N° d'Ordre : D.U. ...
EDSPIC : ...

Université Paris 13

ÉCOLE DOCTORALE GALILÉE

Thèse

présentée par

Nistor GROZAVU

pour obtenir le grade de

DOCTEUR D'UNIVERSITÉ
Spécialité: INFORMATIQUE

Classification Topologique pondérée : approches modulaires, hybrides et collaboratives

Soutenue publiquement le 8 décembre 2009 devant le jury composé de :

Directeur de thèse :	
Younès BENNANI	Professeur Université Paris 13
Rapporteurs :	
Pascale KUNTZ	Professeur Ecole Polytechnique de Nantes
Michel VERLEYSEN	Professeur Université Catholique de Louvain
Examinateurs :	
Alexandre AUSSEM	Professeur Université Lyon 1
Mustapha LEBBAH	Maître de conférences Université Paris 13
Jean-François MARCOTORCHINO	Directeur scientifique THALES
Jean-Luc ZARADER	Professeur Université Paris 6
Jean-Daniel ZUCKER	Directeur de Recherche IRD

N° d'Ordre : D.U. ...
EDSPIC : ...

Paris 13 University

ÉCOLE DOCTORALE GALILÉE

Thesis

presented by

Nistor GROZAVU

for the degree of

DOCTOR OF COMPUTER SCIENCE

Weighted Topological Clustering: Modular, Hybrid and Collaborative approaches

Younès BENNANI	Professor Paris 13 University
Pascale KUNTZ	Professor Ecole Polytechnique de Nantes
Michel VERLEYSEN	Professor Université Catholique de Louvain
Alexandre AUSSEM	Professor Lyon 1 University
Mustapha LEBBAH	Associate Professor Paris 13 University
Jean-François MARCOTORCHINO	Scientific Director THALES
Jean-Luc ZARADER	Professor Paris 6 University
Jean-Daniel ZUCKER	Research Director IRD

Aknowledgement

First and foremost, I want to express my greatest gratitude to my thesis supervisor Prof. Younès Bennani who initiated me into the field of data mining. Under his guidance, I have learned lot in different aspects of conducting research, finding and solve a research problem and prioritizing different research task in data minig field. I am grateful to him contributions, not just to this dissertation, but also to making me a researcher and a better person.

I want to thank Prof. Pascale Kuntz and Prof. Michel Verlysen who have reported my thesis and for theirs constructive remarks. I thank also Prof. Alexandre Aussem, Dr. Mustapha Lebbah, Prof. Jean-François Marcotorchino, Prof. Jean-Luc Zarader and Prof. Jean-Daniel Zucker for accepted being in my thesis jury and for their interesting and relevant comments.

I would also like to thank the Computer Science Laboratory of Paris North University (LIPN), especially the Machine Learning and Applications Team (A3) for their advice and suggestions which have been very helpful.

I am also grateful to several other researchers and students who encouraged me during my work. Dr Mustapha Lebbah is a very nice person and his research and programmes were very helpful for my research. The interaction with Dr Lazhar Labiod led me to propose an hybrid clustering approach. The collaboration with the Data Mining research team from THALES S.A. during the Infom@gic project, especially with Hamid Benhadda, allowed me to file a patent with them presented in Chapter 3.

On a more personal side, I am grateful to all the new friends that I have made during the past few years : Nicoleta Rogovschi, Hanene Azzag, Guénaël Cabanes, Nasserine Benchettara, Sébastien Guérif, Victor Moraru, Maia Iordatii, Valentin Kostenko ... I want to show my appreciation to Dr. Rushed Kanawati with whom I shared my office for more than three years.

Let me reserve my final appreciation to my father Nistor, my mother Maria, and my brother Victor. Without the nurturing, care and love from them, I definitely could not have completed my doctoral degree.

Finally, this thesis is dedicated to the memory of my grandmother Anastasia and my grandfather Traian who left this world while I wrote my thesis.

This work was supported in part by the "Pôle de compétivité Cap Digital" within the Infom@gic project.

Nistor GROZAVU

Paris 13 University,

December 2009

To my parents, A mes parents,

To my brother, A mon frère.

Remerciement

Ce travail a été réalisé dans le cadre du projet Infom@gic du Pole de Compétitivité Cap Digital (Image, Multimedia and Vie numérique).

Notations

SOM - Self Organizing Map
vm-SOM - voting memory based Self Organizing Map
lwo-SOM - local weighting observation Self Organizing Map
gwo-SOM - global weighting observation Self Organizing Map
lwd-SOM - local weighting distance Self Organizing Map
gwd-SOM - global weighting distance Self Organizing Map
dlw-SOM - double local weighting Self Organizing Map
R - objective function; ex: R_{lwo} - objective function for local weighting observations
π - weights vector
$\mathbf{x}$ - samples vector
$\mathbf{w}$ - prototypes vector
$n \times m$ - dataset size (n - number of observations, m - number of variables);
$\Pi^{(d)}$ - Distance weights (in the case of the *dlw*-SOM)
$\Pi^{(o)}$ - Observations weights (in the case of the *dlw*-SOM)
$|\mathcal{W}|$ - ma size (the numberof cells)
N - the size of data (number of samples)
m - number of variables;
RA - Relational Analaysis approach;
RTM - Relational Topological Map;
R_{SOM}^{HCol} - Objective function of the Horizontal Collaborative SOM
R_{SOM}^{VCol} - Objective function of the Vertical Collaborative SOM

Résumé

Cette thèse est consacrée d'une part, à l'étude d'approches de caractérisation des classes découvertes pendant l'apprentissage non-supervisé, et d'autre part, à la classification non-supervisée modulaire, hybride et collaborative. L'étude se focalise essentiellement sur deux axes :

1) la caractérisation des classes en utilisant la pondération et la sélection des variables pertinentes, ainsi que l'utilisation de la notion de mémoire pendant le processus d'apprentissage topologique non-supervisé;

2) l'utilisation de plusieurs techniques de clustering en parallèle et en série : approches modulaires, hybrides et collaboratives.

Nous nous intéressons plus particulièrement dans cette thèse aux cartes auto-organisatrices de Kohonen qui constituent une technique bien adaptée à la classification non-supervisée permettant une visualisation des résultats sous forme d'une carte topologique. Nous proposons plusieurs techniques de pondérations de l'apprentissage de ces cartes ainsi qu'une nouvelle stratégie de compétition permettant de garder en mémoire l'historique de l'apprentissage. En utilisant un test statistique pour la sélection des variables pertinentes pondérées, nous répondons au problème de la réduction des dimensions, ainsi qu'au problème de la caractérisation des classes découvertes.

Concernant le deuxième axe, nous utilisons le formalisme mathématique de l'analyse relationnelle (AR) pour combiner plusieurs résultats de classification. Les avantages de l'AR sont combinés avec ceux des cartes auto-organisatrices pour proposer un nouvel algorithme RTM (Relational Topological Map) alliant les points forts des deux approches.

Enfin, nous proposons une nouvelle approche conçue pour faire collaborer plusieurs classifications topologiques entre elles, en préservant la confidentialité des données. Cette approche est basée sur un formalisme d'apprentissage collaboratif qui dérive ses règles d'adaptations

à partir d'une nouvelle fonction de coût composée de deux termes : un terme de quantification et un autre de collaboration. Ce formalisme est proposé pour deux stratégies : collaboration verticale et horizontale.

Mots-Cles:
apprentissage non-supervisé, cartes auto-organisatrices, apprentissage avec mémoire, pondération, sélection des variables, fusion de partitions, apprentissage collaboratif, analyse relationnelle, clustering hybride, clustering modulaire.

Abstract

This thesis is focused, on the one hand, on clustering anlaysis approaches in the unsupervised topological learning, and on the other hand, to the topological modular, hybrid and collaborative clustering. This study is mainly adressed to two problems:

1) cluster characterization using weighting and selection of relevant features, and the use of the memory concept during the unsupervised topological learning process;

2) and the problem of the ensemble clustering techniques: modular, the hybrid and collaborative approaches.

In this thesis we are particularly interested in Kohonen's self-organizing maps which have been widely used for unsupervised classification and visualization of multidimensional datasets. We offer several weighting approaches and a new strategy which consists in introducing of a memory process into the competition phase by calculating a voting matrix at each learning iteration. Using a statistical test for selecting relevant features, we will respond to the problem of dimensionality reduction, and to the problem of clusters characterization.

For the second problem, we use the relational analysis approach (RA) to combine multiple topological clustering results. The benefits of RA will be combined with those of self-organizing maps in order to obtain a new algorithm called RTM (Relational Topological Map). Finally, we adapt the classical algorithm of self-organizing maps to enable the collaboration between them.

Keywords:
clustering, unsupervised learning, self-organizing maps, weighting, variable selection, fusion, collaboration, relational analysis, hybrid clustering, modular clustering, memory based topological clustering.

Contents

List of Figures

List of Tables

Avant Propos

Rapide parcours de la thèse

Le travail de recherche présenté dans cette thèse est structuré autour de quatre chapitres. La thèse commence par une introduction qui présente le contexte de l'étude ainsi que les objectifs de la thèse, et se termine par le plan du manuscrit.

Le premier chapitre concerne la "**formulation du problème et la présentation des objectifs**". Il présente le contexte de l'étude de la réduction des dimensions dans des grandes masses de données, en expliquant en détail différents types d'algorithmes de classification non supervisée et de la sélection de variables, avec un accent particulier sur la pondération pendant le processus de classification/apprentissage. L'importance de ces approches dans la caractérisation des classes est mise en évidence.
L'objectif principal de la réduction des dimensions est de réduire le nombre des observations (le clustering) ou la taille des variables (la sélection ou l'extraction). Nous établissons alors une typologie des principales méthodes de réduction de dimensions. Dans ce chapitre nous nous intéressons aux méthodes permettant de réduire le nombre d'observations tout en facilitant la description des classes découvertes. La caractérisation sera faite dans le cadre de ce travail à travers un processus de pondération non supervisée des variables. Cette pondération consiste à chercher des poids numériques pour les associer à chaque variable durant le processus d'apprentissage. Dans le cadre de la classification non- supervisée, les travaux présentés dans ce chapitre montrent que:

- la pondération des variables permet d'obtenir de meilleurs résultats de classification;
- une pondération d'attributs indépendamment de la méthode de classification produit de moins bons résultats que les approches intégrées.
- la pondération locale des variables est plus efficace qu'une pondération globale [52, 19];

- les pondérations adaptatives sont plus efficaces que les pondérations analytiques.

Le chapitre se termine par une étude de la complexité de chaque algorithme, ainsi que les différentes méthodes de validation du clustering.

Le **chapitre 2** s'intitule "**Classification topologique pondérée avec mémoire**".

Dans ce chapitre, nous introduisons une nouvelle stratégie d'apprentissage pour les algorithmes de classification topologique basés sur le modèle de Kohonen. Nous introduisons deux notions dans ce modèle : l'apprentissage avec mémoire et l'apprentissage pondéré. En effet, dans ces algorithmes, le processus d'auto-organisation permet de concentrer l'adaptation des poids des connexions essentiellement sur la région de la carte la plus "active". Cette région d'activité est choisie comme étant le voisinage associé au neurone dont l'état est le plus actif. Durant la phase de compétition, le critère de sélection de l'unité la plus active est de chercher celle dont le vecteur de poids est le plus proche au sens de la distance euclidienne de la forme présentée. Il s'agit d'un critère qui est à l'heure actuelle utilisé dans les différentes variantes de l'algorithme des cartes topologiques. Notre contribution se situe au niveau de cette phase de compétition, et propose une nouvelle stratégie pour le choix du neurone le plus actif. Cette nouvelle stratégie consiste à choisir le neurone le plus actif en tenant compte de son historique d'activations appris dans une matrice de vote à partir de l'ensemble des données. Cette approche renforce les interactions entre deux notions : la mémoire et l'apprentissage, si intimement liés qu'on confond souvent les deux. Ces deux notions renvoient cependant à des phénomènes différents. L'apprentissage désigne un processus qui va modifier un comportement ultérieur. La mémoire est la capacité de se rappeler des expériences passées. De plus, non seulement la mémoire dépend de l'apprentissage, mais l'apprentissage dépend aussi de la mémoire. En effet, les connaissances mémorisées constituent une trame sur laquelle viennent se greffer les nouvelles connaissances. Notre approche introduit donc dans le processus de l'apprentissage, au niveau de la compétition, un effet mémoire des activations des neurones. En effet, c'est l'utilisation de cette notion de voisinage qui introduit les contraintes topologiques dans la géométrie finale de la carte. Cette nouvelle approche permet de découvrir la structure des données avec une meilleure qualité (erreur topologique plus faible et pureté de la classification plus importante).

La notion d'apprentissage pondéré est présenté dans ce chapitre à travers une nouvelle approche de classification et de pondération des variables durant un processus non supervisé. Ainsi, l'apprentissage des cartes topologiques est combiné à un mécanisme d'estimation de pertinences des différentes variables sous forme de poids d'influence sur la qualité de la classification non supervisée. Nous proposons plusieurs types de pondérations adaptatives :

- une pondération des observations (locale et globale)

- une pondération des distances entre observations (locale et globale)
- une approche de double pondération.

L'apprentissage simultané des pondérations et des prototypes utilisés pour la partition des observations permet d'obtenir une classification optimisée des données. Un test statistique (ScreeTest) est ensuite réalisé sur ces pondérations pour élaguer les variables non pertinentes. Ce processus de sélection de variables permet enfin, grâce à la localité des pondérations, d'exhiber un sous ensemble de variables propre à chaque groupe (cluster) offrant ainsi sa caractérisation. Pour chaque approche proposé dans ce chapitre la complexité est également calculée, ce qui montrent que les algorithmes proposées augmentent très peu le temps de calcul (la complexité reste linéaire).

Nous expliquons à travers différents exemples l'intérêt de l'estimation des pertinences des variables pour la visualisation et la sélection des variables, ainsi que pour la caractérisation des groupes. Nous montrons aussi que notre approche peut être considérée comme une pseudo-classification croisée. Enfin, contrairement à la classification croisée qui permet aussi de caractériser les groupes visuellement, la méthode proposée dans ce chapitre permet de les caractériser d'une manière automatique. L'estimation du nombre correct de groupes est en relation avec la stabilité de la segmentation et la validité des groupes générés.

