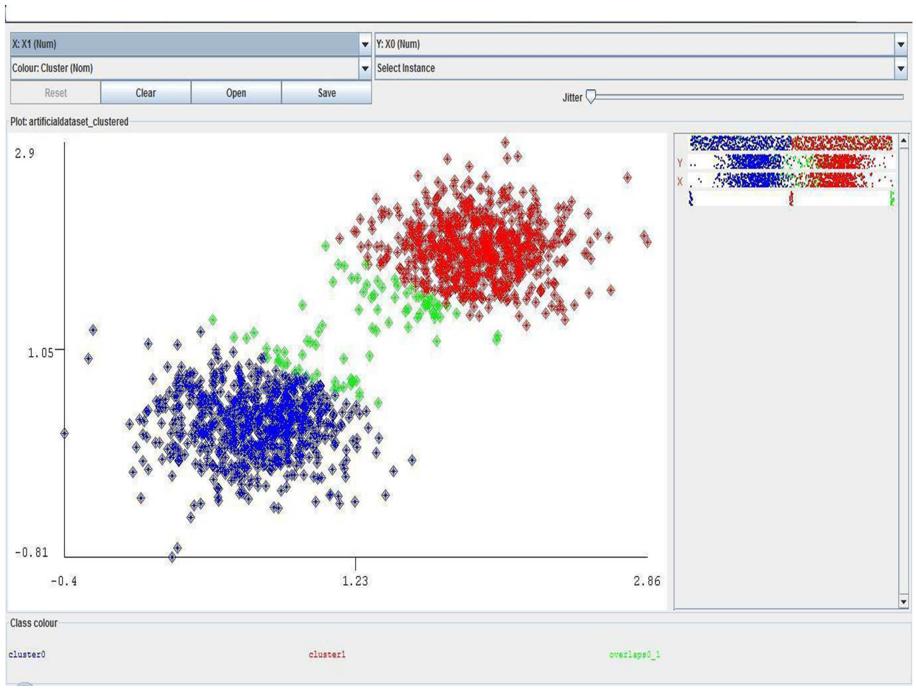

X: X1 (Num)
Y: X0 (Num)
Colour: Cluster (Nom)
Select Instance

Figure 5.8: Example of patterns obtained using Parameterized R-OKM with 2 clusters

obtained patterns. By activating the functionality “Cluster Output”, the panel containing all the learning results is showed. Figure 5.7 shows an example of saved results of the parameterized R-OKM method with two different configurations. By a simple right click on the desired outputs, the visualization can be done by choosing the option “Visualize cluster assignments”.

This option shows a new panel which allows users to visualize the selected groupings in one and two dimensions, responding to attributes selected in the top of the panel. Selected attributes can be easily configured by choosing among the list of attributes in the X or Y axes thus, giving users more flexible visualization and analysis of the outputs. Figure 5.8 shows an example of a non-disjoint partitioning resulting from the Parameterized R-OKM for two clusters in a two dimensional artificial dataset: the red and the blue points constitute observations which are assigned to the first or to the second cluster however, the green points are observations which belong to the intersection of the two clusters. Furthermore, users can inspect attributes and classes for each visualized data points by left clicking on the desired data where statistics related to the indicated observation are showed. It is possible to save the visualization of clusters out to an ARFF file using the

"Save" button and then, load it back to Weka4OC. The two dimensional visualization of clusters can be exported to image file by holding down "shift", "alt" and left-clicking on the panel to save. Available formats include: BMP, JPEG, PNG and postscript. We also notice that similarly to Weka, the "Jiter" option moves slightly the individual points so that in the event of close data points users can reveal hidden multiple occurrences within the initial plot.

5.7 Export of results in Weka4OC

In order to enable further processing of the obtained output, the final binary matrix can be exported and saved in a separated textual file (".txt") using the functionality "Export results". This functionality exports the last outputs performed by the learning method. The saved binary matrix can be used as inputs for other programs, like Matlab, for further analysis and evaluation of the resulting clusters.

5.8 Conclusion

In order to facilitate for data mining practitioner the task of overlapping clustering, we have proposed an interactive software, referred to as Weka4OC, extending the well known data mining tool Weka. Weka4OC supports different types of data and offers many learning techniques for producing a non-disjoint partitioning. Produced patterns can be easily evaluated and analyzed through the visualization tool.

Conclusion

In this thesis, we dealt with the issue of identifying non-disjoint groups from unlabeled data referred to as *overlapping clustering* (Banerjee et al. 2005; Depril et al. 2008; Cleuziou 2013). Although several clustering methods have been proposed in the literature, most of them build disjoint and exclusive clusters. They ignore the possibility that objects belong to more than one cluster. However, there are several applications like document clustering (Gil-García and Pons-Porrata 2010; Pérez-Suárez et al. 2013), social network (Tang and Liu 2009; Wang et al. 2010; Fellows et al. 2011) and image classification (Snoek et al. 2006), where it is common that objects belong to many clusters. For these kinds of applications, overlapping clustering is useful and important.

The study of existing overlapping clustering methods (Qinand and Suganthan 2004; Depril et al. 2008; Masson and Denoeux 2008) in the literature has shown that these methods suffer from several recurrent issues, such as the inability to look for nonlinear separations between clusters, the inability to regulate the sizes of overlaps and the inability to deal with non-vectorial data, among others, which may reduce their performance in real life applications. These recurrent issues motivate researchers to design more perfective overlapping methods.

Challenges and contributions

The first challenge that motivates our works was the identification of non *disjoint clusters with nonlinear boundaries* between clusters. Although data structuring are usually complex with spherical and non-spherical shapes, existing overlapping methods look only for linear separations between clusters. Our contribution consisted on proposing two kernel based methods that produce overlapping clusters with both *linear* and *nonlinear* separations by using Mercer kernel technique. The use of kernel has allows to look for separations in high feature space where the separability of clusters is improved. The first proposed method Kernel Overlapping K-means I (KOKMI) is a centroid based method that generalizes kernel k-means to produce overlapping clusters with nonlinear and non spherical shapes. The second proposed method Kernel Overlapping K-means II (KOKMII) is a medoid based method that is an improvement of KOKMI in terms of efficiency and in terms of algorithm complexity. Obtained results on artificial and real datasets have shown the efficiency of proposed methods.

We also dealt with the challenge of controlling sizes of overlaps between clusters. This characteristic has an important impact on the performance and the validity of overlapping methods. Usually, existing overlapping methods in the literature produce clusterings without possibility of control on the size or the quality of the overlaps. These methods lead to clusters with large overlaps and fail to build an acceptable size of overlaps. Ideally, the method should reveal the clustering that best fit to the data by evaluating partitionings with different sizes of overlaps. Accordingly, we proposed a generic overlapping clustering model which generalizes k-means for the identification of clusters with regulation of overlaps. Different instantiations of the model were defined offering new overlapping layouts with fixed, parameterized or auto-adjusted sizes for the overlaps. Empirical results performed on artificial and real overlapping datasets have shown the efficiency of the proposed model compared to the exiting methods.

Furthermore, we designed an overlapping clustering method able to look for non-disjoint partitioning from sequential textual documents. In fact, grouping documents is an important application that motivates overlapping clustering while each document can discuss several themes and then, it must belong to several groups. However, the application of a learning method for grouping textual document usually requires to prepare a set of documents for numerical analysis (Saracoglu et al. 2007; Aliguliyev 2009) by using the Vector Space Model (VSM). This representation of text avoids correlation between terms and does not give importance to the order of words in the text. By considering all these important constraints, our contribution consisted on the design of KOKM based WSK method which considers textual data as an ordered sequences of words (n-Grams)(Yannakoudakis et al. 1990) and uses WSK as similarity measure between documents. The use of WSK takes into account the correlation between adjacent words in text and keeps their order as they appear in the text. The application of KOKM based WSK to different benchmarks corpus showed that the obtained partitioning could be improved compared to partitioning obtained with existing methods.

Finally, in order to make accessible and useful to researchers and developers working on diverse range of overlapping clustering, most of the clustering methods presented in this thesis and other existing ones were developed in an integrated software referred to as "Weka4OC". This software extends the well known data mining tool "Weka" and offers for users friendly interfaces allowing a quick set up and operation for performing, visualizing, and evaluating non-disjoint partitioning.

Future works

The presented works in this thesis open several directions for future researches. The use of Kernel methods in KOKMI and KOKMII offers many alternatives to perform non-disjoint partitionings in real cases where the input data cannot be described by explicit feature vectors but described by trees or histograms. For such types of data, it would be interesting to investigate the application of an overlapping clustering process using specific designed kernels in the literature such as the Histogram kernel (Barla et al. 2003).

The application of KOKM based WSK to sequential textual document gives a promising results which could be improved by integrating internal and external knowledge. There appears to be a growing interest in the further incorporation of semantic kernel like the the Latent Semantic Kernel (LSK) or using external knowledge taxonomy such as Wordnet or Wikipedia which could be used to improve the semantic relatedness of the terms. In addition, the KOKM based WSK would be applied in other domains where the input data require non-disjoint partitioning and are described by sequential patterns as the case of grouping genes into different metabolic processes. In such application, it would be interesting to make overlapping clustering based on the sequential description of each gene. However, specific kernels should be designed instead of the Word Sequence Kernel used in KOKM based WSK.

Concerning the proposed generic model for identifying overlapping clusters with overlap regulation, we notice that this model supports other principles for controlling and regularizing the overlaps. For example, one could propose to control the overlaps regardless the dispersal of the cluster prototypes from the data to assign. By this way, overlaps would be reduced as the prototypes are more distant. Other ways to control overlaps could be realized by the combination of two regulation principles: for example the two proposed instantiations, Parameterized R-OKM and Adjusted R-OKM could be combined in a common model.

All the proposed methods in these works adopt the geometrical model to take into account the possibility of overlaps. However, the additive model could be also adopted which would offer an interesting way to compare the efficiency of additive and geometrical models for overlapping clustering applications.

In this thesis, we dealt with three challenges which motivate overlapping clustering researches. Many other active challenges could be considered to design more perfective and efficient learning process such as the identification of non-disjoint groups when data contain outliers. The presence of outliers affects the resulting clusters and yields to clusters which do not fit the true structure of data. Designing an overlapping method with outliers identification would make robust the detection of overlapping clusters.

Bibliography

Agrawal, R., J. Gehrke, D. Gunopulos, and P. Raghavan (1998). Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, SIGMOD '98, New York, USA, pp. 94–105.

Aliguliyev, R. M. (2009). Clustering of document collection: A weighting approach. *Expert Systems with Applications 36*(4), 7904–7916.

Amigo, E., J. Gonzalo, J. Artiles, and F. Verdejo (2009). A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval 12*(4), 461–486.

Anderberg, M. (1973). *Cluster analysis for applications.* Probability and mathematical statistics. Academic Press.