Le chapitre 3 s'intitule "**Classification modulaire et Hybride**". Dans la première partie de ce chapitre, une technique de fusion de clustering est proposée, en se basant sur les algorithmes de pondération proposées (g/lwo-SOM, g/ldw-SOM) et l'analyse relationnelle pour fusionner les résultats de ces dernières.

Ainsi, nous proposons une représentation du consensus de clustering comme un ensemble de nouvelles variables qui caractérisent les observations. Ceci conduit à une formulation du problème de fusion sous forme d'un problème de clustering catégorique. Nous proposons d'utiliser *l'Analyse Relationnelle* (RA) comme méthode du consensus pour plusieurs méthodes d'apprentissage non-supervisé. Le consensus de clustering est fourni en tant que solution de la minimisation de la fonction objective d'une donnée consensus du clustering. L'idée principale, partagée avec d'autres algorithmes est que : si de nombreux algorithmes de classification assignent deux observations dans le même cluster (groupe), ça ne sera pas bénéfique de séparer ces observations par le consensus de clustering. Les résultats expérimentaux montrent les avantages et les intérêts de cette nouvelle approche de fusion. Ainsi que l'application sur les donnés wikipedia composées des images montre une réel validation de cette technique.

Dans la deuxième partie, nous proposons un nouveau modèle pour le clustering et la visualisation des données binaires basé sur la notion de voisinage présente dans SOM classique,

mais en utilisant le formalisme de l'analyse relationnelle. Cette approche hybride RTM (Relational topological map) combine les avantages des deux méthodes. En effet, elle permet une identification naturelle du clustering sans fixer le nombre de clusters (ou neurones) et incorpore la notion de voisinage durant l'apprentissage. Un des avantages de la nouvelle méthode hybride RTM est de permettre de construire des cartes topologiques à partir de données binaires en gardant une faible complexité algorithmique. En plus, la comparaison avec des méthodes existantes, montre que l'approche RTM permet de construire des cartes plus structurées et fines.

Le chapitre 4, "**Classification topologique collaborative**", propose une nouvelle approche conçue pour faire collaborer plusieurs cartes SOM (ou variantes de SOM) entre elles en préservant la confidentialité des données. Ayant une collection de bases de données distribuées sur plusieurs sites différents, le problème consiste à partitionner chacune de ces bases en considérant les données des autres bases collaboratrices, sans toutefois omettre de respecter la contrainte de confidentialité qui interdit le partage de données entre les differents sites. Pour ce faire, notre approche se subdivise en deux phases : une phase locale et une phase de collaboration. La phase locale revient à appliquer l'algorithme classique de SOM, localement et independamment sur chacune des bases de données, ce qui se soldera par l'obtention d'une carte SOM pour chacune de ces bases. La phase de collaboration revient à faire collaborer chacune des bases de données avec toutes les cartes SOM associées aux autres bases obtenues lors de la phase locale. Ainsi, comme resultat on obtient sur chacun des sites une carte SOM proche de la carte SOM qu'on aurait obtenu si on avait fait abstraction de la contrainte de confidentialité, à savoir faire collaborer les bases de données elles mêmes. A l'issu des deux phases, toutes les cartes SOM sont enrichies. Ce chapitre présente le formalisme mathématique de l'approche selon deux stratégies (collaboration verticale et horizontale) ainsi que sa validation expérimentale.

Comme perspectives, nous continuons à explorer les trois axes de recherche étudiés dans cette thèse :

- la caractérisation des classes en passant par la pondération et la sélection des variables pertinentes;
- la fusion des classifications, ainsi que le clustering topologique hybride en utilisant l'analyse relationnelle;
- la classification collaborative topologique non-supervisée.

Concernant le première axe, nous pensons intégrer la capacité de caractériser les clusteurs dans le cadre de la fusion et de la collaboration des cartes : avant et après la collaboration. Ainsi, nous envisageons d'adapter ces approches a des données binaires et mixtes.

Pour le clustering topologique hybride, les perspectives consistent à introduire des pondérations pendant l'apprentissage de RTM, pour pouvoir ensuite détecter les variables (binaires) pertinentes et caractériser ainsi les classes obtenues.

Concernant la classification topologique collaborative, nous envisageons de combiner la collaboration verticale et horizontale pour obtenir la collaboration hybride entre les cartes auto-organisatrices. Un autre axe de recherche qui dérive de ces travaux, est le clustering topologique directionnel qui va nous permettre de savoir dans quelle direction faire collaborer les cartes ou les bases de données.

Finalement, comme les algorithmes proposés dans cette thèse nécessitent des paramètres à fixer (la taille de la carte, le pas d'apprentissage, le nombre d'itérations, le paramètre β) nous pensons à long terme de les rendre autonomes.

Introduction

The Problem and the Approach

The subject of the thesis is concentrated around two main problems of the unsupervised learning. Firstly, the problem of dimensionality reduction and cluster characterization is stressed, and a consistent determination of the possible approaches to both of them is given. Secondly, the aspects of the modular, hybrid and collaborative topological clustering are studied.

Dimensionality reduction is a major challenge in the domain of unsupervised learning which deals with the transformation of a high dimensional dataset into a low dimensional space, while retaining most of the useful structure in the original data, retaining only relevant features and observations.
Another important aspect of the dimensionality reduction is the visualization of the results, and the self-organizing maps is one of the most known topological learning techniques which allows clustering and visualization simultaneously.
At the end of the topographic learning, the "similar" data will be collected in clusters, which correspond to the sets of similar observations. These clusters can be represented by a more concise information than the brutal listing of their patterns, such as their gravity center or different statistical moments. As expected, this information is easier to manipulate than the original data points.
The choice of the activity area during self-organization does not take into account the competition history of each unit during learning. This is a criterion, which is currently used in several variations of the topological maps algorithm.

It is well known that no clustering method is perfect because many of clustering algorithms require additional user-specified parameters, such us the optimal number of clusters (k-means), stopping criteria, similarity parameters, smoothing parameter (β in *lwd*-SOM or *gwd*-SOM), and others parameters. There are also, some algorithms which use random initializations and due to this they give different results for each replication, so there is no clear

indication for the best partition result. So, it is clear, that a combination scheme between clustering algorithms could give a better result: number of clusters, purity of segmentation, etc. To resolve this challenge, some approaches for Multi-Clustering Fusion Scheme were proposed in literature as [47, 123, 109]: “Consensus Models”, “LSEC - Least Square Error Combination”, “HCE - Heterogeneous clustering ensemble”, etc.

In an industrial context of increasing competition, companies are constantly called upon to work together (to collaborate) to face up to this strategic issue and to be able to provide the level of required service for their customers. To benefit from this collaboration, the tasks of Data Mining (Clustering, mining and knowledge management ...) should consider all the datasets associated with these collaborating companies although they are distributed on several different sites. Obviously, for confidentiality reasons (ex. medical or bank data), sharing data between collaborating companies is not allowed. So, centralizing their data by combining them into one dataset and then performing the task of Data Mining is not appropriate.

For these two research directions, we have proposed some approaches for the cluster characterization problem (Chapter 2), an original clustering fusion technique using Relational Analysis (RA) and a new topological clustering method based on RA (Chapter 3). We proposed also, a vertical and horizontal collaboration for the topological clustering based on self-organizing maps (chapter 4).
The validation of these proposed techniques is performed on several academical and real datasets (Appendix A.1).

Guide to the Dissertation

The dissertation is structured in four chapters.
chapter 1 deals with the **Data Clustering Analysis** which outlines the background of the dimensionality reduction problem for large datasets, explaining in detail different types of clustering and features selection algorithms, with particular emphasis on weighting during the learning/clustering process.
In this chapter we will try to cover some aspects of the clustering problem, to present the main categories of existed clustering algorithms, to introduce the problem of the clustering analysis through dimensionality reduction and weighting techniques. The importance of these approaches for the cluster characterization is highlighted.
The main goal of the dimensionality reduction is to reduce the number of observations (clustering) or the size of variables (selection or extraction). Then we establish a typology of

significant dimensionality reduction methods. In this chapter we focus on the observations reduction methods while improving cluster characterization, and the cluster characterization is done through an unsupervised variables weighting process.
This weighting process seeks to find numerical weights for each variable involved in the learning process. In the context of unsupervised learning and clustering, the work presented in this chapter show that:

- variables weighting allows better clustering results;
- variables weighting abstracted from the clustering method produces poorer results than integrated approaches.
- local variables weighting is more efficient than global weighting [52, 19];
- adaptive weights are more effective than analytic weights.

The chapter ends with a study of the complexity of each algorithm, as well as different clustering validation methods.

Chapter 2, "**Memory based Weighted Topological Clustering**" presents new methods for the topological learning keeping the history during the learning/clustering process and introducing weighting approaches to attempt clustering characterization.
To preserve the memory during the SOM, our contribution relies on the competitive phase of this unsupervised learning algorithm and proposes a new strategy for choosing the most active cell/neuron. This new strategy is to choose the most active neuron taking into account its historical activations, learned in a voting matrix from the dataset. Indeed, the use of this historic neighborhood, allows introducing of topological constraints in the final geometry of the map. This new unsupervised learning approach (*vm*-SOM) allows discovering of data structure with a better quality (lower topographic error and better clustering purity). The conservation of the historical learning allows us to attempts a good topology of the map, but this technique doesn't take into consideration the relevance of the features or the observations. This is why we proposed an analytical local weighting distance SOM (*lwd*-SOM) and five adaptive weighting approaches for the SOM maps: local weighting observations SOM (*lwo*-SOM), global weighting observations SOM (*gwo*-SOM), local weighting distance SOM (*lwd*-SOM), global weighting distance (*gwd*-SOM) and finally the double local weighting SOM (*dlw*-SOM).

The main idea of the **Chapter 3**, **"Modular and Hybrid Clustering"**, is that combining multiple classification or regression models typically provides superior results compared to using a single model. A modular clustering represents a cluster ensemble approach which

can combine individual results from the distributed computing entities, under two scenarios [121, 108]:

- Object-distributed data. Each clustering algorithm has access to only a subset of all objects, and thus can only cluster the observed objects. For example, corporations tend to split their customers regionally for more efficient management.
- Feature-distributed data. In this scenario, each clustering algorithm sees only a limited number of features or attributes of each object, i.e. it observes a particular aspect or view of the data.

The main work in this chapter is situated around the Relational Analysis technique in order to attempt a clustering fusion and a topological unsupervised learning. Our contribution is to propose a new effective schema to deal with the clustering fusion problem by using proposed weighted SOM algorithms (g/lwo-SOM, g/lwd-SOM) to cluster the data, and the Relational Analysis technique to fusion the obtained clustering results.

As an application, for the proposed clustering fusion schema, an image retrieval system is presented using topological learning and a new interactive learning using user information/response. Also, in this work, we show the usage of unsupervised learning for images, we do not possess labels, and we will not take into account the corresponding text for the images. The used dataset contains 17812 images extracted from wikipedia pages, each of which is described by it colors and texture.

Finally, this chapter introduce a hybrid formalization of self-organizing maps in terms of Relational Analysis. This formalism is dedicated to data in the form of a binary matrix or a sum of binary matrices. This model is based on the Kohonen model (conservation of topological order) and uses the formalism of the Relational Analysis optimizing a criterion defined from the Condorcet criterion taking into account common similarities and dissimilarities for each pair of objects.

Finally, **Chapter 4 "Collaborative Clustering"** tackles the problem of collaborative clustering and presents an approach of collaborative clustering in order to preserve the confidentiality of data using self-organizing maps. Having a collection of datasets distributed on several different sites, the problem is to partition each of these data, considering other collaborating datasets, by respecting the confidentiality constraints that prohibit sharing of data between different sites. To do this, our approach is divided into two steps: a local step and a collaboration step. The local step would apply the standard algorithm of Kohonen, locally and independently to each dataset, which will result in obtaining a SOM map for each of these datasets. The collaboration phase will consist in collaborating each dataset with all

SOM maps associated with other datasets obtained during the local step. Thus, as a result is obtained a SOM map on each site which is close to the SOM map that could be obtained if we had ignored the constraint of confidentiality. At the end of both phases, all SOM maps will be enriched. This chapter presents the mathematical formalism of the approach and its validation. The proposed approach has been validated on multiple databases and experimental results have shown very promising performance.

Chapter 1

Data Clustering Analysis

1.1 Introduction

Analogous and complementary way for the development of supervised learning techniques, the so-called non-supervised techniques has been developed. The aim is to learn some information about non-labeled data. There are many cases where is no information about the clusters of all learning data. This lack of knowledge can have several causes: deficiency of information on the data, too large dataset to be manually labeled, like in speech recognition as well as in bioinformatics.
Different areas are interested in this problem with different names: statistics, data analysis, pattern recognition, psychology, etc. The term "unsupervised learning" is used in connectionism to describe these techniques as "clustering", "reunification", "classification by distance", "quantification", "aggregation", "partitioning"...

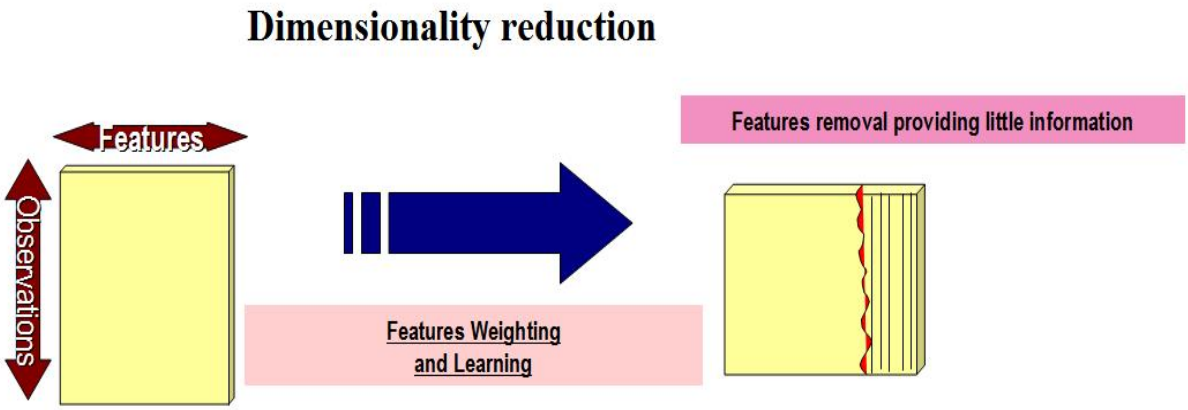


Figure 1.1: The principle of the dimensionality reduction

One of the important challenge in clustering is the dimensionality reduction which deals with the transformation of a high dimensional dataset into a low dimensional space, while retaining most of the useful structure in the original data; retaining only relevant features and observations. The data objects became to have a large number of features and its became to be more complicated to analyze its (like multimedia data analysis and bioinformatics). These are extreme examples of situations where dimension reduction (DR) is necessary because the number of variables p exceeds the number of examples n and it's called by staticians like "Big p Small n" [121].
Nevertheless, traditional algorithms used in machine learning and pattern recognition applications are often susceptible to the well-known problem of the *curse of dimensionality*.

Dimension reduction techniques are often applied as a data pre-processing step or as part of the data analysis to simplify the data model. Dimensionality reduction can be achieved by using a clustering technique to reduce the number of observations or a features selection approach to reduce the features space as is shown in the figure 1.1.

One of the most used techniques among many others in the data mining field is the clustering. The aim of this technique is to synthesize and summarize huge amounts of data by splitting it into small and homogeneous clusters such that the data (observations) inside the same cluster are more similar to each other than to the observations inside the other clusters. This definition assumes that there exists a well defined clustering quality measure that quantifies how much homogeneous are the obtained clusters [35].
As a fundamental pattern recognition problem, a well-designed clustering algorithm usually involves the following four design phases: data representation, modeling, optimization, and validation [47]. The data representation phase predetermines what kind of cluster structures can be discovered in the data. Generally, clustering problems can be divided into two categories: hard clustering and fuzzy clustering. In hard clustering, a data point belongs to one and only one cluster, while in fuzzy clustering, a data point may belong to two or more clusters with some probabilities.
Since clustering is an unsupervised process and most of the clustering algorithms are very sensitive to their initial assumptions, some sort of evaluation is required to assess the clustering results in most of the applications. Validity indices are measures that are used to evaluate and asses the results of a clustering algorithm.

In principle, the more information we have about each samples(pattern), the better a learning algorithm is expected to perform. This seems to suggest that we should use as many features as possible. But, this is not the case in practice, because some variables could be less important than others or even to represent noise in the respective dataset. The advantages or motivation for dimension reduction are the follows:

- The identification of relevant features is very useful for knowledge data mining.
- For many learning algorithms, the training and/or classification time increases with the number of features.
- It's very useful to detect noise or irelevant features in order to do a good learning/classification.
- The visualization of the data or of the clustering result is clearly in a low dimension.

The main drawback of dimensionality reduction is the possibility of information loss by eliminating some relevant features, but using some weighting techniques we can reduce this risk (see the proposed weighted approaches : g/lwo-SOM, g/lwd-SOM and dlw-SOM).

In this chapter we will try to cover some aspects of the clustering problem, to present the main categories of existed clustering algorithms, to introduce the problem of the clustering analysis trough dimensionality reduction and weighting techniques. In the next, we'll give more details inclusively the algorithms for the approaches which are more closely to our subject.

1.2 Data pre-processing

"*The capacity of digital data storage worldwide has doubled every nine months for at least a decade, at twice the rate predicted by Moore's Law for the growth of computing power during the same period.*" [1]
(Fayyad et al., 2002)

The real data don't have a standart format, it can be in different formats (numerical, binary, mixed, multimodal, etc.) with different values and different sizes. Usually the clustering algorithms are not adapted for all the types of dataset and because clustering is an unsupervised machine learning approach, the algorithms are not able to identify if the data are unbalanced or to decide which type of standardisation to use for the data and when. That's why in the machine learning domain there were introduced some pre-processing research directions like data discretization, balancing, data normalization and transformation, etc.

1.2.1 Balancing

One of the problems in pre-processing is that the real datasets don't always have a balanced cluster distribution, but can have clusters of different sizes (from a cluster which contains some observations to clusters with thousands of observations) and, in this case the clustering problem becomes difficult due to unbalance between the clusters, and usually algorithms can't obtain a good accuracy of clustering results [11], [121]. The are two types of Balancing problem:

- Sample Balancing. Each cluster should contain roughly the same number of samples and, if it is not the case, the clustering algorithms must take into account this problem by involving some additional techniques.
- Variables Balancing. Each cluster should contain roughly the same amount of feature values.

Strelh et al. [121] proposed to assign a weight to each object and then softly constrain the sum of weights in each cluster. For sample balanced clustering, it is proposed to assign to each sample $\mathbf{x}_j$ the same weight $\pi_j = 1/n$. To obtain value balancing properties, a sample $\mathbf{x}_j$'s weight is set to $\pi_j = \frac{1}{v}\sum_{i=1}^{m} x_{i,j}$ for $v = \sum_{j=1}^{n}\sum_{i=1}^{m} x_{i,j}$ and the sum of all weights is 1.

1.2.2 Data Normalization or Transformation

Preparing data for clustering requires some transformation, such as normalization or standardization. Data standardization makes data dimension less which is used to define standart indices.

In literature there were proposed many transformation and normalization methods [45], [70], [96]:

- Variance normalization represents a linear transformation which scales the values so that their variance is equal to 1. This is a convenient way to use Mahalanobis distance measure without changing the distance calculation procedure.
- Normalization of range of values which provides a linear transformation which scales the values between [0, 1]. This is one of the most used normalizations for data clustering.
- Similar to the previous one is the logistic normalization which ensures that all values in the future will have the range [0, 1].
- Logarithmic normalization is a technique that provide a exponentially distribution of the values of a vector component.
- Discrete histogram equalization. This algorithm orders the values and replaces each value by its ordinal number. Finally, it scales the values so that they are between [0,1]. This one is useful for both discrete and continuous variables, but as the saved normalization information consists of all unique values of the initialization dataset, it may use considerable amounts of memory.

1.3 Numerical and Categorical Data Measures

The first phase of any clustering algorithm is to compute a distance matrix or a dissimilarity matrix, which is a matrix size $n \times n$, where n is the number of samples of the dataset. The choice of the distance is very important for the clustering method, and this choice is made by taking into account the algorithm and the data type [33].

The table 1.1 shows the most known and used distance measures.

Table 1.1: Distance measures

Distance	Formula
Euclidian	$\sum_{i=1}^{m} \sqrt{(x_i - y_i)^2}$
Square Euclidean	$d_{sE}(x,y) = d_E(x,y)^2 = \sum_{j=1}^{m}(x_j - y_j)^2.$
Manhattan	$\sum_{i=1}^{m} \lvert x_i - y_i \rvert$
Maximum	$max\lvert x_i - y_i \rvert$
Average	$(\frac{1}{m}\sum_{j=1}^{m}(x_j - y_j)^2)^{\frac{1}{2}}$
Minkowski	$\sqrt[\lambda]{\sum_{i=1}^{m} \lvert x_i - y_i \rvert \lambda}$
Mahalanobis	$\sqrt{(x-y)S^{-1}(x-y)^T}$
Canberra	$\sum_{i=1}^{m} \frac{\lvert x_i - y_i \rvert}{(\lvert x_i \rvert + \lvert y_i \rvert)}$

In this work we use the Euclidian distance which is the most adapted for our algorithms and for the numerical data.
Between two data points x and y the Euclidian distance is defined as following:

$$d_E(x,y) = \left[\sum_{i=1}^{m}(x_i - y_i)^2)\right]^{\frac{1}{2}} \tag{1.1}$$

where x_i and y_i are the ith variable value for x and y respectively.

In the learning/clustering approaches the most used is the squared Euclidian distance (d_{sE})

Table 1.2: Similarity measures for binary vectors

Measure	Formula for $s(x, y)$
Jaccard	$\frac{S_{11}(x,y)}{S_{11}(x,y)+S_{01}(x,y)+S_{10}(x,y)}$
Dice	$\frac{S_{11}(x,y)}{2S_{11}(x,y)+S_{01}(x,y)+S_{10}(x,y)}$
Pearson	$\frac{S_{11}(x,y)S_{00}(x,y)-S_{01}(x,y)S_{10}(x,y)}{\sqrt{(S_{11}(x,y)+S_{01}(x,y))(S_{11}(x,y)+S_{10}(x,y))(S_{01}(x,y)+S_{00}(x,y))(S_{10}(x,y)+S_{00}(x,y))}}$
Yule	$\frac{S_{11}(x,y)S_{00}(x,y)-S_{01}(x,y)S_{10}(x,y)}{S_{11}(x,y)S_{00}(x,y)+S_{01}(x,y)S_{10}(x,y)}$
Kulzinsky	$\frac{S_{11}(x,y)}{S_{01}(x,y)+S_{10}(x,y)}$
Rogers-Tanimoto	$\frac{S_{11}(x,y)+S_{00}(x,y)}{S_{11}(x,y)+2(S_{01}(x,y)+S_{10}(x,y))+S_{00}(x,y)}$
Condorcet	$c_{ii'} = \sum_{k=1}^{m} c_{ii'}^k$ $c_{ii'}^k = 1$ if x_i and x_j have the same modality of variable V^k; 0 - otherwise

defined as follows:

$$d_{sE}(x, y) = d_E(x, y)^2 = \sum_{i=1}^{m}(x_i - y_i)^2. \tag{1.2}$$

For some algorithms the Euclidian distance is modified into the weighted Euclidian distance by introducing the weights used during the learning process in the distance expression.

In this section we will define as well some similarity measures used for the binary data. We will pay more attention to the Condorcet vote which is described in chapter 4. The table 1.2 shows the most interesting similarity measures used for the binary vectors.

In this table, the symilarity $S\,x, y$ represents:

- S_{11} represents the total number of attributes where x and y both have a value of 1;
- S_{01} represents the total number of attributes where the attribute of x is 0 and the attribute of y is 1.
- S_{10} represents the total number of attributes where the attribute of x is 1 and the attribute of y is 0.
- S_{00} represents the total number of attributes where x and y both have a value of 0.

1.4 Types of clustering methods

Clustering techniques are very diversified and they have been continuously developed for over a half century depending upon the optimization techniques, main methodology (statistical methods, system modelling, signal processing), and application areas. These algorithms are divided into two main categories of clustering: hierarchical [38] and objective function-based clustering.
In the next sections of this chapter we will introduce the main families of clustering algorithms. We divided clustering methods in:

- Hierarchical Clustering
- Graph-based Clustering Algorithms
- Center-based Clustering
- Grid-based Clustering Algorithms
- Density-based Clustering Algorithms
- Model-based Clustering Algorithms
- Relational analysis based clustering algorithms

In the next sections, we give some clustering approaches for each category, and for the most important (closely to our proposed methods) - the respective algorithm will be presented.

1.4.1 Hierarchical Clustering

Clustering algorithms are generally classified as partitional clustering and hierarchical clustering, based on the properties of the generated clusters ([37]; [58]; [69]; [70]). Partitional clustering divides data samples into a single partition, whereas a hierarchical clustering algorithm groups data with a sequence of nested partitions (figure 1.2).

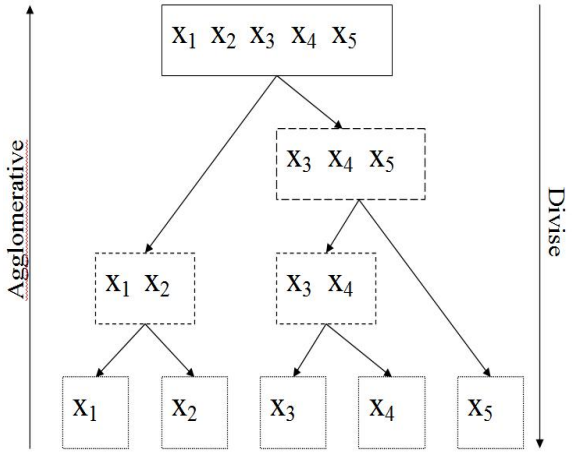


Figure 1.2: The principle of the Hierarchical Clustering Algorithm

As it is shown in the figure 1.2 there is two types of the hierarchical clustering methods: agglomerative approach and divide approach. Divide hierarchical clustering method starts from a cluster which contains all the data and divide this cluster until obtaining the desired clusters. Contrarily, agglomerative hierarchical clustering method starts from n clusters (n data) and will merge these clusters until obtaining a cluster containing the whole data.

1.4.1.1 Agglomerative Hierarchical Clustering

Agglomerative clustering starts with n clusters, each of which includes exactly one data point. A series of merge operations is then followed that eventually forces all objects into the same group. There are some extensions of this algorithm using different criteria presented in the figure 1.3 [31].

The general agglomerative clustering can be summarized by the procedure 1.

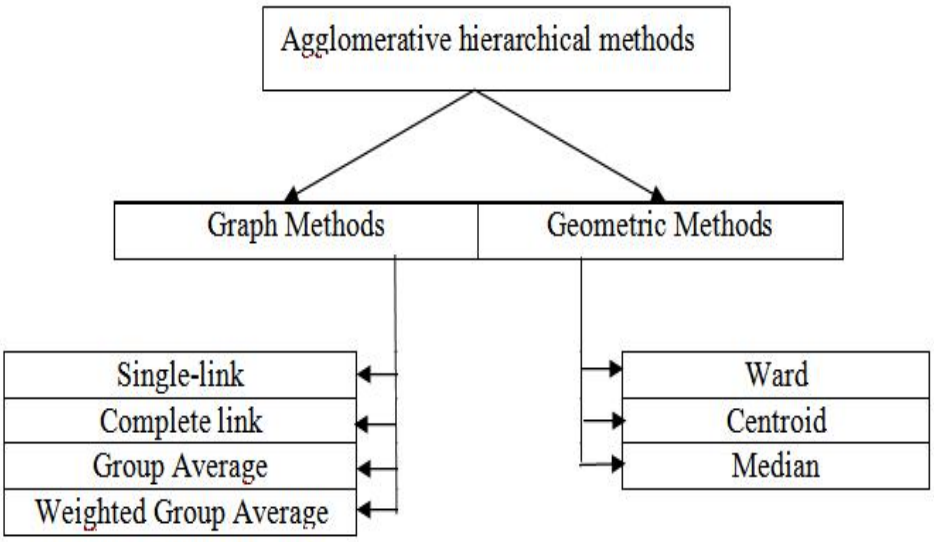


Figure 1.3: Types of Hierarchical Clustering Algorithms

Algorithme 1 : HCA Algorithm.