Ankerst, M., M. M. Breunig, H.-P. Kriegel, and J. Sander (1999). Optics: ordering points to identify the clustering structure. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, SIGMOD '99, pp. 49–60.

Banerjee, A., C. Krumpelman, S. Basu, R. J. Mooney, and J. Ghosh (2005). Model based overlapping clustering. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, Chicago, USA, pp. 532–537.

Barla, A., F. Odone, and A. Verri (2003). Histogram intersection kernel for image classification. In *Proceedings of the International Conference on Image Processing (ICIP)*, Barcelone, Spain, pp. 513–516.

Baumes, J., M. Goldberg, and M. Magdon-Ismail (2005). Efficient identification of overlapping communities. In *Proceedings of the IEEE International Conference on Intelligence and Security Informatics*, ISI'05, Atlanta, GA, pp. 27–36.

Ben-Hur, A., D. Horn, H. T. Siegelmann, and V. Vapnik (2001). Support vector clustering. *Journal of Machine Learning Research 2*, 125–137.

BenN'Cir, C., G. Cleuziou, and N. Essoussi (2013a). Identification of non-disjoint clusters with small and parameterizable overlaps. In *Proceedings of the IEEE International Conference on Computer Applications Technology (ICCAT)*, Sousse, Tunisia, pp. 1–6.

BenN'Cir, C., G. Cleuziou, and N. Essoussi (2013b). Un modèle générique pour les k-moyennes a recouvrements ajustables. In *Proceedings of the Workshop EGC 2013: Fouille de données complexes*, Toulouse, France, pp. 81–92.

BenN'Cir, C., G. Cleuziou, and N. Essoussi (2014a). Généralisation des k-moyennes pour produire des recouvrements ajustables. In *Proceedings of the International Conference on Extraction et Gestion des Connaissances (EGC)*, Rennes, France, pp. 221–232.

BenN'Cir, C., G. Cleuziou, and N. Essoussi (2014b). Generalization of c-means for identifying non-disjoint clusters with overlap regulation. *International Journal of Pattern Recognition Letters 45*, 92–98.

BenN'Cir, C. and N. Essoussi (2011). Classification recouvrante basée sur les méthodes à noyau. In *On Actes of the 43 International Conference Journées de statistiques (JDS 2011)*, Gammart, Tunisia.

BenN'Cir, C. and N. Essoussi (2012a). Improved overlapping k-means for detecting overlapping clusters. In *In Proceedings of the Third Meeting on Statistics and Data Mining MSDM'12*, Hammamet, Tunisia, pp. 137–142.

BenN'Cir, C. and N. Essoussi (2012b). Overlapping patterns recognition with linear and non-linear separations using positive definite kernels. *International Journal of Computer Applications (IJCA) 56*, 1–8.

BenN'Cir, C., A. Zenned, and N. Essoussi (2013). Non-disjoint grouping of text documents based word sequence kernel. In *Proceedings of the International Conference on Extraction et Gestion des Connaissances (EGC)*, Toulouse, France, pp. 253–262.

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

Bezdek, J. C. (1981). Pattern recognition with fuzzy objective function algoritms. *Plenum Press 4(2)*, 67–76.

Bonchi, F., A. Gionis, and A. Ukkonen (2011). Overlapping correlation clustering. In *Proceedings of the IEEE International Conference on Data Mining*, ICDM'11, vancouver, Canada, pp. 51–60.

Bonchi, F., A. Gionis, and A. Ukkonen (2013). Overlapping correlation clustering. *Knowledge and Information Systems 35*(1), 1–32.

Camastra, F. and A. Verri (2005). A novel kernel method for clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27*, 801–804.

Cancedda, N., E. Gaussier, C. Goutte, and J. Renders (2003). Word-sequence kernels. *Journal of Machine Learning Research 3*, 1059–1082.

Chan, E. Y., W.-K. Ching, M. K. Ng, and J. Z. Huang (2004). An optimization algorithm for clustering using weighted dissimilarity measures. *Pattern Recognition 37*(5), 943–952.

Chapelle, O., B. Schölkopf, and A. Zien (Eds.) (2006). *Semi-Supervised Learning.* MIT Press.

Cleuziou, G. (2008). An extended version of the k-means method for overlapping clustering. In *Proceedings of the International Conference on Pattern Recognition ICPR*, Florida, USA, pp. 1–6.

Cleuziou, G. (2009). Two variants of the okm for overlapping clustering. *Advances in Knowledge Discovery and Management 2*, 149–166.

Cleuziou, G. (2013). Osom: A method for building overlapping topological maps. *Pattern Recognition Letters 34*(3), 239–246.

Cortes, C. and V. Vapnik (1995). Support vector networks. *Machine Learning 20*, 273–297.

Cristianini, N., C. Campbell, and C. Burges (2002). Editorial: Kernel methods: Current research and future directions. *Machine Learning 46*(1-3), 5–9.

Davis, G. B. and K. M. Carley (2008). Clearing the fog: Fuzzy, overlapping groups for social networks. *Social Networks 30*(3), 201–212.

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

Depril, D., I. V. Mechelen, and T. F. Wilderjans (2012). Lowdimensional additive overlapping clustering. *Journal of Classification 29*(3), 297–320.

Depril, D., I. Van Mechelen, and B. Mirkin (2008). Algorithms for additive clustering of rectangular data tables. *Computational Statistics and Data Analysis 52*(11), 4923–4938.

DeSarbo, W. and W. Cron (1988). A maximum likelihood methodology for clusterwise linear regression. *Journal of Classification 5*(2), 249–282.

Diday, E. (1984). Orders and overlapping clusters by pyramids. Technical Report 730, INRIA, France.

Duda, R., P. Hart, and D. Stork (2001). *Pattern Classification.*

Ester, M., H.-P. Kriegel, J. Sander, and X. Xu (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the*

Second International Conference on Knowledge Discovery and Data Mining, pp. 226–231.

Fay, M. P. and M. A. Proschan (2010). Wilcoxon-Mann-Whitney or t-test? On assumptions for hypothesis tests and multiple interpretations of decision rules. *Statistics Surveys 4*, 1–39.

Fellows, M. R., J. Guo, C. Komusiewicz, R. Niedermeier, and J. Uhlmann (2011). Graph-based data clustering with overlaps. *Discrete Optimization 8*(1), 2–17.

Filippone, M., F. Camastra, F. Masulli, and S. Rovetta (2008). A survey of kernel and spectral methods for clustering. *Pattern Recognition 41*(1), 176–190.

Fraley, C. and A. E. Raftery (2003). Enhanced model-based clustering, density estimation, and discriminant analysis software: Mclust. *Journal of Classification 20*(2), 263–286.

Fu, Q. and A. Banerjee (2008). Multiplicative mixture models for overlapping clustering. In *Proceedings of the IEEE International Conference on Data Mining*, ICDM'08, Washington, USA, pp. 791–796.

Gil-García, R. and A. Pons-Porrata (2010). Dynamic hierarchical algorithms for document clustering. *Pattern Recogn. Lett. 31*(6), 469–477.

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

Goil, S., H. Nagesh, and A. Choudhary (1999). Mafia: Efficient and scalable subspace clustering for very large data sets. Technical report.

Goldberg, M., S. Kelley, M. Magdon-Ismail, K. Mertsalov, and A. Wallace (2010). Finding overlapping communities in social networks. In *Proceedings of the IEEE International Conference on Social Computing (SocialCom)*, pp. 104–113.

Graepel, T. and K. Obermayer (1998). Fuzzy topographic kernel clustering. In *Proceedings of the GI Workshop on Fuzzy Neuro Systems*, pp. 90–97.

Gregory, S. (2007). An algorithm to find overlapping community structure in networks. In *Knowledge Discovery in Databases: PKDD 2007*, Volume 4702 of *Lecture Notes in Computer Science*, pp. 91–102.

Gregory, S. (2008). A fast algorithm to find overlapping communities in networks. In *Machine Learning and Knowledge Discovery in Databases*, Volume 5211 of *Lecture Notes in Computer Science*, pp. 408–423.

Guha, S., R. Rastogi, and K. Shim (1998). Cure: an efficient clustering algorithm for large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, SIGMOD '98, Washington, USA, pp. 73–84.

Guha, S., R. Rastogi, and K. Shim (2000). Rock: A robust clustering algorithm for categorical attributes. *Information System 25*(5), 345–366.

Han, J. and M. Kamber (2000). *Data Mining: Concepts and Techniques.* Morgan Kaufmann Publishers Inc.

Hasan, M. A., S. Salem, and M. J. Zaki (2011). Simclus: an effective algorithm for clustering with a lower bound on similarity. *Knowl. Inf. Syst. 28*(3), 665–685.

Hattum, P. and H. Hoijtink (2009). Market segmentation using brand strategy research: Bayesian inference with respect to mixtures of log-linear models. *Journal of Classification 26*(3), 297–328.

Heller, K. and Z. Ghahramani (2007). A nonparametric bayesian approach to modeling overlapping clusters. In *Proceedings of AISTATS*, San-juan, Puerto Ric, pp. 187–194.

Hinneburg, A. and H.-H. Gabriel (2007). Denclue 2.0: fast clustering based on kernel density estimation. In *Proceedings of the International Conference on Intelligent Data Analysis*, IDA'07, Berlin, Heidelberg, pp. 70–80.

Hinneburg, A. and D. A. Keim (1998). An efficient approach to clustering in large multimedia databases with noise. In *KDD*, Nevada,USA, pp. 58–65.

Inokuchi, R. and S. Miyamoto (2004). Lvq clustering and som using a kernel function. In *Proceedings of IEEE International Conference on Fuzzy Systems*, pp. 1497–1500.

Isa, D., V. Kallimani, and L. H. Lee (2009). Using the self organizing map for clustering of text documents. *Expert Systems with Applications 36*(5), 9584–9591.

Jain, A. K. (2010). Data clustering: 50 years beyond k-means. *Pattern Recognition Letters 31*(8), 651–666.

Jardine, N. and C. van Rijsbergen (1971). The use of hierarchical clustering in information retrieval. *Information Storage and Retrieval 7*, 217–240.

Jason, W., S. Bernhard, E. Eleazar, L. Christina, and N. William (2003). Dealing with large diagonals in kernel matrices. *Annals of the Institute of Statistical Mathematics 55*(2), 391–408.

Karypis, G., Eui, and V. K. News (1999). Chameleon: Hierarchical Clustering Using Dynamic Modeling. *Computer 32*(8), 68–75.

Kaufman, L. and P. J. Rousseeuw (2008). *Partitioning Around Medoids*, pp. 68–125. John Wiley and Sons, Inc.

Krishnapuram, R. and J. M. Keller (1993). A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems 1*, 98–110.

Lingras, P. and C. West (2004). Interval set clustering of web users with rough k-means. *Journal of Intelligent Information System 23*(1), 5–16.

Liu, Y., C. Wu, and M. Liu (2011). Research of fast som clustering for text information. *Expert Systems with Applications 38*(8), 9325–9333.