Input: Data set X, n - number of samples, k - number of clusters

for $i = 1$ to n **do**

Compute the proximity matrix (usually based on the distance function) for the k clusters;

end for

for $j = 1$ to k **do**

Compute/Search the minimal distance $d(C_i, C_j) = \min_{1 \leq m,l \leq k, m \neq l} d(C_m, C_l)$ where $d(.,.)$ is the distance function

end for

for $p = 1$ to k **do**

Update the proximity matrix by computing the distances between the cluster C_{ij} and the other clusters;

end for

REPEAT steps 2 and 3 until only one cluster remains.

For this work we used the Hierarchical Clustering algorithm with Wards criterion to avoid merging empty cells. This procedure will allow us to avoid clustering “cleaning” by eliminating the cells/clusters which have no captured samples.

1.4.2 Graph-based Clustering Algorithms

The main idea of the graph-based clustering algorithms is that firstly they build a graph (or hypergraph) and, secondly, they apply a clustering algorithm to partition the graph. The most used graph-based clustering algorithms are: ROCK, Chameleon, Cactus, etc.
ROCK (Robust Clustering using links) [55], [54] is an agglomerative hierarchical clustering algorithm that employs links to the sampled points to merge them into the clusters.
The criterion function which maximizes the k number of clusters is the following:

$$E_l = \sum_{i=1}^{k} n_i \times \sum_{p_q, p_r \in C_i} \frac{link(p_q, p_r)}{n_i^{1+2f(\theta)}} \tag{1.3}$$

where C_i denotes cluster i of size n_i and $f(\theta)$ is a function of the parameter θ. This function of θ is dependent on the dataset as well as the kind of clusters we are interested in, and has the property: each point belonging to cluster C_i has approximately $n_i^{f(\theta)}$ neighbors in C_i. To determine this function is a difficult problem. One of possibility for the $f(\theta)$ is $\frac{1-\theta}{1+\theta}$.
In order to decide whether to merge clusters, a *goodness measure* was proposed:

$$g(C_i, C_j) = \frac{link[C_i, C_j]}{(n_i + n_j)^{1+2f(\theta)} - n_i^{1+2f(\theta)} - n_j^{1+2f(\theta)}} \tag{1.4}$$

where $link[C_i, C_j]$ is the number of cross links between clusters C_i and C_j, that is:

$$link[C_i, C_j] = \sum_{p_q \ inC_i, p_r \in C_j} link(p_q, p_r) \tag{1.5}$$

The general ROCK algorithm is summarized in the procedure 2.

1.4.3 Center-based Clustering

Center-based algorithms are very effective for clustering high-dimensional datasets. The goal of these algorithms is to minimize their own objective functions. This type of clustering methods is the most used in the machine learning applications. One of the most well-known center-based clustering algorithm is the k-means algorithm.

Algorithme 2 : ROCK Algorithm.

Input: Data set X, n - number of samples, k - number of clusters

for $i = 1$ to $n - 1$ **do**
  for $j = i + 1$ to n **do**
    compute the $link(S)$ ($link \leftarrow link(p_i, p_j)$)
  end for
end for
for $i = 1$ to n **do**
  Build the local heap $q[i]$: $q[i] \leftarrow BuildLocalHeap(link, X)$
end for
Build a global heap Q that is ordered in decreasing order of the best goodness measure and contains all the clusters:
while size$(Q) > k$ **do**
  $u \leftarrow max(Q)$
  $v \leftarrow max(q[u])$
  delete(Q, v)
  $w \leftarrow$ merge(u, v)
  for $x \in q[u] \cup q[v]$ **do**
    $link[x, w] \leftarrow link[x, u] + link[x, v]$
    delete$(q[x], u)$; delete$(q[x], v)$ insert$(q[x], w, g(x, w))$; insert$(q[w], x, g(x, w))$
    update$(Q, x, q[x])$
  end for
  insert$(Q, w, q[w])$ deallocate$(q[u])$; deallocate$(q[v])$
end while
REPEAT

1.4.3.1 The k-means Algorithm

The basic clustering k-means algorithm [7], [87] is summarized as follows:

1) One fixes the k partitions randomly or based on some knowledge. One computes the matrix of cluster prototype $M = [m_1, ..., m_K]$;

2) One assigns each sample to the nearest cluster C_l:

$$x_j \in C_l, \ \ if \|x_j - m_l\| < \|x_j - m_i\| \ \ for \ \ j = 1, ...N, i \neq l, \ \ and \ \ i = 1, ..., K; \tag{1.6}$$

3) Update of the prototype matrix is based on the new partition:

$$m_i = \frac{1}{N_i} \sum_{x_j \in C_i} x_j; \tag{1.7}$$

4) Repeat steps 2 and 3 until the stabilization.

One can treat the k-means algorithm as an optimization problem. In this sense, the goal of the algorithm is to minimize a given objective function under certain conditions. Let $X = x_1, x_2, ..., x_n$ be a dataset with n samples/observations and k a fixed integer. The k-means objective function can be defined as follows:

$$R_{k-means}(U, Z) = \sum_{l=1}^{k} \sum_{i=1}^{n} \sum_{j=1}^{m} u_{il} d_E(x_{i,j}, z_{l,j}), \tag{1.8}$$

where

- U is an n x k matrix that satisfies the following conditions:
  1) $u_{i,l} \in 0, 1$ for $i = 1, 2, ..., n$, $l = 1, 2, ..., k$, $u_{ij} = 1$ indicates that the object i is allocated to cluster l,
  2) $\sum_{l=1}^{k} u_{i,l} = 1$ for $i = 1, 2, ..., n$.
- $Z = Z_1, Z_2, ..., Z_k$ is a set of objects representing the centroids of the k clusters;
- $d_E(x_{i,j}, z_{l,j})$ is the Euclidian distance (formula 1.1) between the object i and the centroid of the cluster l on the jth variable.

The optimization problem $R_{k-means}$ can be solved by iteratively solving the following two subproblems [65], [66]:

1) Subproblem $P1$: Fix $Z = \hat{Z}$ and solve the reduced problem $R_{k-means}(U, \hat{Z})$
2) Subproblem $P2$: Fix $U = \hat{U}$ and solve the reduced problem $R_{k-means}(\hat{U}, Z)$

In order to solve these subproblems Huang et. al proposed the following theorems:
Theorem1.1. In subproblem $P1$, let $\hat{Z}$ be fixed. Then the objective function $R_{k-means}(U, \hat{Z})$ is minimized if and only if:

$$u_{i,l} = \begin{cases} 1 \;\; if \;\; \sum_{j=1}^{m} d_E(x_{i,j}, z_{l,j}) \leq \sum_{j=1}^{m} \; d_E(x_{i,j}, z_{t,j}) \;\; for \; 1 \leq t \leq k \\ 0 \;\; otherwise \end{cases} \tag{1.9}$$

Theorem1.2. In the subproblem P2, let $\hat{U}$ be fixed. Then the objective function $R_{k-means}(\hat{U}, Z)$ is minimized if and only if:

$$z_{l,j} = \frac{\sum_{i=1}^{n} u_{i,l} x_{i,j}}{\sum_{i=1}^{n} u_{i,l}} \quad for 1 \leq l \leq k \ \ and \ \ 1 \leq j \leq m. \tag{1.10}$$

Based on this optimization problem, the k-means algorithm is defined as in the procedure 3.

Algorithme 3 : The k-means algorithm (optimization problem).

Input: Data set X, Number of clusters k, Dimension d:
Choose an initial U^0 and solve $R_{k-means}(U, Z)$ to obtain W^0;
Let Iter be the number of iterations;
for $t = 0$ to Iter **do**
  Let $\hat{U} \leftarrow U^t$ and solve $R_{k-means}(\hat{U}, Z)$ to obtain Z^{t+1}
  if $R_{k-means}(\hat{U}, Z^t) = R_{k-means}(\hat{U}, Z^{t+1})$ **then**
    Output $\hat{U}, Z^t$;
    Break;
  end if
  Let $\hat{Z} \leftarrow Z^{t+1}$ and solve $R_{k-means}(U^t, \hat{Z})$ to obtain U^{t+1};
  if $R_{k-means}(U^t, \hat{Z}) = R_{k-means}(U^{t+1}, \hat{Z})$ **then**
    Output $U^t, \hat{Z}$;
    Break;
  end if
end for
Output U^{Iter+1}, Z^{Iter+1}

1.4.3.2 x-Means Algorithm

In order to escape to fix the number of clusters for the k-means algorithm, Pelleg and Moore (2000) [110] proposed the x-Means algorithm by optimizing the Bayesian information criterion (BIC).
Using the BIC criterion and the spherical Gaussian distribution, the loglikelihood of the data is obtained as follows:

$$l(D) = log \prod_{i=1}^{n} P(x_i) = \sum_{i=1}^{n} (log \frac{1}{\sqrt{2\pi}\hat{\delta}^d} - \frac{1}{2\hat{\delta}^2} \|x_i - \mu_{(i)}\|^2 + log \frac{|C_{(i)}|}{n}). \tag{1.11}$$

The BIC or Schwarz [118] criterion is used to find the best model and to split the centroids. So, during the learning process, new centroids are added by splitting some centroids using the Schwarz criterion.

1.4.3.3 Trimmed k-Means Algorithm

The goal of the Trimmed K-means [67] approach is to estimate the centers of the k clusters which is not explicitly stated.

Let $X = x_1, x_2, ..., x_n$ be the samples of a given population in the $\Re^d$ dimension. Let F be the common distribution of the data, and $\Phi : \Re^+ \rightarrow \Re$ - the penalty function and $1 - \gamma \in (0, 1)$ be the level of trimming. So, the trimmed k-means of the set of samples X is defined by minimizing the following function:

$$\min_{Y} \min_{m_1,m_2,...,m_k \subset \Re^d} \frac{1}{\lfloor n\gamma \rfloor} \sum_{x \in Y} \Phi(\inf_{1 \leq j \leq k} \|x - m_j\|), \tag{1.12}$$

where $\lfloor x \rfloor$ represent the smallest integer $\geq x$, and $\lfloor n\gamma \rfloor$ - data points of the set X clustered in Y clusters.

Still, for the high-dimensional datasets, computing of all the centres is computationally expensive, and some extension of these algorithms were proposed like Impartial trimmed k-means (ITkM) which drops out a small proportion of the data before starting the search of the centers of groups [29].

1.4.4 Fuzzy Clustering Algorithms

For hard partitional clustering, each data object is exclusively associated with a single cluster. Fuzzy clustering approaches [17], [72] extends this notion to associate each data point to with every cluster with a degree of membership.

Let X be a dataset size $n \times m$, and let c be an integer $[1....n]$. The fuzzy c-partition is defined

by a matrix $U = (u_{li})$ size $c \times n$ with the constraints:

$$u_{li} \in [0, 1], 1 \leq l \leq c, 1 \leq i \leq n, \tag{1.13}$$

$$\sum_{l=1}^{c} = 1, 1 \leq i \leq n, \tag{1.14}$$

$$\sum_{i=1}^{n} > 0, i \leq l \leq c, \tag{1.15}$$

where u_{li} represent the membership degree between a sample i and the cluster l. There is also a hard partition extension of this technique, which computes the w_{li} as follows:

$$w_{li} = \begin{cases} 1 \;\; if \;\; l = \arg\max_{1 \leq j \leq c} u_{ji}, \\ 0 \;\; otherwise \end{cases} \tag{1.16}$$

1.4.5 Search-based Clustering Algorithms

Contrary to hard clustering algorithms, the search-based clustering approaches can explore the solution space beyond the local optimality in order to find a global optimal clustering which allows to fit the initial data. One of the most known and used search-based algorithms are the Genetic Algorithms (GA) which were proposed by Hooland (1973) [62], [63], [64], [51] inspired by the evolution and population of genes. Many extensions of classical clustering algorithms to the genetic algorithms were proposed [97], as: genetic k-means algorithm (GKA) [78], CONCLUS [13], the hierarchical cluster merging algorithms (HCMA) [48], etc.

1.4.6 Grid-based Clustering Algorithms

The idea of the Grid-based algorithms is to use the cells from a multiresolution grid data structures to form the clusters. So, the algorithms do firstly a quantization of the initial data space to form the grid and then all the process is performed in the new space.
There are two types of grid-based clustering approaches using the ordinary grid or the adaptive grid like shown in the figure 1.4 and 1.5. A general grid-based algorithm consist in the procedure 4.

One of the interesting grid based methods is the OptiGrid proposed by Keim in 1999 [61] which suggests cutting the initial data space into two half-spaces and to build the grid using the following density function which allows to cluster the data:

$$\hat{f}^X = \frac{1}{nh} \sum_{i=1}^{n} K(\frac{x - x_i}{h}) \tag{1.17}$$

Algorithme 4 : The general procedure for the grid-based model clustering algorithm

Input: Data set X; size of the grid : M
Create the grid structure from M cells
for $i = 1$ to M **do**
  Compute the cell density D_M
end for
Sort the cells accordingly to D_M.
for $j = 1$ to M (for the sorted vector) **do**
  Identify the cluster centers and b the clusters.
end for

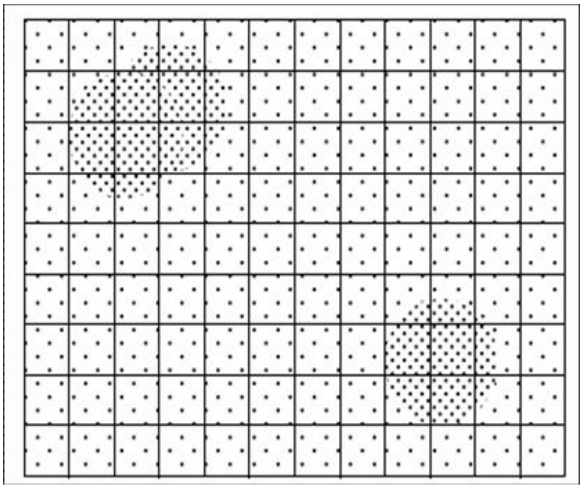


Figure 1.4: Ordinary Grid - based Clustering Algorithms

and

$$x \in S, c(x) = \sum_{i=1}^{k} 2^i H_i(x) \tag{1.18}$$

where,

$$H(x) = \begin{cases} 1 \;\; if \;\; \sum_{i=1}^{d} w_i x_i \geq 1, \\ 0 \;\; otherwise. \end{cases} \tag{1.19}$$

where h is smoothness level and K represent the kernel density estimator.
Most of the grid-based algorithms require user's information in order to set the grid size or the density thresholds. So, the main difficulty of these approaches is the choose of these parameters. To escape this problem, in literature there were proposed [98] a technique of

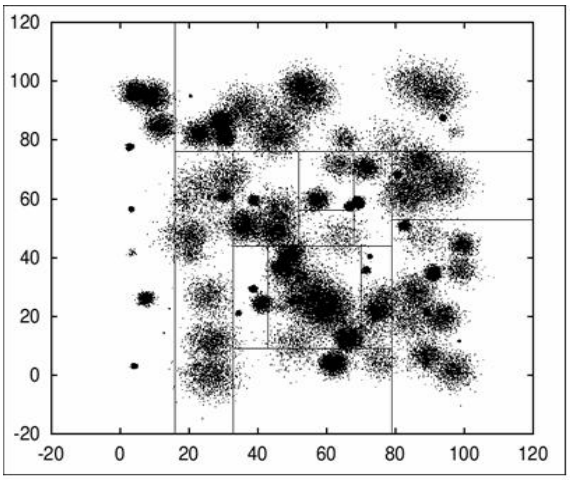


Figure 1.5: Adaptive Grid - based Clustering Algorithms (figure taken from [61])

adaptive grids that automatically determines the grid size using a heuristic method based on data distribution.

1.4.7 Density-based Clustering Algorithms

The algorithms based on the density look for more dense/concentrated data and aim at separating it from others by low-density regions. Instead of center-based algorithms, density-based clustering algorithms do not require the number of clusters, since they can automatically detect the clusters. In literature there were proposed many density-based algorithms like: BRIDGE, DENCLUE, DBCLASD, DBSCAN, GF-DBSCAN, l-DBSCAN, etc [30], [86], [36], [60], [135], [126], [132].
The main idea of density-based cluster is that a density based neighborhood called Eps-neighborhood is computed for each point of a cluster. The Eps-neighborhood ($\epsilon > 0$) has to contain a number of points, i.e. the density in the Eps-neighborhood of points has to exceed some threshold [36]. We will show the principle of the DBSCAN algorithm and we'll start with the definitions of the density-reachable and density-connected points used by all density-based algorithms.

Definition 1.1 (Density-reachable.) *A point x is density-reachable from a point y if there is a sequence of points* $x = x_1, x_2, ..., x_i = y$ *so that* x_l *is directly density-reachable from* x_{l+1} *for* $l = 1, 2, ..., i - 1$.

Definition 1.2 (Density-connected.) *Two points x and y are density-connected with respect to ϵ and N_{min} if there is a point z such that both x and y are density-reachable from z with respect to ϵ and N_{min}, where N_ϵ is the neighborhood of the point x defined as:*

$$N_\epsilon(x) = y \in X : d(x, y) \leq \epsilon, \tag{1.20}$$

where X is the set of samples and $d(.,.)$ is a certain distance.

For the DBSCAN algorithm, the construction of a cluster C within an ϵ and N_{min} for a subset of samples X is done with the respect to the following conditions:

1) The condition of maximality:
   $\forall x, y \in X$ if $x \in C$ and y is density-reachable from x with respect to ϵ and N_{min}, then $y \in C$.

2) The condition of connectivity:
   $\forall x, y \in C$ and y are density-connected with respect to ϵ and N_{min}

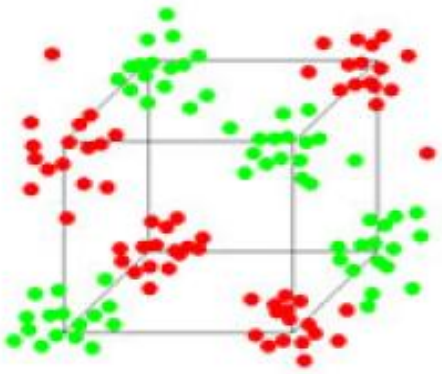


Figure 1.6: Example of density separated dataset

The negative point of density-based clustering algorithms is that, if the dataset has no low-density regions and contains non-spatial variables it is difficult to find the separation between the clusters; this is the case for un-"density-connected sets". As it is shown in the figure 1.6 for this dataset it is easy to detect clusters using a density-based algorithm because all the clusters are separated between them. But, if there are regions of clusters which are intersected between them (like in the case in the figure 1.7), these algorithms will consider this intersection like a cluster which is not the case. This is because, in literature there

were introduced some extensions of the classical density algorithms to treat these problems, like GDBSCAN [117] which take into account the non-spatial variables using the notion of *MinWeight*. *MinWeight* use a weighted cardinality function *wCard* for the sets of objects. The weight of a single sample x can be expressed by the weighted cardinality of the singleton containing the x, i.e. $wCard(x)$. So, using weighted techniques, the problem of un-"density-connected sets" can be resolved.

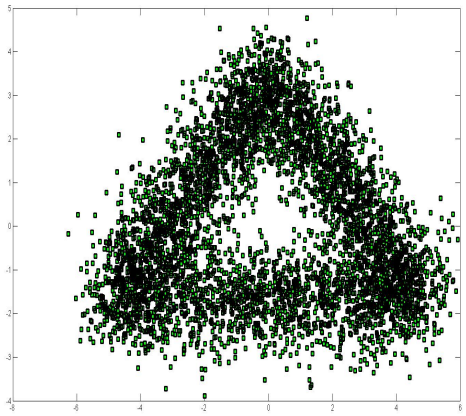


Figure 1.7: Example of density non-separated dataset

1.4.8 Model-based Clustering Algorithms

The model-based clustering approaches are developed using probability models which assumes that the data are generated by a mixture of probability distributions in which each component represents a cluster [39], [113].
The general schema for this type of clustering methods is given in the Algorithm 5. While other clustering algorithms have the problem to select the best clustering method for a given dataset and to determine the "correct" number of clusters, the model-based clustering approaches reduce this clustering problem to the model selection problem which could be done automatically in the probability framework.

1.4.9 Relational analysis based clustering algorithms

Another interesting but not very known "family" of clustering algorithms is the Relational Analysis based clustering methods which use the statistical relational notions as Condorcet

Algorithme 5 : The general procedure for the model-based clustering approaches

Input: Data set X with the n dimension:
for $i = 1$ to n **do**
  Apply an Agglomerative model-based clustering
  Output: Dendogram
end for
Initialization for the EM algorithms: the number of components k and other initials parameters.
for $j = 1$ to k **do**
  EM algorithm
end for
Determine the final model using a criterion (BIC for example)
OUTPUT: Final estimated model with the final number of components

voting measure [26] (Condorcet similarity measure in table 1.2) to cluster the data [88], [89], [90]. This type of algorithms was proposed for the first time by Marcotorchino in 1979 [91]. This approach were developed initially for the binary data, and we have a big interest in this approach thanks to its accuracy to cluster the data. We can mention that this technique was used during the Infom@gic project by the Thales Communications S.A. and it makes part of our work in order to find the best clustering results (see Chapter 4).

Relational analysis is a statistical data analysis which can be used for large numeric datasets in order to cluster the data. It was initiated and developed by F. Marcotorchino and P. Michaud [91, 92] at the IBm's European Center of Applied Mathematics (ECAM) in 1980. The main idea of RA is the usage of Condorcet Concept in order to transform initial matrix into a squared matrix. The similarity between the features is represented here like agreement and disagreement. So, the terms $c_{ii'}$ of the Condorcet's matrix represent the degree of agreement between the individual x_i and individual x_j taken into consideration all variables measured on the population (dataset).
So, to cluster a dataset $\mathcal{X}$ composed of n observations $(x_1, x_2, ..., x_n)$ described by m variables $(V^1, V^2, ..., V^m)$, the algorithm firstly starts by transforming each column V^k into a relational matrix C^k with the general term $c^k_{ii'}$ defined by:

$$c^k_{ii'} = \begin{cases} 1 & \text{if } x_i \text{ and } x_{i'} \text{ have the same modality of variable } V^k \\ 0 & \text{otherwise} \end{cases} \tag{1.21}$$

This term represents the similarity between the observations x_i and $x_{i'}$, with respect to variable V^k. Once all the m matrices C^k are built up, we construct a global relational matrix C called "Condorcet's matrix" of general term $c_{ii'}$ representing the global similarity of x_i and

$x_{i'}$ with respect to the whole set of the m variables: $c_{ii'} = \sum_{k=1}^{m} c_{ii'}^{k}$. This global similarity has the so called "self maximal similarity property defined by: $c_{ii'} \leq \mathcal{M}_{ii'} \forall x_i, x_{i'}$, where $\mathcal{M}_{ii'} = Min(c_{ii}, c_{i'i'})$ is the "maximum possible similarity" between the two observations x_i and $x_{i'}$.

Using the global similarity $c_{ii'}$ and the "maximum possible similarity" $\mathcal{M}_{ii'}$ between x_i and $x_{i'}$, the authors [91] define their dissimilarity $\overline{c}_{ii'}$ as the complement of their global similarity to their "maximum possible similarity":

$$\overline{c}_{ii'} = \mathcal{M}_{ii'} - c_{ii'} \tag{1.22}$$

Two observations will be, a priori, in the same cluster of the final expected partition as soon as their similarity will be greater than their dissimilarity i.e.: $c_{ii'} \geq \overline{c}_{ii'}$. The required final partition will be represented by a $N \times N$ binary squared matrix Y with general term $y_{ii'}$ defined as follows:

$$y_{ii'} = \begin{cases} 1 & \text{if } x_i \text{ and } x_{i'} \text{ are in the same cluster} \\ & \text{of the final partition} \\ 0 & \text{otherwise} \end{cases} \tag{1.23}$$

This partition will be obtained by maximizing the Condorcet's criterion $\mathcal{C}(Y)$ defined hereafter:

$$\mathcal{C}(Y) = \sum_{i=1}^{n} \sum_{i'=1}^{n} (c_{ii'} y_{ii'} + \overline{c}_{ii'} \overline{y}_{ii'})$$

where:

$$\overline{y}_{ii'} = 1 - y_{ii'} \tag{1.24}$$

Using the expressions (1.22) and (1.24), the criterion $\mathcal{C}(Y)$ can be rewritten as:

$$\mathcal{C}(Y) = \sum_{i=1}^{n} \sum_{i'=1}^{n} (2c_{ii'} - \mathcal{M}_{ii'}) y_{ii'} + \sum_{i=1}^{n} \sum_{i'=1}^{n} \overline{c}_{ii'} \tag{1.25}$$

As the second member of the sum of expression (1.25) is a constant, maximizing the Condorcet's criterion is equivalent to maximizing the following criterion $\mathcal{C}'(Y)$

$$\mathcal{C}'(Y) = \sum_{i=1}^{n} \sum_{i'=1}^{n} \left(c_{ii'} - \frac{\mathcal{M}_{ii'}}{2} \right) y_{ii'}$$

The cost function of the criterion $\mathcal{C}'(Y)$ will be positive when the similarity $c_{ii'}$ between two observations x_i and $x_{i'}$ is greater or equal to half of their "possible maximal similarity". This condition is sometimes very difficult to reach, especially when the number of variables (or descriptors) is very high compared to the number of observations i.e. $m >> n$, this is usually

the case when the dataset to be clustered is a set of documents. In that case, the number of clusters of the final partition will be so high that it could deprive the clustering task of its interest for practical purpose. As the goal of the clustering task is to summarize the amount of data into simpler structures, to avoid this problem, a solution consists in relaxing the cost function related to the clustering criterion. To reach that goal it is sufficient to replace the coefficient 1/2 of $\mathcal{M}_{ii'}$ by a parameter α such that $0 < \alpha < 1/2$. The new formulation of the criterion $\mathcal{C}'(Y)$ will be then:

$$\mathcal{C}'(Y) = \sum_{i=1}^{n} \sum_{i'=1}^{n} (c_{ii'} - \alpha \times \mathcal{M}_{ii'})\, y_{ii'}$$

Thus, the mathematical formulation of the relational analysis clustering problem is:

$$\max_{Y} \mathcal{C}'(Y)$$

under the constraints:

$$\begin{cases} y_{ii'} \in \{0,1\} & \forall\ (O_i, O_{i'}) \in \mathcal{P}^2 & \text{(binarity)} \\ y_{ii} = 1 & \forall\ O_i \in \mathcal{P} & \text{(reflexivity)} \\ y_{ii'} - y_{i'i} = 0 & \forall\ (O_i, O_{i'}) \in \mathcal{P}^2 & \text{(symmetry)} \\ y_{ii'} + y_{i'i''} - x_{ii''} \leq 1 & \forall\ (O_i, O_{i'}, O_{i''}) \in \mathcal{P}^3 & \text{(transitivity)} \end{cases}$$

One of the disadvantages of this approach is the α coefficient which is not clear to fix because its value depends on the dataset and on the desired number of clusters. That's why there were proposed a heuristic version of the Relational Analysis Clustering method which is defined in the next section.