Liu, Z.-G., J. Dezert, G. Mercier, and Q. Pan (2012). Belief c-means: An extension of fuzzy c-means algorithm in belief functions framework. *Pattern Recognition Letters 33*(3), 291–300.

Lodhi, H., N. Cristianini, J. Shawe-Taylor, and C. Watkins (2001). Text classication using string kernel. *The Journal of Machine Learning Research 2*, 419–444.

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Berkeley Symposium on Mathematical Statistics and Probability*, Volume 1, pp. 281–297.

Magdon-Ismail, M. and J. Purnell (2011). Ssde-cluster: Fast overlapping clustering of networks using sampled spectral distance embedding and gmms. In *Proceedings of the IEEE International Conference on Social Computing (socialcom)*, pp. 756–759.

Mann, H. and D. Whitney (1947). On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics 18*, 50–60.

Masson, M.-H. and T. Denoeux (2008). Ecm: An evidential version of the fuzzy c-means algorithm. *Pattern Recognition 41*(4), 1384 – 1397.

Mirkin, B. G. (1987). Method of principal cluster analysis. *Automation and Remote Control 48*, 1379–1386.

Mirkin, B. G. (1990). A sequential fitting procedure for linear data analysis models. *Journal of Classification 7*(2), 167–195.

Ng, R. and J. Han (2002). Clarans: A method for clustering objects for spatial data mining. *IEEE Trans. on Knowl. and Data Eng. 14*(5), 1003–1016.

Oberreuter, G. and J. D. Velsquez (2013). Text mining applied to plagiarism detection: The use of words for detecting deviations in the writing style. *Expert Systems with Applications 0*(9), 3756–3763.

Pérez-Suárez, A., J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, and J. E. Medina-Pagola (2013). Oclustr: A new graph-based algorithm for overlapping clustering. *Neurocomputing 109*(0), 1–14.

Pérez-Suárez, A., J. F. Martnez-Trinidad, J. A. Carrasco-Ochoa, and J. E. Medina-Pagola (2013). An algorithm based on density and compactness for dynamic overlapping clustering. *Pattern Recognition 46*(11), 3040–3055.

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

Rokach, L. (2010). A survey of clustering algorithms. *Data Mining and Knowledge Discovery Handbook*, 269–298.

Saracoglu, R., K. Ttuncu, and N. Allahverdi (2007). A fuzzy clustering approach for finding similar documents using a novel similarity measure. *Expert Systems with Applications 33*(3), 600–605.

Saracoglu, R., K. Tutuncu, and N. Allahverdi (2008). A new approach on search for similar documents with multiple categories using fuzzy clustering. *Expert Systems with Applications 34*(4), 2545–2554.

Schikuta, E. and M. Erhart (2002). In *Proceedings of the international conference on Intelligent data analysis*, IDA'02, manchester, United Kindom, pp. 513–524.

Schölkopf, B., A. Smola, and K.-R. Müller (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation 10*(5), 1299–1319.

Selim, S. Z. and M. A. Ismail (1984). K-means-type algorithms: A generalized convergence theorem and characterization of local optimality. *IEEE Transactions on Pattern Analysis and Machine Intelligence 6*(1), 81–87.

Sheikholeslami, G., S. Chatterjee, and A. Zhang (2000). Wavecluster: a wavelet-based clustering approach for spatial data in very large databases. *The VLDB Journal 8*(3-4), 289–304.

Snoek, C. G. M., M. Worring, J. C. van Gemert, J.-M. Geusebroek, and A. W. M. Smeulders (2006). The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of the ACM International Conference on Multimedia*, MULTIMEDIA '06, New York, USA, pp. 421–430.

Suárez, A. P., J. F. M. Trinidad, J. A. Carrasco-Ochoa, and J. E. Medina-Pagola (2009). A new incremental algorithm for overlapped clustering. In *CIARP*, pp. 497–504.

Tang, L. and H. Liu (2009). Scalable learning of collective behavior based on sparse social dimensions. In *Proceedings of the ACM Conference on Information and knowledge Management*, Hong Kong, China, pp. 1107–1116.

Trohidis, K., G. Tsoumakas, G. Kalliris, and I. P. Vlahavas (2008). Multi-label classification of music into emotions. In *Proceedings of the International Symposium on Music Information Retrieval ISMIR*, Philadelphia ,USA, pp. 325–330.

Tsoumakas, G., I. Katakis, and I. Vlahavas (2010). Mining Multi-label Data. In *Data Mining and Knowledge Discovery Handbook*, Chapter 34, pp. 667–685.

Wang, Q. and E. Fleury (2011). Uncovering overlapping community structure. In *Complex Networks*, Volume 116 of *Communications in Computer and Information Science*, pp. 176–186.

Wang, W., J. Yang, and R. R. Muntz (1997). Sting: A statistical information grid approach to spatial data mining. In *Proceedings of the International Conference on Very Large Data Bases*, VLDB '97, San Francisco, USA, pp. 186–195.

Wang, X., L. Tang, H. Gao, and H. Liu (2010). Discovering overlapping groups in social media. In *Proceedings of IEEE International Conference on Data Mining*, Sydney, AUS, pp. 569–578.

Wieczorkowska, A., P. Synak, and Z. Ras (2006). Multi-label classification of emotions in music. In *Intelligent Information Processing and Web Mining*, Volume 35 of *Advances in Soft Computing*, pp. 307–315.

Wilderjans, T., E. Ceulemans, I. Mechelen, and D. Depril (2011). Adproclus: A graphical user interface for fitting additive profile clustering models to object by variable data matrices. *Behavior Research Methods 43*(1), 56–65.

Wilderjans, T. F., D. Depril, and I. V. Mechelen (2012). Additive biclustering: A comparison of one new and two existing als algorithms. *Journal of Classification 30*(1), 56–74.

Wu, Z., W. Xie, and J. Yu (2003). Fuzzy c-means clustering algorithm based on kernel method. In *Proceedings of the International Conference on Computational Intelligence and Multimedia Applications*, ICCIMA '03, Washington, USA.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval 1*, 67–88.

Yannakoudakis, E., I. Tsomokos, and P. Hutton (1990). n-grams and their implication to natural language understanding. *Pattern Recognition 23*(5), 509–528.

Zhang, D. and S. Chen (2002). Fuzzy clustering using kernel method. In *Proceedings of the International Conference on Control and Automation*, Xiamen, China, pp. 123–127.

Zhang, D. and S. Chen (2003). Kernel-based fuzzy and possibilistic c-means clustering. In *Proceedings of the International Conference on Artificial Neural Networks*, Istanbul, Turkey, pp. 122–125.

Zhang, D. and S. Chen (2004). A novel kernelised fuzzy c-means algorithm with application in medical image segmentation. *Artificial Intelligence in Medicine 32*(1), 37–50.

Zhang, S., R.-S. Wang, and X.-S. Zhang (2007a). Identification of overlapping community structure in complex networks using fuzzy c-means clustering. *Physica A: Statistical Mechanics and its Applications 374*(1), 483–490.

Zhang, S., R.-S. Wang, and X.-S. Zhang (2007b). Identification of overlapping community structure in complex networks using fuzzy c-means clustering. *Physica A:*

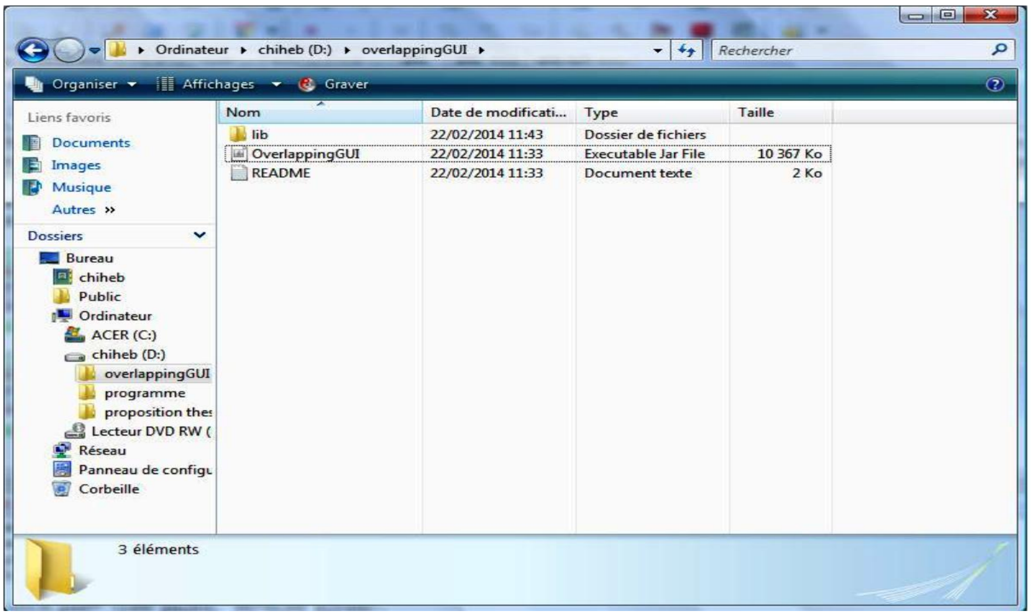
Statistical Mechanics and its Applications 374 (1), 483–490.

Zhang, T., R. Ramakrishnan, and M. Livny (1996). Birch: an efficient data clustering method for very large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, SIGMOD '96, Montreal, Canada, pp. 103–114.

A. Overlapping clustering using Weka4OC

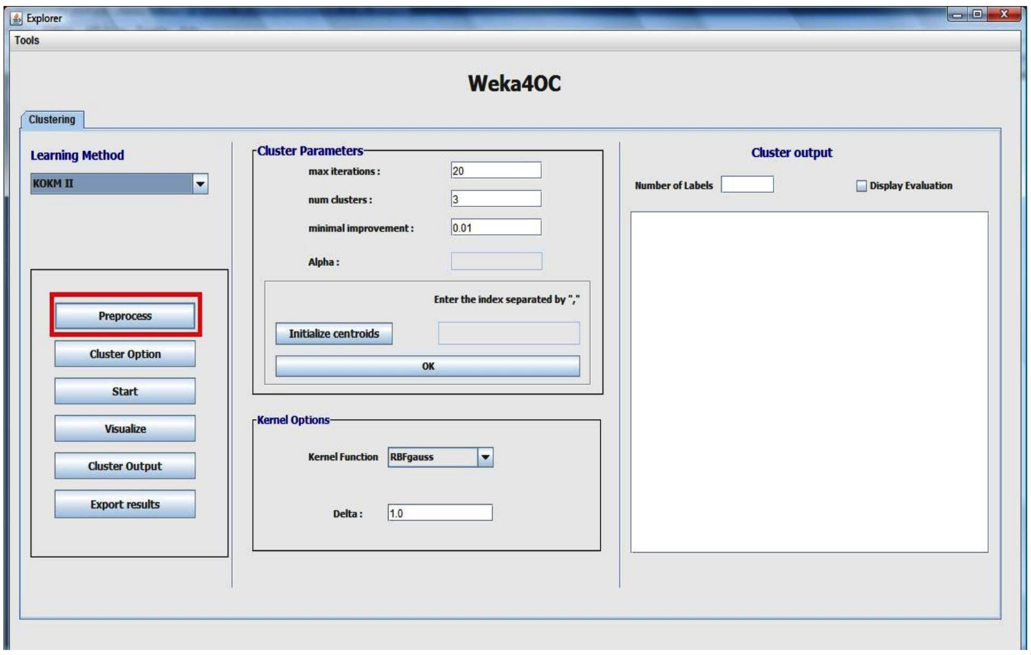
We describe in the following a use case of using Weka4OC for performing overlapping clustering using KOKMII method on Iris dataset.