1.4.9.1 The RA heuristic

The exact solution to the problem above can be obtained by linear programming techniques when the studied population is relatively small (few hundreds). But, in practice, the dataset size can often exceed hundreds of thousands or millions of observations. This situation leads to use heuristics, to get the "best" and closest partition to the exact one, in reasonable time processing. Below we'll give the description of the heuristics which was used by the relational analysis methodology [14], [81].

Phase 1

This step consists in initializing the clustering process by building a first partition:

1) Initialization: take randomly a first observation which constitutes the first cluster of the unknown partition

2) take an observation $x_i \in \mathcal{P}$, and compute its link $\mathcal{L}_{i\mathcal{V}}$ (expression 1.26) with all the existing clusters $\mathcal{V}$.

$$\mathcal{L}_{i\mathcal{V}} = \sum_{i' \in \mathcal{V}} \mathcal{L}_{ii'} \tag{1.26}$$

where the link $\mathcal{L}_{ii'}$ between x_i and $x_{i'}$:

$$\mathcal{L}_{ii'} = c_{ii'} - \alpha \times \mathcal{M}_{ii'} \tag{1.27}$$

This observation is assigned to the cluster which has the biggest strictly positive link with. If all the links are negative, then we create a new cluster to put in this new observation.

3) **Repeat** this process until all observations of population $\mathcal{P}$ had been assigned to a cluster.

Phase 2

At the end of the first step, we obtain a partition with a number of clusters.

1) **Merging two clusters:** Now, the algorithm will take the clusters one after another and will compute the link $\mathcal{L}_{\mathcal{V}\mathcal{V}'}$ (expression 1.28) of each cluster $\mathcal{V}$ with all the others $\mathcal{V}'$.

$$\mathcal{L}_{\mathcal{V}\mathcal{V}'} = \mathcal{A}_{\mathcal{V}\mathcal{V}'} - \alpha \times \mathcal{M}_{\mathcal{V}\mathcal{V}'} \tag{1.28}$$

where the agreement $\mathcal{A}_{\mathcal{V}\mathcal{V}'}$ between the two clusters:

$$\mathcal{A}_{\mathcal{V}\mathcal{V}'} = \sum_{i \in \mathcal{V}} \sum_{i' \in \mathcal{V}'} c_{ii'}.$$

The disagreement $\overline{\mathcal{A}}_{\mathcal{V}\mathcal{V}'}$ between the two clusters is:

$$\overline{\mathcal{A}}_{\mathcal{V}\mathcal{V}'} = \sum_{i \in \mathcal{V}} \sum_{i' \in \mathcal{V}'} \overline{c}_{ii'},$$

and the possible maximal agreement $\mathcal{M}_{\mathcal{V}\mathcal{V}'}$ between the two clusters:

$$\mathcal{M}_{\mathcal{V}\mathcal{V}'} = \sum_{i \in \mathcal{V}} \sum_{i' \in \mathcal{V}'} \mathcal{M}_{ii'}.$$

The algorithm will, then, merge the clusters, which have the best link (higher strict positive value). This must be carried out as long as there is a possibility to improve the criterion $\mathcal{C}'(Y)$.

2) **Transferring an observation from a cluster to another one**. When no cluster's merging is possible, one take the observations of each cluster and compute the link $\mathcal{L}_{i\mathcal{V}}$ (expression 1.26) of each observation x_i with the other clusters $\mathcal{V}$. If an observation has a better link with another cluster than its own, then this observation is transferred from its own cluster to this new cluster. This will be carried out, as long as improvement of the criterion $\mathcal{C}'(Y)$ occurs.

When, no observation's transfer is possible, we turn back to the merging step to see whether it is possible to improve the Condorcet's criterion by merging other clusters. These four steps will be applied, until no more improvements of the criterion occurred.

1.4.9.2 Relational Analysis based k-means Algorithm

In order to avoid the number of clusters for the k-means algorithm, Lazhar Labiod [81] proposed a Relational Analysis approach for the k-means algorithm, introducing the next interclass inertia:

$$I_Y = \frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}\|x_i - x_{i'}\|^2\frac{y_{ii'}}{y_i} \tag{1.29}$$

where the relational features $y^k_{ii'} = \sum_k \chi_{ik}\chi_{i'k}$ is the general term of the unknown equivalence's relationship Y; χ_{ik} and $\chi_{i'k}$ are dummy features from partitions matrix. The term x_i represent the ith vector of the dataset X.

In place of the Euclidian distance used in the classical k-means algorithm, this formulation allows to use another distance, or a similarity/dissimilarity or any other function which is noted $D_{ii'}$. So, replacing the Squared Euclidian distance between two samples (i, i') by the $D_{ii'}$, the clustering problem has the next relational form:

$$\min_Y I_W(Y) \tag{1.30}$$

$$\tag{1.31}$$

where

$$I_W(Y) = \frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N} D_{ii'}\frac{y_{ii'}}{y_{i.}} \tag{1.32}$$

$$subject\ \ to: \tag{1.33}$$

$$\begin{cases} Y \quad equivalency's\ \ relation\ \ on\ \ IxI \\ \sum_{i=1}^{N}\sum_{i=1}^{N}\frac{y_{ii'}}{y_{i.}y_{.i'}} = K \end{cases} \tag{1.34}$$

To manage different types of data, there is a general relational formulation of the k-means algorithm proposed by the same authors using different criteria for each type of data. This unified clustering approach is illustrated as follows:

$$(\min_{Y}) \ or \ (\max_{Y}) \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} K_{ii'} \frac{y_{ii'}}{y_{i0}} \tag{1.35}$$

under the constraints 1.34.
Where K is the affinity matrix, which could be a distance matrix, a similarity matrix, or a difference matrix. This matrix is defined as follows:

$$K_{ii'} = \begin{cases} D_{ii'} \in \Re^{+} \quad (Distance) \mapsto Type \ (1) \\ S_{ii'} \in \Re^{+} \quad (Similarity) \mapsto Type \ (1) \\ D_{ii'} - S_{ii'} \in \Re \quad (Distance - Similarity) \mapsto Type \ (2) \\ S_{ii'} - D_{ii'} \in \Re \quad (Similarity - Distance) \mapsto Type \ (2) \end{cases} \tag{1.36}$$

So, one has two types of criteria:

- 1st category : (criterion with positive or zero cost):
  This criterion is based on distance or on similarity. The cost of criterion is always positive or zero ($S_{ii'}, D_{ii'} \in \Re^{+} \ \forall i, i'$);
- 2nd category : (criterion with real cost) :
  The criterion of the real cost concerns the approaches which use simultaneously a distance and a similarity measure. Depending on the profile of each sample, this criterion's cost could be positive and even negative. These criteria has Condorcet's cost type ($(c_{ii'} - \overline{c_{ii'}})$ or $(\overline{c_{ii'}} - c_{ii'}) \forall i, i'$).

1.4.10 Self Organizing Maps - SOM

The self-organizing maps introduced by [76] have been widely used for unsupervised classification and visualization of multidimensional datasets [27]. There is a wide variety of algorithms for topological maps derived from the original model proposed firstly by Kohonen ([18], [82], [139], [53], [127]). These models are different from each other, but share the same idea to present the large data in a simple geometric relationship on a reduced topology like is shown in the figure 1.8.
This model consists in unsupervised classification of a learning set $A = \{x^{(i)} \in \Re^{n}, i = 1...N\}$

where the observation $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, ..., x_j^{(i)}, ..., x_n^{(i)})$ is of dimension n. This classical model consists in a discrete set C of cells (neurons) called map. This map has a discrete topology defined by undirected graph; usually it is a regular grid in two dimensions. The inclusion of the proximity notion on the map size C makes it necessary to define a topological neighborhood. To model the influence notion of a cell k on a cell l, which depends on their proximity, we use a kernel function K, $(K \geq 0 \ \ et \ \ lim_{|x|\to\infty} K(x) = 0)$. The mutual influence between two units k and l is defined by the function $K_{k,l}(.)$:

$$K_{ij} = \frac{1}{\lambda(t)} exp\left(-\frac{d_1^2(i,j)}{\lambda^2(t)}\right) \tag{1.37}$$

where $\lambda(t)$ is the temperature's function modeling the neighborhood's range:

$$\lambda(t) = \lambda_i(\frac{\lambda_f}{\lambda_i})^{\frac{t}{t_{max}}} \tag{1.38}$$

with λ_i and λ_f are the initial temperature (for example $\lambda_i = 2$ and $\lambda_f = 0.5$) and the final temperature; and t_{max} - the maximum allotted time (t - the number of iterations, x - the number of learning examples). The Manhattan distance $d_1(.,.)$ between two map units r and s with the coordinates (k, m) and (i, j) is defined by:

$$d_1(r,s) = |i-k| + |j-m| \tag{1.39}$$

The function $K_{k,l}(.)$ is a Gaussian introduced for each neuron of the map with a global neighborhood. The size of this neighborhood is limited by the standard Gaussian deviation $\lambda(t)$. The units that are beyond this range have a significant influence (but not null) on the considered cell. The range $\lambda(t)$ is a function decreasing in time, so, the neighborhood function $K_{k,l}(.)$ will have the same trend with a standard deviation decreasing in time.
For each cell k of the grid is associated a reference (prototype) vector $w^{(k)} = (w_1^{(k)}, w_2^{(k)}, ..., w_i^{(k)}, ..., w_n^{(k)})$ of size n. We note by W the set of prototypes. The learning of this model will be reached by minimizing the distance between input pattern and prototypes of the map, weighted by the neighborhood. A gradient algorithm can be used for this. The criterion to minimize in this case is defined by:

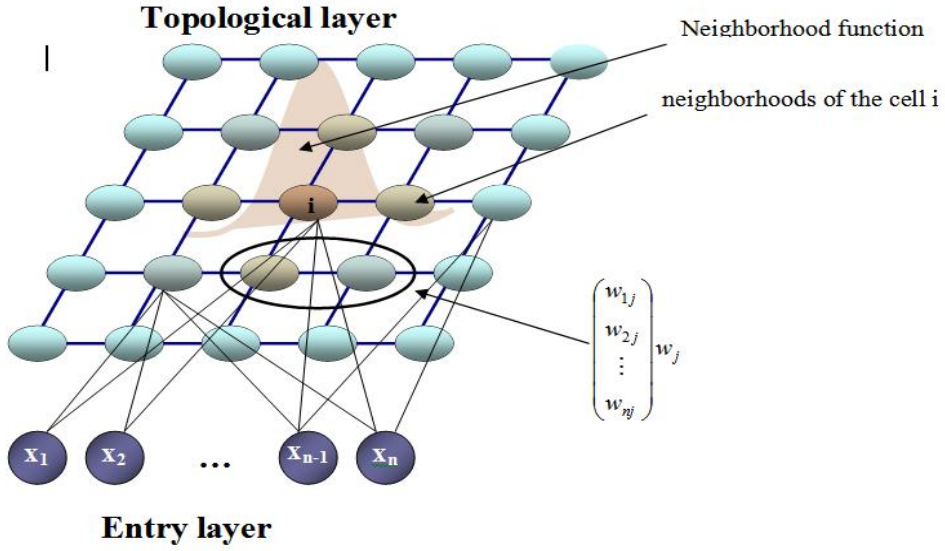


Figure 1.8: The principle of the Self-Organizing Map

$$R_{SOM}(\chi, \mathcal{W}) = \sum_{i=1}^{N} \sum_{j=1}^{|w|} \mathcal{K}_{j,\chi(\mathbf{x}^{(i)})} \|\mathbf{x}^{(i)} - \mathbf{w}^{(j)}\|^2 \tag{1.40}$$

where χ assigns each pattern (observation) $x^{(i)}$ to a single cell of the SOM.
At the end of the learning, the self-organizing map determines a data partition in $|w|$ groups associated with each cell k of the map. Each group or cell is associated with a reference vector $w^{(k)} \in \Re^n$, which will be the representative, the "local mean" or the prototype of the observation's set associated with this cell.

1.5 Complexity of the clustering algorithms

If the clustering algorithms have the same goal and are similar between them, their computational time differs for each type of technique. In the table 1.3 we show the complexity of the main algorithms presented in this chapter. n stands for the size of the observations, m is the variable size, C means the number of clusters or cells in the case of the maps, m_m represents the maximum number of neighborhoods per data point and m_a is the average number of neighbors per same data point.

Table 1.3: Complexity for the clustering algorithms

Name	Computational Complexity
HCA	$O(c)$
k-means	$O(Cnm)$
x-means	$O(c)$
trimmed-k-means	$O(c)$
ROCK	$O(n^2 + n.m_m.m_a + n^2 \log n)$
DBSCAN	$O(n. \log n)$
OptiGrid	$O(n \times c)$
RA	$O(n \times C)$
RA Kmeans	$O(n \times C)$
SOM	$O(n \times C)$

For example, to compute the complexity of the classical SOM algorithm, we analyze the complexity for each phase.
Let's show the complexity of the case with parallel computing of the SOM proposed in [57]. If the total number of neurons is K, the minimum time to find the best matching unit is $O(k)$. The neighborhood is computing in $O(1)$, independent of the neighborhood shape or updating scheme. The updating of neurons takes place in $O(m)$. Thus, the total time for the SOM training is $O(m + K)$. But, in practice, the parallelism is not always possible due to the limitations in communication and parallel processing.

Table 1.4: Time Complexity in SOM computation (table taken from [23])

Step	Ideal ($P = K$)	Shared memory multiprocessor ($P < K$)	Tree shape multiprocessor ($P = K$)
Broadcast $t = [t_1, ..., t_m]$	$O(1)$	$O(m)$	$O(m \log P)$
Compute distances	$O(m)$	$O(m \times K/P)$	$O(m \times K/P)$
Find local winner	-	$O(K/P)$	$O(K/P)$
Find global winner	$O(K)$	$O(P)$	$O(\log P)$
Broadcast winner index	$O(1)$	$O(1)$	$O(\log P)$
Compute neighborhood	$O(1)$	$O(K/P)$ or $O(1)$	$O(K/P)$ or $O(1)$
Update	$O(m)$	$O(m \times K/P)$	$O(m \times K/P)$
Total	$O(m + K)$	$O(P + m \times K/P)$	$O(m \log P + m \times K/P)$

In the table 1.4 the K is the number of cells from the map; m - the input vector dimension; and P is the number of processors. In the respective table for each neuron there is a corresponding process, but in our case we do not implement a parallel computation and we don't have broadcast phases. On the contrary, computation time increases to $O(n \times C)$ where n is the number of observations and C is the map size.