1) Double click to run the GUI (require a Java Runtime Environment (JRE) 6.0 or later)



Nom	Date de modificati...	Type	Taille
lib	22/02/2014 11:43	Dossier de fichiers	
OverlappingGUI	22/02/2014 11:33	Executable Jar File	10 367 Ko
README	22/02/2014 11:33	Document texte	2 Ko

2) Load and preprocess data



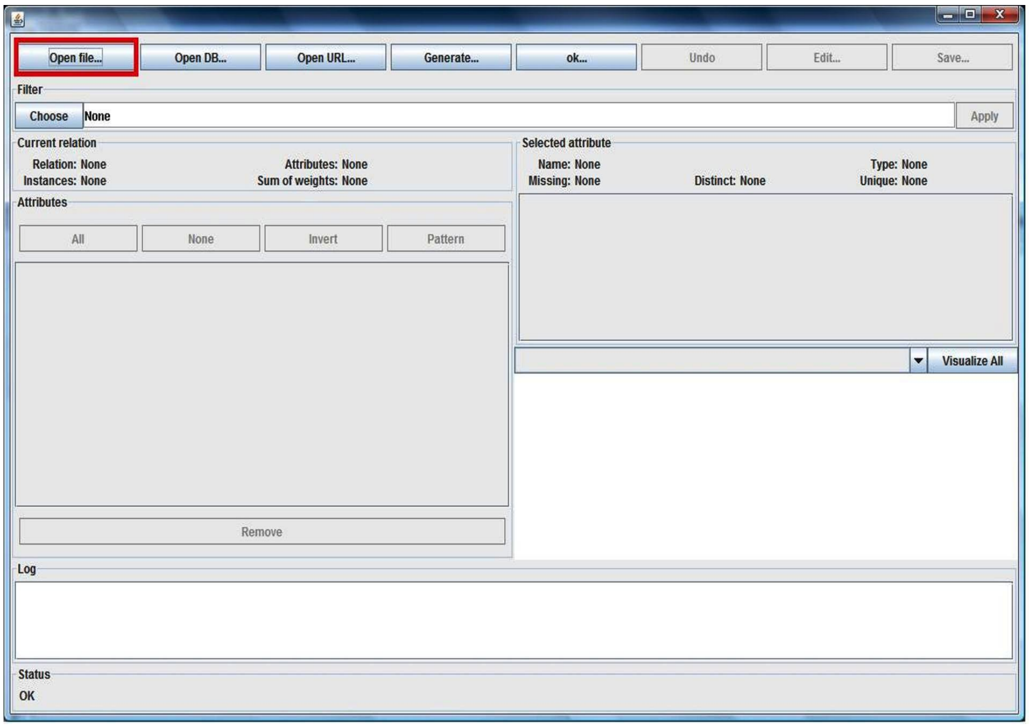
Learning Method: KOKM II

Cluster Parameters
max iterations : 20
num clusters : 3
minimal improvement : 0.01
Alpha :
Enter the index separated by ","

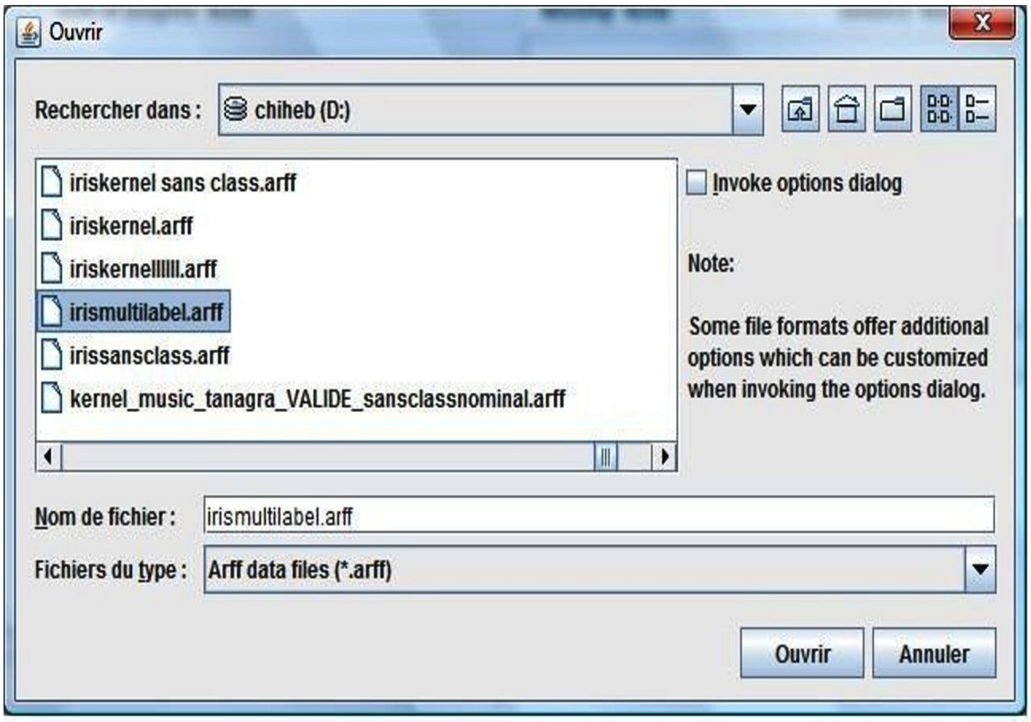
Kernel Options
Kernel Function: RBFgauss
Delta : 1.0

Number of Labels

3) Choose a file containing the description of the data set Iris.



Relation: None
Instances: None
Attributes: None
Sum of weights: None
Name: None
Missing: None
Distinct: None
Type: None
Unique: None
OK



chiheb (D:)

iriskernel sans class.arff
iriskernel.arff
iriskernelllllll.arff
irismultilabel.arff
irissansclass.arff
kernel_music_tanagra_VALIDE_sansclassnominal.arff

Invoke options dialog

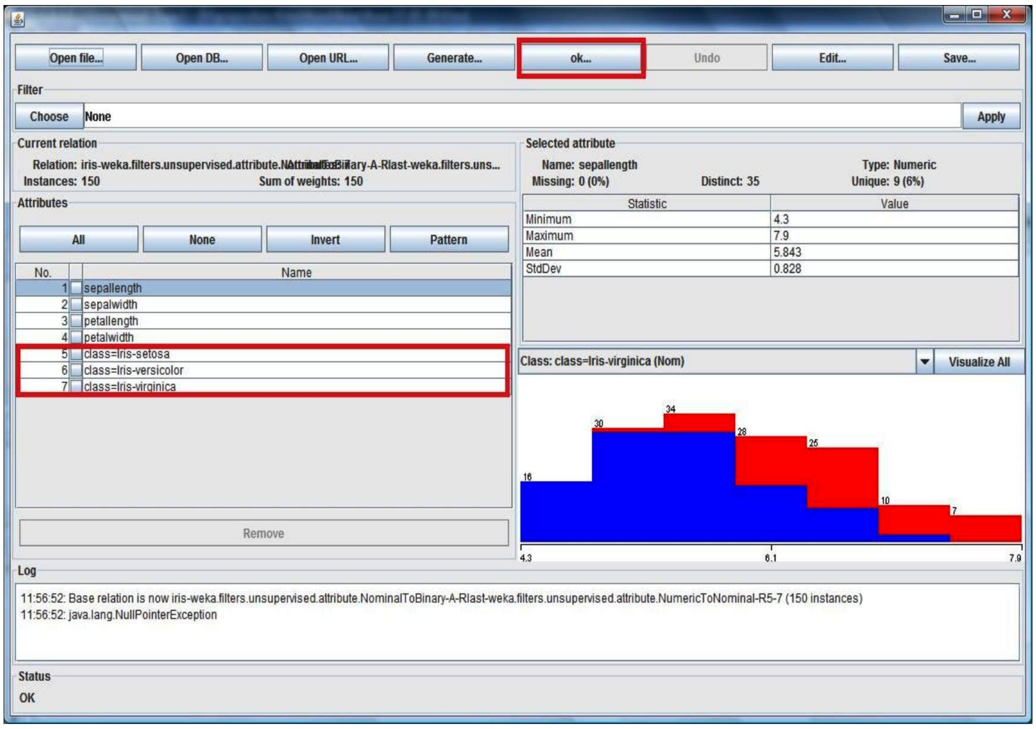
Note:

Some file formats offer additional options which can be customized when invoking the options dialog.

Nom de fichier : irismultilabel.arff

Fichiers du type : Arff data files (*.arff)

4) The loaded data contain 4 attributes describing the data and 3 binary attributes describing the class of each observation.



Filter: None

Current relation
Relation: iris-weka.filters.unsupervised.attribute.NominalToBinary-A-Rlast-weka.filters.uns...
Instances: 150 Sum of weights: 150

Attributes

No.	Name
1	sepallength
2	sepalwidth
3	petallength
4	petalwidth
5	class=Iris-setosa
6	class=Iris-versicolor
7	class=Iris-virginica

Selected attribute
Name: sepallength Type: Numeric
Missing: 0 (0%) Distinct: 35 Unique: 9 (6%)

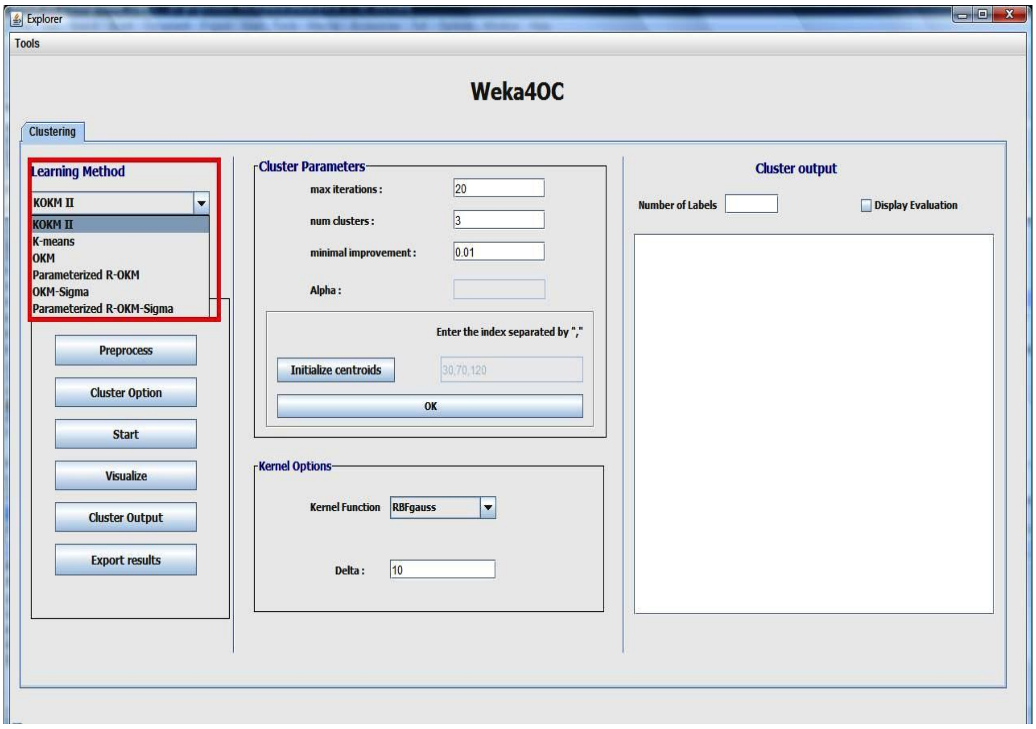
Statistic	Value
Minimum	4.3
Maximum	7.9
Mean	5.843
StdDev	0.828

Class: class=Iris-virginica (Nom)

Log
```
11:56:52: Base relation is now iris-weka.filters.unsupervised.attribute.NominalToBinary-A-Rlast-weka.filters.unsupervised.attribute.NumericToNominal-R5-7 (150 instances)
11:56:52: java.lang.NullPointerException
```

Status
OK

5) Choose the learning method KOKMII.



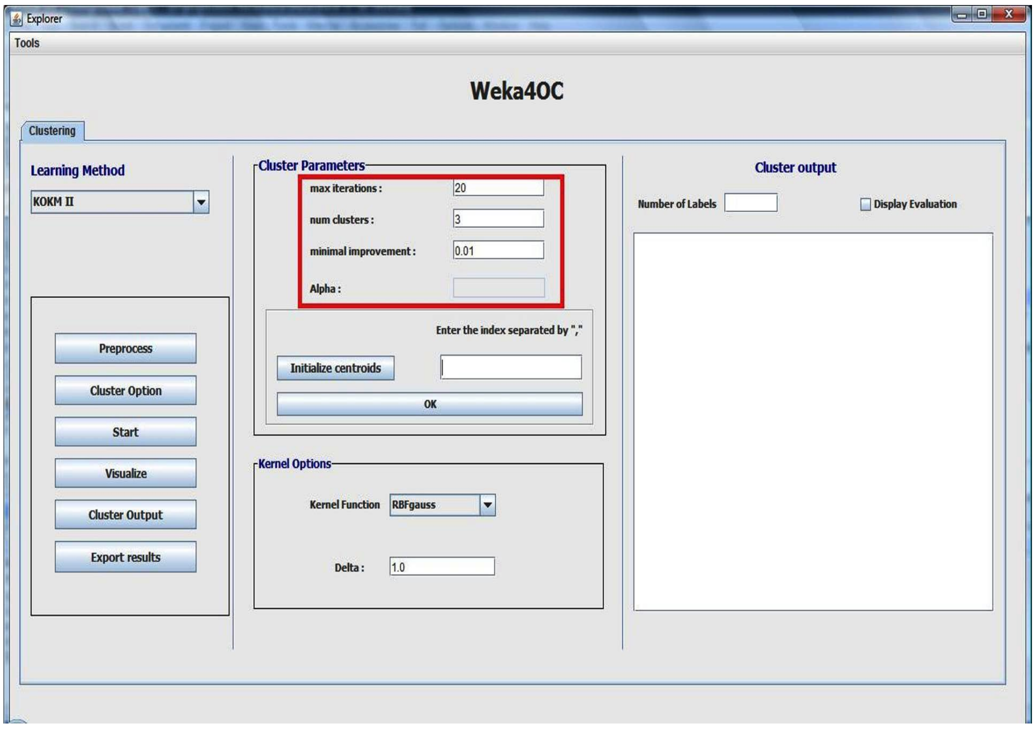
Learning Method: KOKM II

Cluster Parameters
max iterations : 20
num clusters : 3
minimal improvement : 0.01
Alpha :
Enter the index separated by ",": 30,70,120

Kernel Options
Kernel Function: RBFgauss
Delta : 10

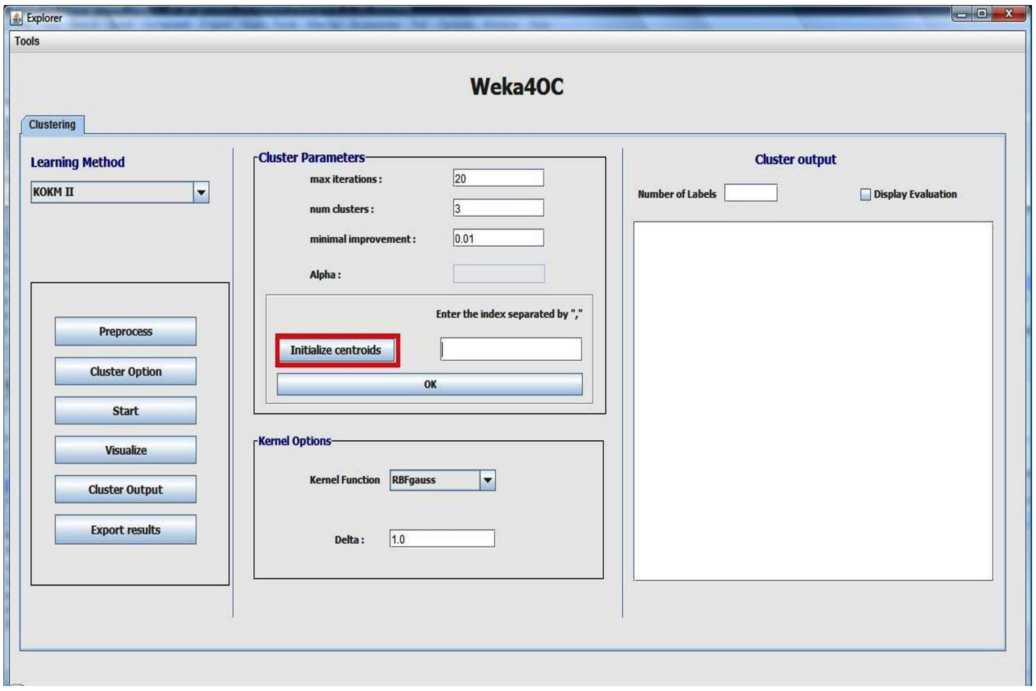
Number of Labels:

6) Configure cluster Parameters. We consider the following parameters: Max iteration= 20; num cluster= 3; minimal improvement= 0.01;



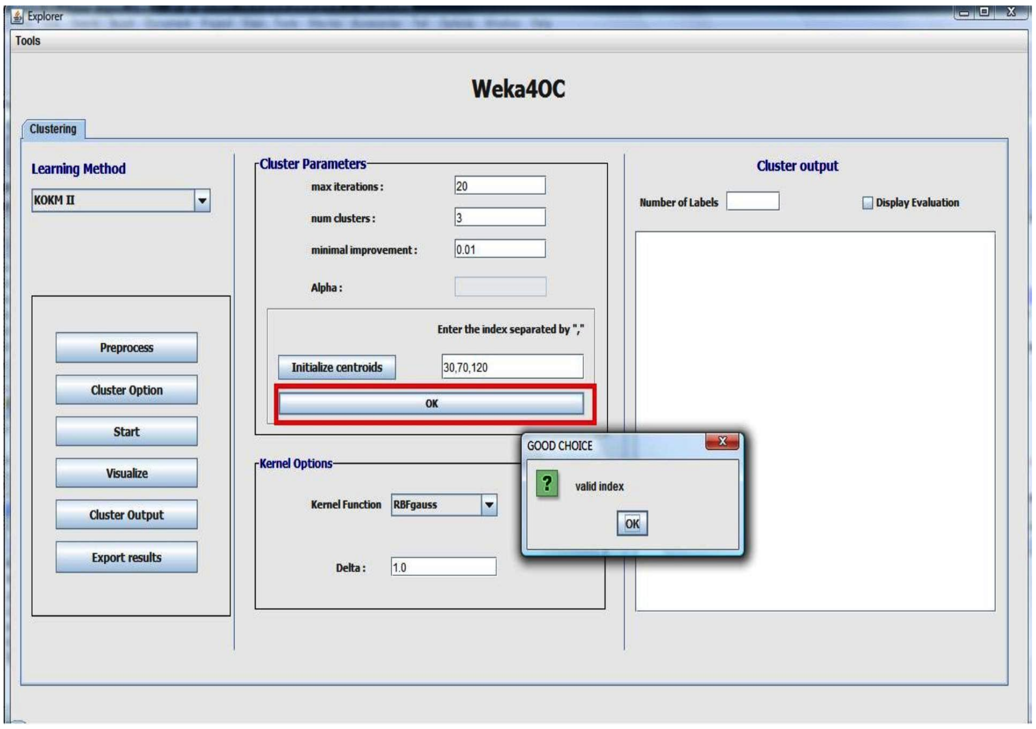
Learning Method: KOKM II
max iterations : 20
num clusters : 3
minimal improvement : 0.01
Alpha :
Kernel Function: RBFgauss
Delta : 1.0

7) Initialize clusters representatives. We consider the following observations as initial seeds: 30, 70, 120.