1.6 Cluster Validation

Since clustering is an unsupervised process and most of theses algorithms are very sensitive to their initial assumptions, some evaluation is required to describe/analyze the clustering results [45, 69, 19]. Cluster validity represents the goodness measure of a clustering result relative to others created by other clustering algorithms, or by the same algorithm using different parameter values.
In general, there are three fundamental criteria to investigate the cluster validity: external criteria, internal criteria, and relative criteria. In the following we show main clustering validity indices.

1.6.1 Davies-Bouldin Validity Index

The DB index [32] is an internal index between two clusters and it's computing as follows: A similarity measure R_{ij} between the clusters C_i and C_j is defined based on a measure of dispersion of a cluster C_i, let s_i, and a dissimilarity measure between two clusters d_{ij}. The R_{ij} index is defined to satisfy the following conditions:

- $R_{ij} \geq 0$
- $R_{ij} = R_{ji}$
- if $s_i = 0$ and $s_j = 0$ then $R_{ij} = 0$
- if $s_j \geq s_k$ and $d_{ij} = d_{ik}$ then $R_{ij} \geq R_{ik}$
- if $s_j = s_k$ and $d_{ij} < d_{ik}$ then $R_{ij} \geq R_{ik}$

So, these conditions impose to R_{ij} to be a non-negative and symmetric. To satisfy the above-mentioned conditions, we have: $R_{ij} = \frac{(s_i + s_j)}{d_{ij}}$.
Then, the DB index is defined as:

$$DB_{nc} = \frac{1}{n_c} \sum_{i=1}^{n_c} R_i \tag{1.41}$$

$$and \;\; R_i = max_{i=1,...,n_c, i \neq j} R_{ij}, \;\; i = 1, ..., n_c \tag{1.42}$$

The DB_{nc} is the average similarity between each cluster $c_i, i = 1, ..., n_c$ and its most similar one. So, we seek clusterings that minimize the DB, and thus have minimum possible similarity with the clusters. Some variants of this index were proposed in literature which are based on Minimum Spanning Tree (MST), Relative Neighborhood Graph (RNG) and the Gabriel Graph (GC) concepts.

1.6.2 Silhouette Validation Method

The Silhouette validation technique (Rousseeuw, 1987 [116]) calculates the silhouette width for each sample, average silhouette width for each cluster and overall average silhouette width for a total dataset. Using this approach each cluster could be represented by so-called silhouette, which is based on the comparison of its tightness and separation. The average silhouette width could be applied for evaluation of clustering validity and also could be used to decide how good is the number of selected clusters.

To construct the silhouettes $S(i)$ the following formula is used:

$$S(i) = \frac{(b(i) - a(i))}{max\ a(i), b(i)} \tag{1.43}$$

where $a(i)$ is the average dissimilarity of i-object to all other objects in the same cluster; $b(i)$ - minimum of average dissimilarity of i-object to all objects in other clusters (in the closest cluster).

It is followed from the formula that $-1 \leq s(i) \leq 1$. If silhouette value is close to 1, it means that sample is "well-clustered" and it was assigned to a very appropriate cluster. If silhouette value is about zero, it means that the sample could be assigned to another closest cluster as well, and the sample lies equally far away from both clusters. If silhouette value is close to -1, it means that sample is "misclassified" and is merely somewhere in between the clusters. The overall average silhouette width for the entire plot is simply the average of the $S(i)$ for all objects in the whole dataset.

The largest overall average silhouette indicates the best clustering (number of clusters). Therefore, the number of cluster with maximum overall average silhouette width is taken as the optimal number of clusters.

1.6.3 Rand Index

The Rand index [114] or Rand measure is a measure of the similarity between two data clusters.

Given a set of n objects $S = \{O_1, ..., O_n\}$, suppose $U = \{u_1, ..., u_R\}$ and $V = \{v_1, ..., v_C\}$ represent two different partitions of S such that $\cup_{i=1}^{R} u_i = S = \cup_{j=1}^{C} v_j$ and $u_i \cap u_{i'} = \emptyset = v_j \cap v_{j'}$ for $1 \leq i \neq i' \leq R$ and $1 \leq j \neq j' \leq C$. Suppose that U is our external criterion and V is the clustering result, we define the following:

- a, the number of pairs of objects that are placed in the same class in U and in the same cluster in V;
- b, the number of pairs of objects in the same class in U but not in the same cluster in V;
- c, the number of pairs of objects in the same class in V but different in U;
- d, the number of pairs of objects in different classes and different clusters in both partitions.

The both, a + b can be considered as agreements between U and V, and c + d as the number of disagreements between these two partitions. The Rand index [114] is computed:

$$R = \frac{a+b}{a+b+c+d}\ ; \tag{1.44}$$

The Rand index has a value between 0 and 1, with 0 indicating that the two data clusters do not agree on any pair of points and 1 indicating that the data clusters are exactly the same.

1.6.4 Class Accuracy (Purity Index)

A simple approximation of accuracy for unsupervised learning that employs external class information was described by Topchy et al. (2003) [3]. By finding the optimal correspondence between a dataset annotated class labels and the clusters in a given partition, a performance measure may be derived that reflects the proportion of instances that were correctly assigned. A high value for this measure generally indicates a high level of agreement between a clustering and the annotated natural classes. This measure is only applicable when the number of clusters k is the same as the number of natural classes.
The purity index is a simple and transparent evaluation measure. To compute purity, each cluster is assigned to the class which is most frequent in the cluster, and then the accuracy of this assignment is measured by counting the number of correctly assigned data. Formally:

$$\text{purity}(\Omega, \mathbb{C}) = \frac{1}{N} \sum_{k} \max_{j} |\omega_k \cap c_j|$$

where $\Omega = \{\omega_1, \omega_2, \ldots, \omega_K\}$is the set of clusters and $\mathbb{C} = \{c_1, c_2, \ldots, c_J\}$ is the set of classes. Bad clusterings have purity values close to 0, a perfect clustering has a purity of 1 .

1.6.5 Jaccard index

In this index (Jaccard, 1912), which has been commonly applied to assess the similarity between different partitions of the same dataset, the level of agreement between a set of class labels C and a clustering result Ω is determined by the number of pairs of points assigned to the same cluster in both partitions:

$$J(C, \Omega) = \frac{a}{a + b + c} \tag{1.45}$$

where a denotes the number of pairs of points with the same label in C and assigned to the same cluster in Ω, b denotes the number of pairs with the same label, but in different clusters and c denotes the number of pairs in the same cluster, but with different class labels. The index produces a result in the range $[0, 1]$, where a value of 1 indicates that C and Ω are identical.

1.6.6 SOM quality measures

As our approaches are based on the Self-Organizing Maps, we'll introduce the validation measures for the obtained map in order to be able to compare our map with other algorithms which build the map using the same initialization parameters.

1.6.6.1 Quantization error

One of the most used criteria to assess the quality of a topological map is the quantization error. This error measures the average distance between each data vector and its winning neuron (BMU: Best Match Unit). It is calculated using the following expression:

$$Qe = \frac{1}{N}\sum_{i=1}^{N} ||x^{(i)} - w^{(\chi(x^{(i)}))}||^2 \tag{1.46}$$

where N represents the number of data and $w^{(\chi(x^{(i)}))}$ is the closest prototype to $x^{(i)}$.

1.6.6.2 Topographic error

One can also measure the quality of the map by estimating the topological preservation. The most used index to quantify this information is the topographic error, which was proposed by Kohonen ([75], [74], [8], [112]). This error measures the proportion of all examples for which the first and second winning neurons (BMU) are not adjacent vectors. The idea of this index is to show how the granularity SOM map can preserve the topology of the data. The topographic error is defined by:

$$Te = \frac{1}{N}\sum_{i=1}^{N} \alpha(x^{(i)}) \tag{1.47}$$

where the function $\alpha(x^{(i)}) = 1$ if the first and the second BMU of $x^{(i)}$ are adjacent and 0 otherwise.

1.7 Cluster Characterization

As it is shown in the beginning of this chapter, in the machine learning domain there are many clustering algorithms based on different statistical techniques which can provides dif-

ferent clustering results for the same dataset. Usually this clustering results are analyzed and processed by an user(expert) of the domain, but the problem appears when dealing with the high dimensional datasets. On such dataset, the human expertise(evaluation) became difficult even impossible (ex.: How to characterize a cluster formed from 2000 observations in a 6400 dimension?). This is why, in the last decade, the researchers in the machine learning has give a great attention to the dimensionality reduction techniques or to the cluster analysis automated techniques [129]. The clustering characterization doesn't means only dimensionality reduction, it is an unsupervised tool which allow to detect the structure of the data. To attempts the clustering characterization we can use or the Subspace Clustering algorithms or the Dimensionality reduction techniques on the formed clusters. Sometimes these two types of clustering analysis techniques are equivalent.

1.7.1 Subspace Clustering

Subspace clustering attempts to find clusters in different subspaces of the same dataset. In other words, the subspace clustering methods finds clusters with their relevant features by removing unrelevant features [105], [104]. This technique is closest to features selection, but it is doing automatically for each cluster in order to not allow the lose of the information because not all the clusters has the same relevant features. There are two types of the subspace clustering methods shown in the figure 1.9.

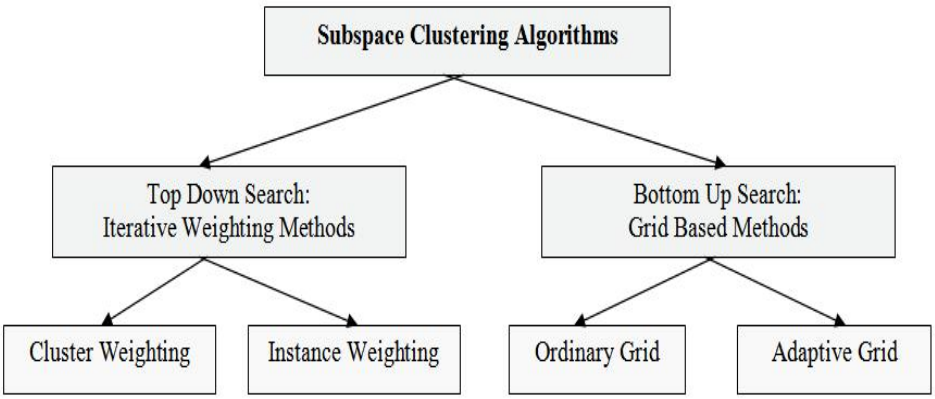


Figure 1.9: Subspace clustering algorithms

1.7.1.1 Top-Down Subspace Search Methods

This type of approaches starts by finding the clusters in the all features space with equally weighted dimensions. Sometimes the clustering process is doing simultaneously with cluster weighting. The updated weights are then used to update the clusters. The negative points of these approaches is that they requires multiple iterations for finding best clusters using the entire dataset, which could be computationally very expensive. In the last decade, in literature were proposed many Top-Down Subspace algorithms like PROCLUS, ORCLUS, FINDIT, $\delta - Clusters$, COSA, etc. One of the most know of these algorithms is the PROCLUS Subspace Clustering algorithm which is described bellow. COSA algorithm is also an interesting subspace algorithm which use an variable weighting technique to find the relevant subsets of the variables.

PROCLUS

PROCLUS (PROjected CLUStering) [4] was the first top-down subspace clustering algorithm which represent an extension of the *k*-medoid algorithm [25] in the subspace clustering. This algorithm selects a set of *k* medoids and iteratively improves the clustering.

The PROCLUS algorithm uses a three phase approach: the initialization, iteration, and the refinement phase. The first phase uses a greedy technique [4], [105], to samples the original dataset and to select potential medoids. The idea is that each cluster must be represented by at least one instance in the selected set. In the iteration phase, the algorithm iteratively compute the quality of the medoids in order to replace the bad medoids with the new ones. Cluster quality is based on the average distance between instances and the nearest medoid. After selecting the subspaces from each medoid, the average Manhattan distance is computed to assign the samples to the medoids and to form the clusters. The refinement phase compute new dimensions for each medoid and updates/reassigns the samples to the medoids based on new clusters.

COSA

Clustering On Subsets of Attributes (COSA) [40] is an iterative Top-Down Algorithm that assigns weights to each feature for each sample(instance). The algorithm uses a k nearest neighbors (*knn*) and start with a equally weighted variables. The computed neighborhoods are used to estimate the respective variables weights for each sample. Higher weights are assigned to those variables that have a smaller dispersion within the groups of the *knn*. This process is doing until the weights are stabilized. We show the algorithm in the procedure 6:

Algorithme 6 : COSA algorithm

Input: Data set X, n - number of samples, m - number of variables, λ - clustering exponential parameter
Initialize the weights $W = \frac{1}{m}; \eta = \lambda$
for $i = 1$ to n **do**
  Compute distances d_{ij}
  c $\longleftarrow$ clustering algorithm ($D_{ij}[W]$)
  compute weights $W = w_{l_1}^{L}$
  $\eta = \eta + \alpha\lambda$
end for
until W stabilizes
OUTPUT: the new clusters c

The distance $d_{ij}[W]$ between the samples i and j based on a weighted inverse exponential mean of $d_{ijk}{}_{k=1}^{n}$ with the scale parameter η is computing using the following expression:

$$D_{ij}^{(\eta)} = -\eta \log \left\{ \sum_{k=1}^{n} w_k exp(-\frac{d_{ijk}}{\eta}) \right\} \tag{1.48}$$

where η is the exponential parameter which allows to choose a larger variables set if the exponent is increasing or less variables set if η is decreasing. The value α from the procedure 6 has the goal to control the increasing rate in the value of the parameter η for a given value of λ.

This algorithms is a well-adapted for detecting relevant features set from a data, but due to its parameters became hardly to use for a non-expert user.