Learning Method: KOKM II
max iterations : 20
num clusters : 3
minimal improvement : 0.01
Alpha :
Kernel Function: RBFgauss
Delta : 1.0

8) Validate initial seeds



KOKM II

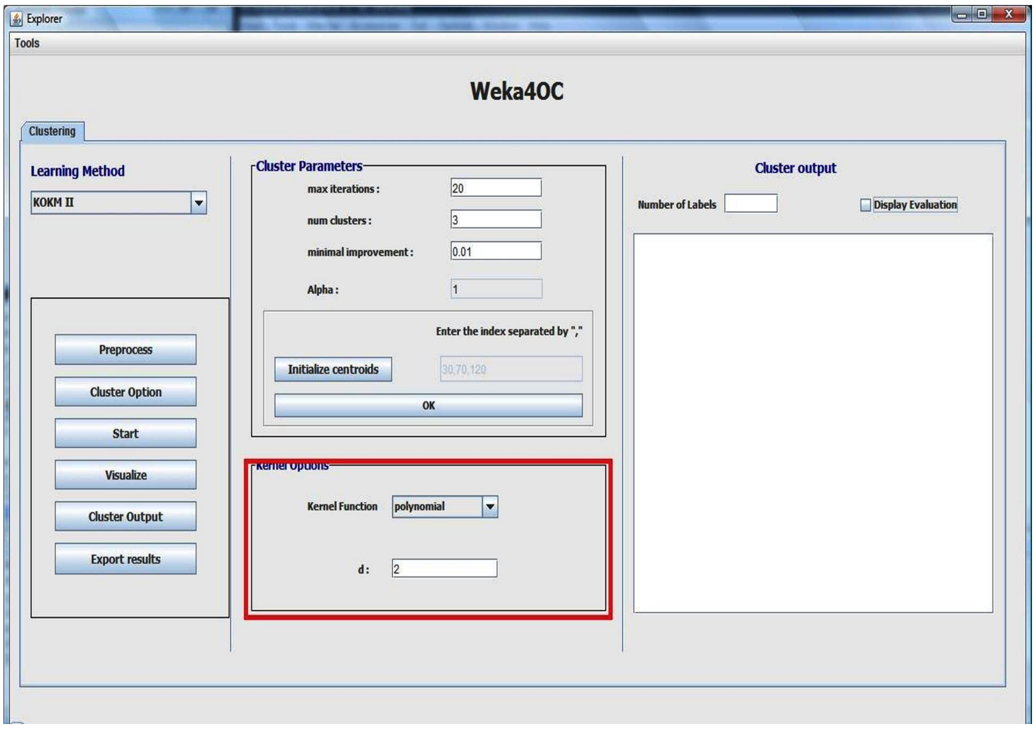
Cluster Parameters
max iterations : 20
num clusters : 3
minimal improvement : 0.01
Alpha :
Enter the index separated by ","
30,70,120

Kernel Options
Kernel Function RBFgauss
Delta : 1.0

Number of Labels

valid index

9) Configure kernel options. These parameters are required only for kernel based methods. We consider using Polynomial kernel with parameter $d = 2$.



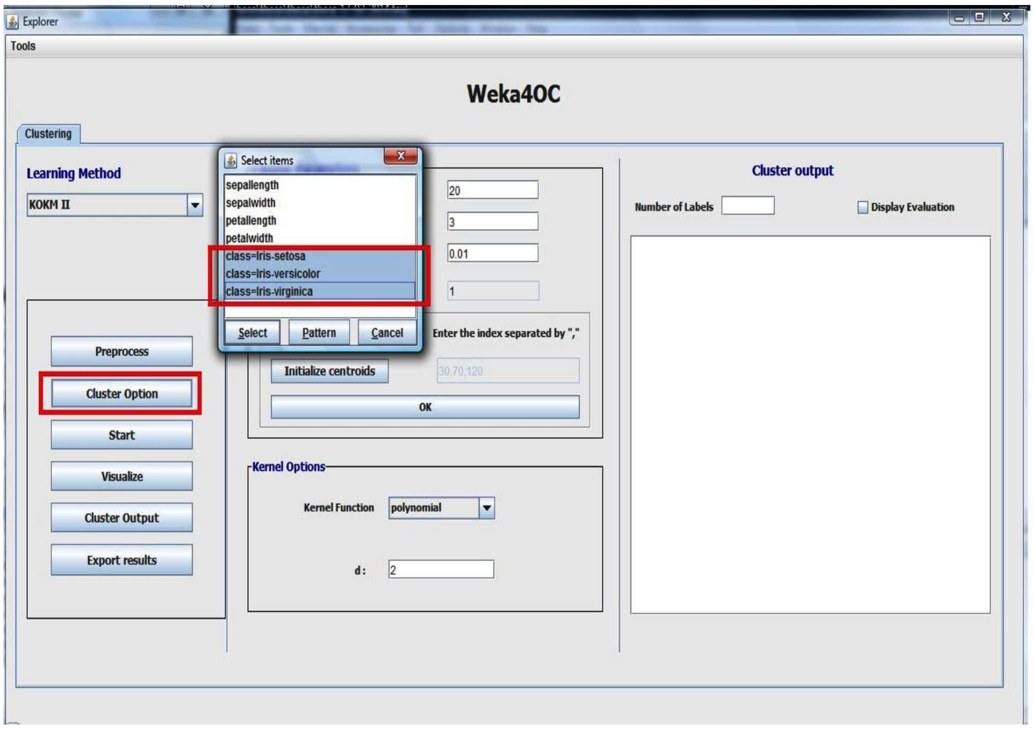
KOKM II

Cluster Parameters
max iterations : 20
num clusters : 3
minimal improvement : 0.01
Alpha : 1
Enter the index separated by ","
30,70,120

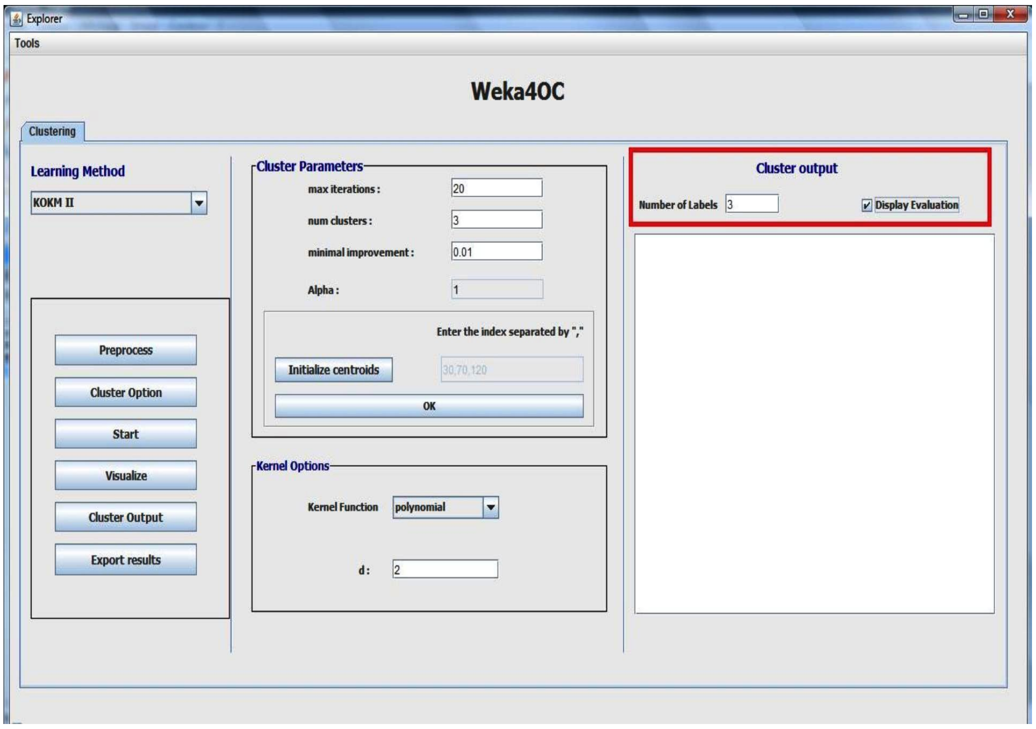
Kernel Options
Kernel Function polynomial
d : 2

Number of Labels

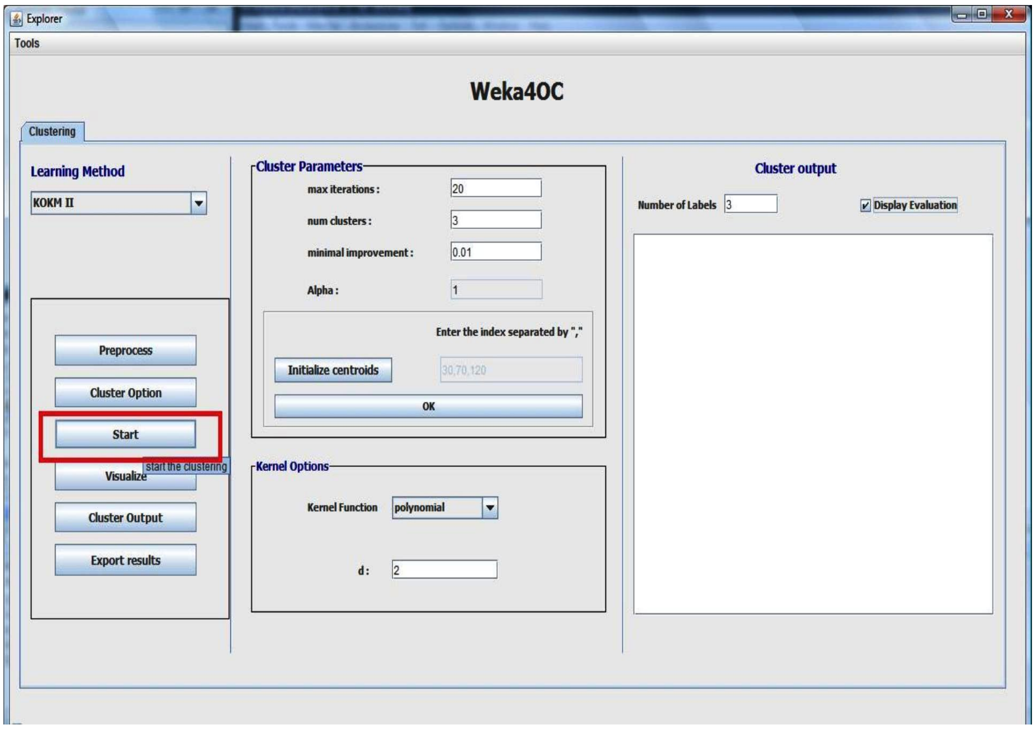
10) Ignore Class attributes while performing the learning



11) Configure outputs for displaying evaluation of the obtained overlapping clusters. Check "display evaluation" and set the number of Labels to 3.



12) Start the learning



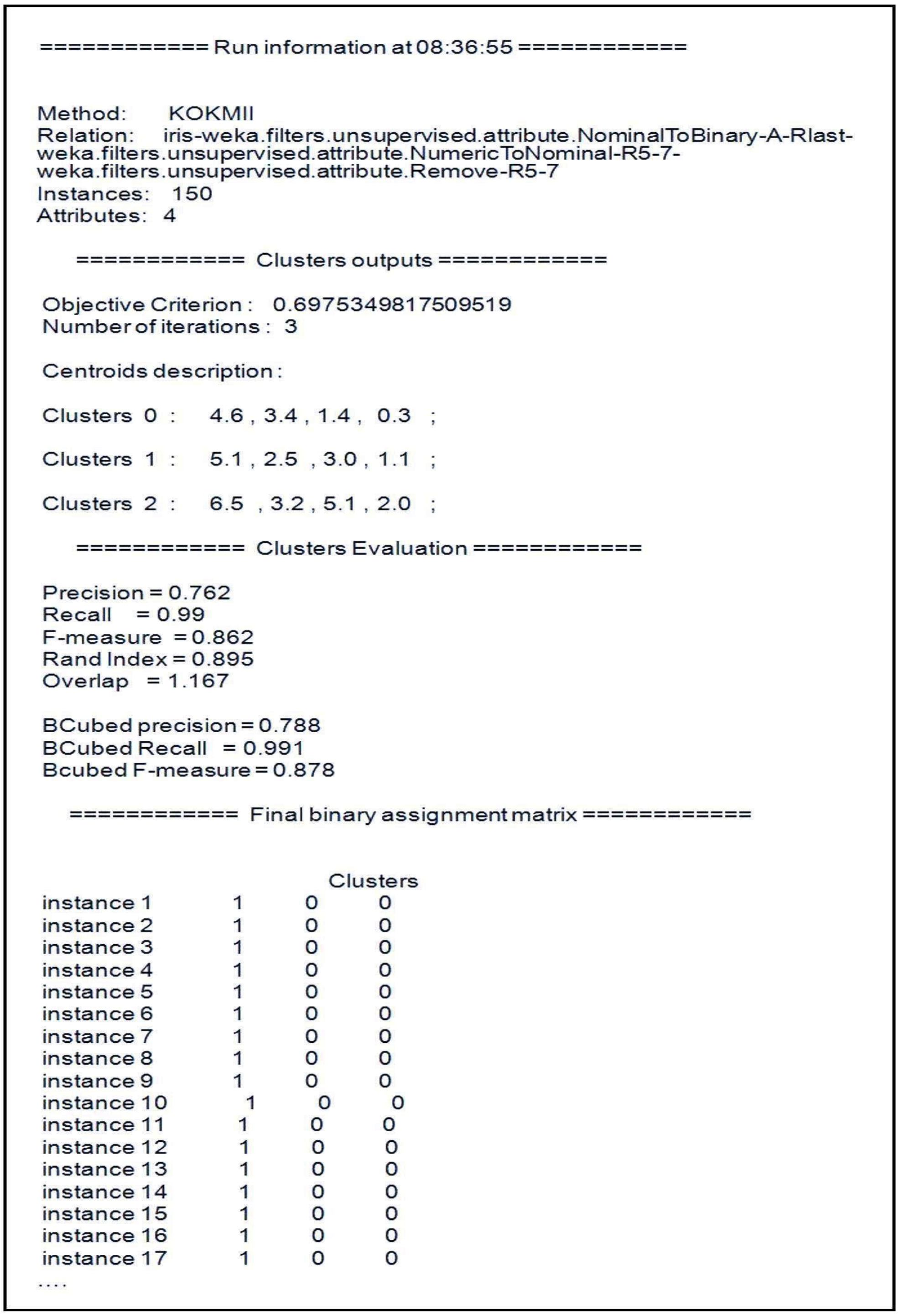
Learning Method: KOKM II

Cluster Parameters
max iterations : 20
num clusters : 3
minimal improvement : 0.01
Alpha : 1
Enter the index separated by ","
30,70,120

Kernel Options
Kernel Function: polynomial
d : 2

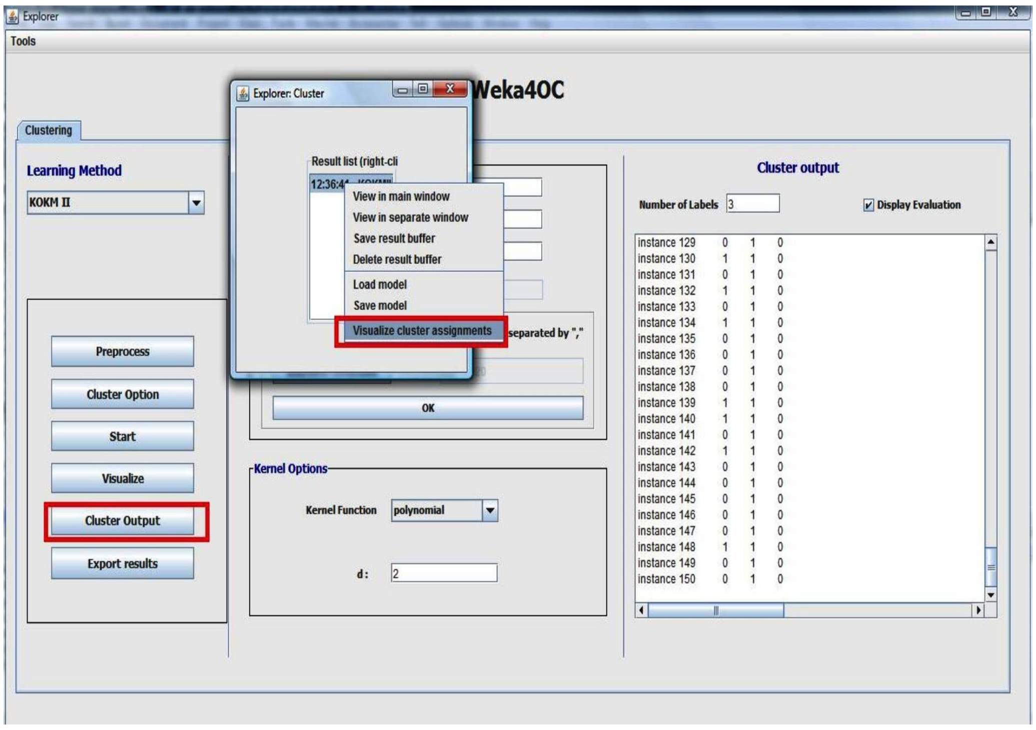
Number of Labels: 3
[✓] Display Evaluation

13) Outputs and evaluation of the obtained clusters



```
============ Run information at 08:36:55 ============


Method:     KOKMII
Relation:   iris-weka.filters.unsupervised.attribute.NominalToBinary-A-Rlast-
weka.filters.unsupervised.attribute.NumericToNominal-R5-7-
weka.filters.unsupervised.attribute.Remove-R5-7
Instances:   150
Attributes:  4

    ============  Clusters outputs ============

 Objective Criterion :   0.6975349817509519
 Number of iterations :  3

 Centroids description :

 Clusters  0  :     4.6 , 3.4 , 1.4 ,  0.3   ;

 Clusters  1  :     5.1 , 2.5  , 3.0 , 1.1   ;

 Clusters  2  :     6.5  , 3.2 , 5.1 , 2.0   ;

    ============  Clusters Evaluation ============

 Precision = 0.762
 Recall    = 0.99
 F-measure  = 0.862
 Rand Index = 0.895
 Overlap   = 1.167

 BCubed precision = 0.788
 BCubed Recall  = 0.991
 Bcubed F-measure = 0.878

    ============  Final binary assignment matrix ============


                           Clusters
 instance 1          1       0       0
 instance 2          1       0       0
 instance 3          1       0       0
 instance 4          1       0       0
 instance 5          1       0       0
 instance 6          1       0       0
 instance 7          1       0       0
 instance 8          1       0       0
 instance 9          1       0       0
 instance 10          1       0       0
 instance 11         1       0       0
 instance 12         1       0       0
 instance 13         1       0       0
 instance 14         1       0       0
 instance 15         1       0       0
 instance 16         1       0       0
 instance 17         1       0       0
....
```

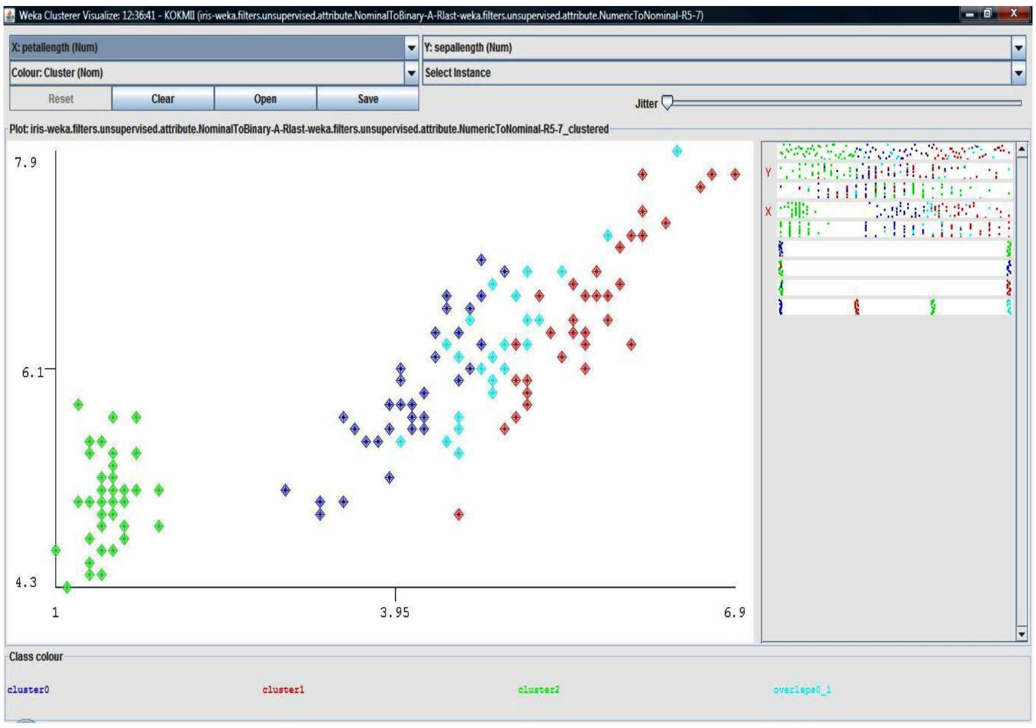
14) Visualize overlapping clusters



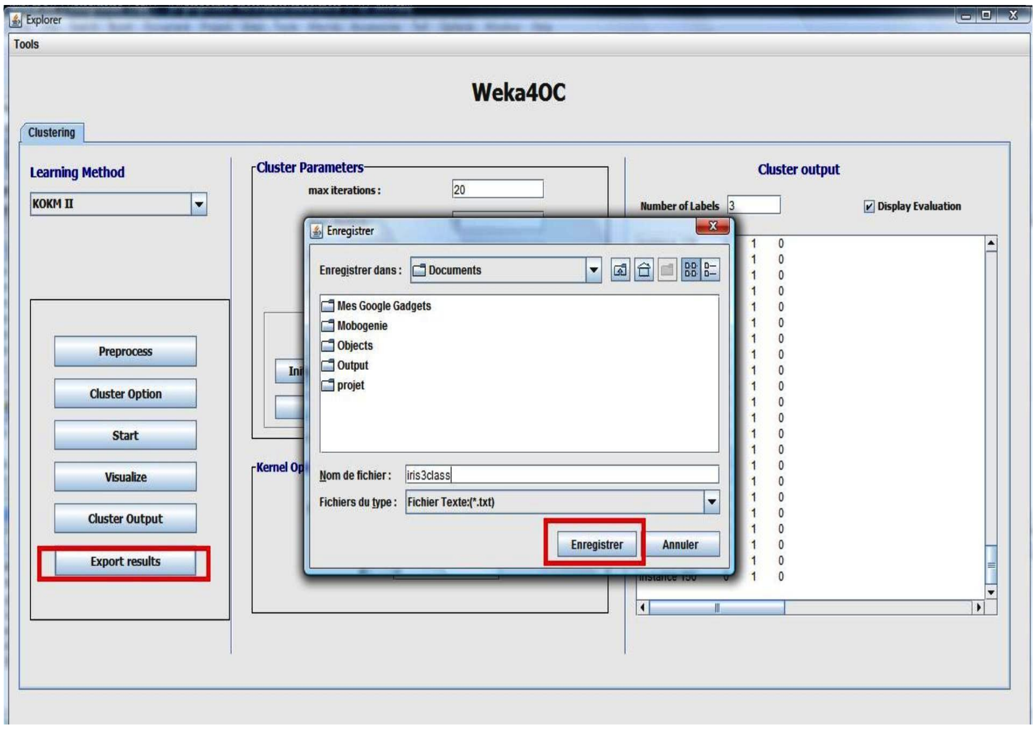
Number of Labels 3

```
instance 129   0   1   0
instance 130   1   1   0
instance 131   0   1   0
instance 132   1   1   0
instance 133   0   1   0
instance 134   1   1   0
instance 135   0   1   0
instance 136   0   1   0
instance 137   0   1   0
instance 138   0   1   0
instance 139   1   1   0
instance 140   1   1   0
instance 141   0   1   0
instance 142   1   1   0
instance 143   0   1   0
instance 144   0   1   0
instance 145   0   1   0
instance 146   0   1   0
instance 147   0   1   0
instance 148   1   1   0
instance 149   0   1   0
instance 150   0   1   0
```

15) Configure X and Y axes


Weka Clusterer Visualize: 12:36:41 - KOKMII (iris-weka.filters.unsupervised.attribute.NominalToBinary-A-Rlast-weka.filters.unsupervised.attribute.NumericToNominal-R5-7)
X: petallength (Num)
Y: sepallength (Num)
Colour: Cluster (Nom)
Select Instance
Reset
Clear
Open
Save
Jitter
Plot: iris-weka.filters.unsupervised.attribute.NominalToBinary-A-Rlast-weka.filters.unsupervised.attribute.NumericToNominal-R5-7_clustered
7.9
6.1
4.3
1
3.95
6.9
Y
X
Class colour
cluster0
cluster1
cluster2
overlaps0_1

16) Export results into a txt file.



max iterations : 20

Number of Labels 3

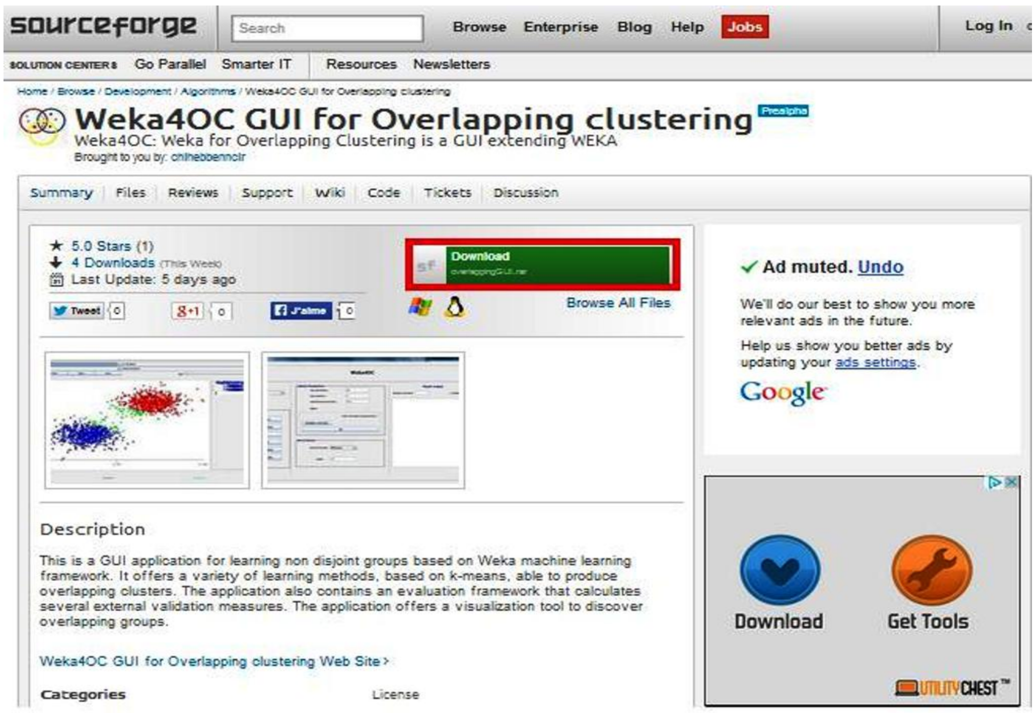
Enregistrer dans : Documents

Nom de fichier : iris3class

Fichiers du type : Fichier Texte:(*.txt)

B. Availability and installation of Weka4OC

Weka4OC is a software that is freely available under the GNU general public license agreement. The program information can be found by conducting a search on the Sourceforge web site for Weka4OC overlapping clustering or going directly to the site at “https://sourceforge.net/projects/overlappingclus”.



Home / Browse / Development / Algorithms / Weka4OC GUI for Overlapping clustering

Weka4OC GUI for Overlapping clustering Prealpha

Weka4OC: Weka for Overlapping Clustering is a GUI extending WEKA

Brought to you by: chihebbenncir

★ 5.0 Stars (1)
4 Downloads (This Week)
Last Update: 5 days ago

✓ Ad muted. Undo

We'll do our best to show you more relevant ads in the future.

Help us show you better ads by updating your ads settings.

Google

Description

This is a GUI application for learning non disjoint groups based on Weka machine learning framework. It offers a variety of learning methods, based on k-means, able to produce overlapping clusters. The application also contains an evaluation framework that calculates several external validation measures. The application offers a visualization tool to discover overlapping groups.

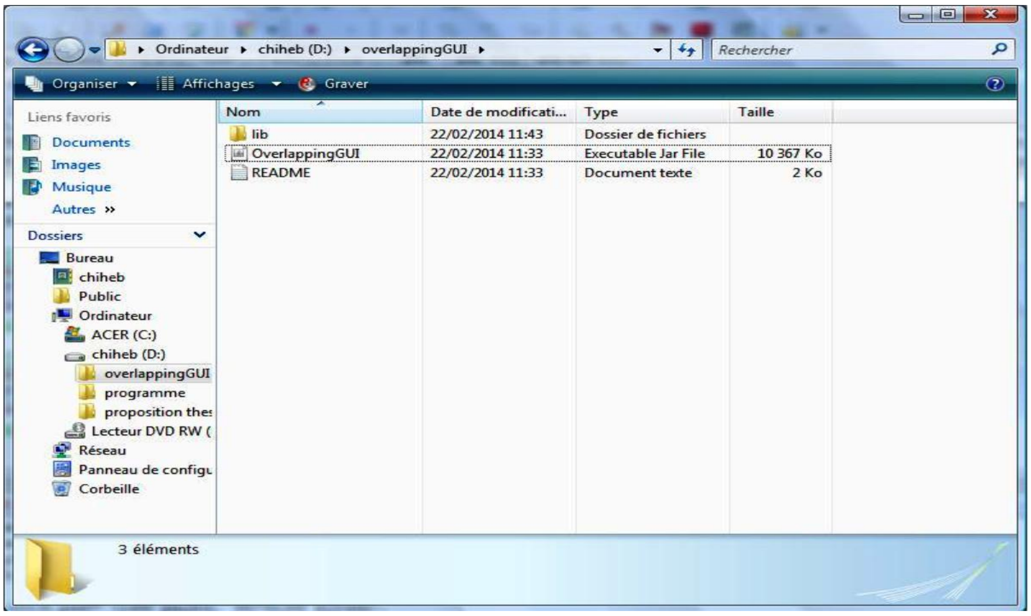
Weka4OC GUI for Overlapping clustering Web Site >

Categories

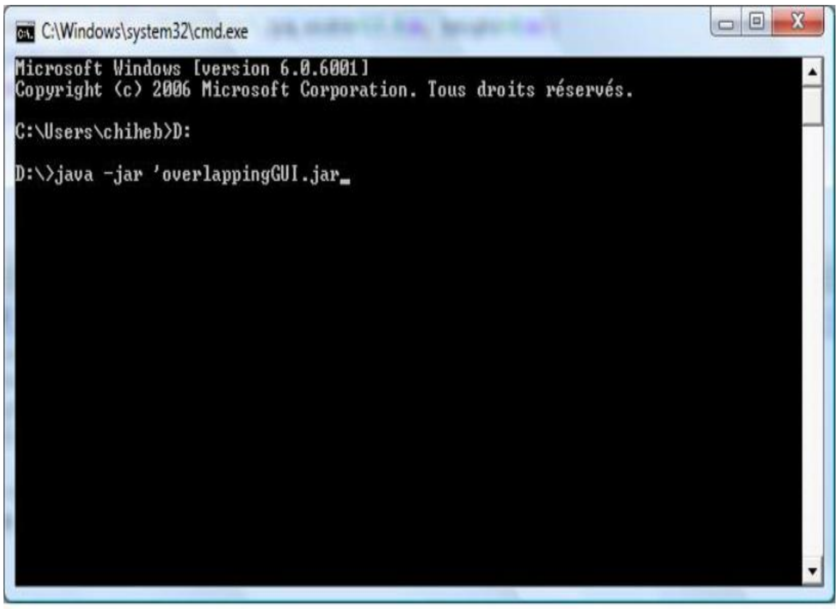
License

When prepared to download the software, it is best to select the latest application from the selection (if exist) offered on the site. The format for downloading the application is offered in zipped folder “OverlappingGUI.rar” that provides the complete program on the end users machine that is ready to use when unzipped.

The program can easily be run by double clicking on the file “OverlappingGUI.jar” and requires at least Java Runtime Environement 1.6. The program can be also run from the command line by typing: “java -jar ’OverlappingGUI.jar’ ”.



Nom	Date de modificati...	Type	Taille
lib	22/02/2014 11:43	Dossier de fichiers	
OverlappingGUI	22/02/2014 11:33	Executable Jar File	10 367 Ko
README	22/02/2014 11:33	Document texte	2 Ko



```
Microsoft Windows [version 6.0.6001]
Copyright (c) 2006 Microsoft Corporation. Tous droits réservés.

C:\Users\chiheb>D:

D:\>java -jar 'overlappingGUI.jar_
```