1.7.1.2 Bottom-Up Subspace Search Methods

The Bottom-Up search methods use the grids clustering approaches to cluster the data but firstly create a histogram for each variable and select the samples with a density above a given threshold. If there are dense samples for a k number of variables, the algorithm will form a subspace using these samples for the corresponding set of features reducing the search space. The disavantage of this type of methods is that obtaining good results depends on the proper tuning of the grid size and the density threshold parameters. These can be difficult to estimate, even to set, since they are used across all of variables in the dataset. In literature we find some bottom-up subspace methods like CLIQUE, ENCLUS, MAFIA, DOC, Cell Based Clustering (CBF), etc [5], [24], [50], [22]. Some of these algorithms use a static grid

to divide each variable into bins (CLIQUE and ENCLUS), and some of them use data driven strategies to determine the cut points for the bins on each variable.

1.7.2 Techniques for Feature's space selection

Dimensionality reduction techniques can be divided into the follows categories [69, 12]:

- Feature selection and feature weighting.
- Feature extraction.
- Feature grouping.

1.7.2.1 Feature Selection and Feature Weighting

Features selection [56] is a process used in the machine learning applications, where a subset of features are selected for the data for application of a learning algorithm. This is an important stage of pre-processing and is a solution to escape the course of dimensionality. On other hand, feature weighting, assigns weights to different variables to indicate if a variable is most important than another. Most of feature selection/weighting algorithms are adapted to supervised learning.

There are three types of approaches:

- Filters Approaches;
- Wrappers Approaches;
- Embedded Approaches;

Filter methods

The filter based algorithms is weighting the feature in the pre-processing using the dataset alone; most of them are supervised.

One of the supervised selection based algorithm is the Constraint Score Algorithm [140]. Given a dataset $X = x_1, x_2, ..., x_m$, the supervision information *must-link (ML)* and *cannot-link (CL)* are the follows:

$$ML = (x_i, x_j)|x_i \ and \ x_j \ belong \ to \ the \ same \ class \tag{1.49}$$

and the *CL* constraints:

$$CL = (x_i, x_j)|x_i \ and \ x_j \ belong \ to \ different \ classes \tag{1.50}$$

Let f_{ri} to be the feature r of the i-th example, and, to evaluate the score of the r^{th} feature using the constraints CL and ML there are two different score functions to be minimized:

$$C_r^1 = \frac{\sum_{(x_i,x_j)\in M}(f_{ri} - f_{rj})^2}{\sum_{(x_i,x_j)\in C}(f_{ri} - f_{rj})^2} \tag{1.51}$$

$$C_r^2 = \sum_{(x_i,x_j)\in M}(f_{ri} - f_{rj})^2 - \lambda \sum_{(x_i,x_j)\in C}(f_{ri} - f_{rj})^2 \tag{1.52}$$

where λ is a regularization coefficient, which balance the contributions of the two terms in the equation. Usually $\lambda < 1$.

Algorithme 7 : Constraint Score

Input: Data set X, the constraints M and C, λ

for $i = 1$ to n features **do**

compute the constraints M and C

end for

Rank the features according to their constraint scores in ascending order.

Output: The ranked features list

An example of unsupervised feature selection based algorithm is the Laplacian Score (LS) proposed by Xiaofei He et all in 2004 [59].
Using this approach, a reasonable criterion for choosing a relevant feature is to minimize the following objective function:

$$L_r = \frac{\sum_{ij}(f_{ri} - f_{rj})^2 S_{ij}}{Var(f_r)} \tag{1.53}$$

where $Var(f_r)$ represent the variance of the feature r. The Laplacian Score algorithm is defined in the procedure 8.

Algorithme 8 : Laplacian Score

for $i = 1$ to n features **do**
  Construct a nearest neighbor graph G with m modes
  if x_i and x_j are close, i.e. x_i is among k nearest neighbors of x_j **then**
    One put an edge between the node i and j
  end if
  if i and j are connected **then**
    Put $S_{ij} = e^{-\frac{\|x_i - x_j\|^2}{t}}$, where t is a constant
  else
    if Otherwise **then**
      $S_{ij} = 0$
    end if
  end if
  Compute Laplacian graph for the feature r:

$$f_r = [f_{r1}, f_{r2}, ..., f_{rm}]^T, \tag{1.54}$$

$$D = diag(S\,1), \quad 1 = [1, ..., 1]^T, \tag{1.55}$$

$$L = D - S\,; \tag{1.56}$$

  Compute the Laplacian Score of the r-th feature:
  Let:

$$\tilde{f}_r = f_r - \frac{f_r^T D 1}{1^T D 1} 1 \tag{1.57}$$

$$L_r = \frac{\tilde{f}_r^T L \tilde{f}_r}{\tilde{f}_r^T D \tilde{f}_r} \tag{1.58}$$

end for

Wrappers Approaches

One of the wrapper feature selection for estimation by the kNN algorithm is the Regression, Gradient guided, feature Selection algorithm (*RGS*) proposed by Amir Navot et. al [99]

which have the goal to find subsets of features that induce a small estimation error in a supervised mode. The RGS algorithm involves a feature weighted version of the k-nearest-neighbor algorithm. The evaluation function which scores weight vectors (w) using a kNN is the follows:

$$e(w) = \frac{1}{2}\sum_{x\in X}(f(x) - \hat{f}_w(x))^2 \tag{1.59}$$

where $\hat{f}_w(x)$ is the value assigned to x by the weighted kNN estimator, defined as following:

$$\hat{f}(x) = \frac{1}{Z}\sum_{x'\in N(x)} f(x')e^{-d(x,x')/\beta} \tag{1.60}$$

where $d(x, x') = \|x - x'\|_2^2$ is the l_2 norm, $Z = \sum_{x'\in N(x)} e^{-d(x,x')/\beta}$ is a normalization factor with a fixed β parameter, and $N(x) \subset X$ represent the set of the k nearest points to x in dataset X.

Algorithme 9 : RGS: Regression, Gradient guided feature Selection

Initialization: $w = (1, 1, ..., 1)$

for $t = 1$ to T **do**

Choose randomly a distance x from X

Compute the gradient of $e(w)$:

$$\nabla e(W) = -\sum_{xinX}(f(x) - \hat{f}_w(x))\nabla_w \hat{f}_w(x) \tag{1.61}$$

$$\nabla_w \hat{f}_w(x) = \frac{-\frac{4}{\beta}\sum_{x'',x'\in N(x)} f(x'')\alpha(x', x'')u(x', x'')}{\sum_{x'',x'\in N(x)} \alpha(x', x'')} \tag{1.62}$$

where $\alpha(x', x'') = e^{-\|x-x'\|_w^2+\|x-x''\|_w^2/\beta}$

and $u(x', x'') \in \Re^n$ is a vector with $u_i = w_i[(x_i - x_i')^2 + (x_i - x_i'')^2]$

Update:

$w = w + \eta_t \nabla_e(w) = w(1 + \eta_t \nabla_w \hat{f}_w(x))$

where η_t is a decay factor.

end for

Embedded Algorithms

This type of algorithms represents techniques which modify the learning algorithm in order to have the ability to perform feature selection. There is no an explicit feature selection step before/after the clustering, but the algorithm automatically builds a classifier with a small number of features. This kind of feature selection algorithms are similar to the subspace clustering. One of interesting embedded approach is the LASSO algorithm (Least Absolute Shrinkage and Selection Operator) [79], [120].

For a given loss function l, in order to find the parameter β, LASSO minimizes the empirical risk by including a constraint on the weight coefficients:

$$R_{LASSO}(\beta_0, \beta) = \sum_{i=1}^{n} l(y_i, \beta_0 + \sum_{l=1}^{d} x_{il}\beta_l') \tag{1.63}$$

subject to $|\beta_l| \leq \lambda_l$ for $l = 1, ..., d$.

The pairs $(y_1, x_1), ..., (y_n, x_n)$ represent the input/output pairs where the output are $y_i \in Y$ and the input : $x_i \in X$ respectively.
One note that $X = X_1 \otimes X_2 \otimes ...X_d$ and X_l are subsets of $\Re^{pl}$, $l = 1, ..., d$. Finally, $x_i = (x_{i1}, ..., x_{id})$ where $x_{il} \in X_l$, and $\beta = (\beta_{i1}, ..., \beta_{id})$ are the regression coefficients where $\beta_l \in \Re^{pl}$.

There are also another examples of embedded feature selection algorithms like MARS (multivariate adaptive regression splines) [41], [115] which performs the variables selection by choosing the variables used in the polynomial spline; or the ESFS approach (Embedded feature selection method based on SFS) and SFS - Sequential Forward Selection [134], etc. The automatic detection of the relevance in the neural networks is an another example of an embedded approach which uses a Bayesian method to estimate the feature weights [128], [130].

1.7.2.2 Feature Extraction

Feature extraction algorithms construct a small set of new features using a mapping technique from a high dimensional data. There are some proposed well-known algorithms for the unsupervised techniques like the PCA (Principal Components Analysis) or the Factor

Analaysis (FA) [28]; and PLS (Partial Least Squares) [20], [101] or LDA (Linear Discriminant Analysis) [138], [6] for the supervised learning. In this section, we describe the unsupervised technique for the feature selection, especially the PCA which is used in the next chapters of this work.

PCA: Principal Component Analysis

The PCA algorithm is probably the most known feature extraction technique which allows to find a hyperplane such that, upon projection to this one, the data variance is best preserved. The PCA algorithm is doing in 4 steps:

Algorithme 10 : Principal Component Analysis

Input: Data set X

for $i = 1$ to n features **do**

Substract the mean matrix M from the data;

Compute the standart deviations from the mean (sd);

Compute the covariance matrix (C);

Calculate the eigenvector and eigenvalues of the covariance matrix:

$\mathbf{V}^{-1}\mathbf{CV} = \mathbf{D}$

Rearrange the eigenvectors and eigenvalues;

Sort the columns of the eigenvector matrix V and eigenvalue matrix D in a decreasing order;

Compute the cumulative energy content for each eigenvector:

$g[m] = \sum_{q=1}^{m} D[p,q]$ for $p = q$ and $m = 1, \ldots, M$

Output: Select a subset of the eigenvectors;

end for

We use this technique to do the data and cluster analysis in order to compare them this our results (Chapter 3).

1.7.2.3 Feature grouping

The idea of the feature grouping approaches is to combine several existing features in order to obtain a new set of relevant and correlated features. The most direct way to attempt feature

grouping is to cluster the features using a clustering method on the features (to transpose the initial dataset).
One of the most used features grouping method is the Varclus technique available on the SAS/STAT procedure [100]. This variable selection method is doing in two phases. The first is a nearest component sorting phase where cluster components are computed for each iteration and each variable is assigned to the component which has the highest squared correlation. The second phase associates a search algorithm where each variable is tested to see if assigning it to an another features cluster increases the amount of variance explained. There are some types of feature grouping techniques which were proposed like: Bi-clustering, Co-clustering, Double-clustering, Coupled-Clustering and Simultaneous clustering which have the same goal to do the clustering and the variables selection simultaneously.

1.8 Weighting approaches for the clustering algorithms

Almost all the classical algorithms have situations when they doesn't work well for a given dataset, especially in a high dimensional space, like: the density-based clustering approaches will give a wrong result for the density-unseparated data; the center-based algorithms - if all the centers computed in a dataset are close; the relational analysis techniques - if the relation of each pairwise are very similar between them. For all these types of algorithms, in literature were proposed weighting approaches adapted for them: GDBSCAN for the density-based approaches, ω-k-means - k-means type algorithm, weighted Condorcet criteria - for the Relational Analysis, etc. Weighting approaches allow to adjust the data for the respective clustering algorithms during the learning process. In the next sections we will introduce some weighting approaches which are similar to our work.

1.8.1 wLVQ2

One of the interesting weighting approach in the supervised learning is the ωLVQ2 which is an extension of the Learning Vector Quantization (LVQ) proposed by Y. Bennani in 1999 [16]. The ωLVQ2 technique is an adaptive weighting approach and it is based on weighting the features depending on their contribution to discrimination.
This approach is doing the feature weighting and the classification simultaneously as it is shown in the figure 1.10. The first layer of this schema compute the weights and in the second layer the standard LVQ2 is applied.

The objective function for this problem for a training vector x which belong to the class

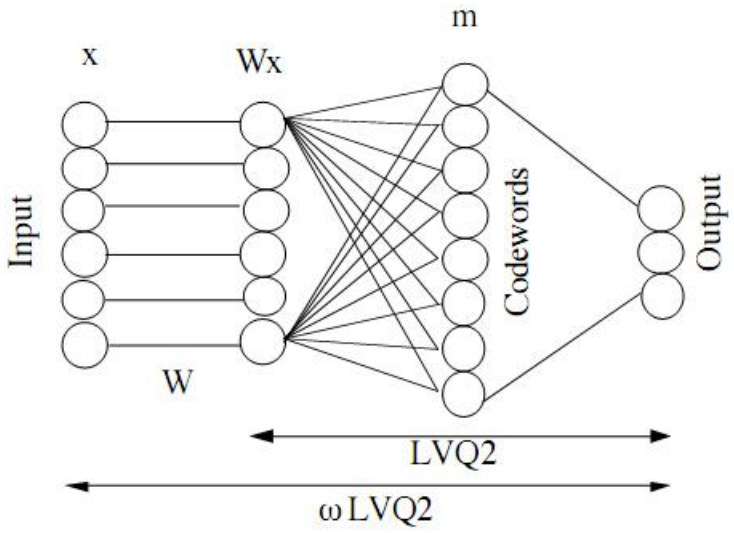


Figure 1.10: Weighting Learning Vector Quantization (figure taken from [16])

C_k, and the nearest codeword m_i which belong to the class C_i is the following :

$$R_{\omega LVQ2}(x, m, W) = \begin{cases} \|W_x - m_j\|^2 - \|W_x - m_i\|^2 & if\ C_k = C_j \\ 0, & otherwise \end{cases} \tag{1.64}$$

Where W is the weighting coefficients matrix, m_i is the nearest codeword vector to W_x and m_j is the second nearest codeword vector to W_x from the class C_j.

The adaptation rules for this problem are the following:

$$\omega LVQ2 \begin{cases} m_j(t+1) = m_j(t) + \alpha(t)(Wx(t) - m_j(t)); \\ m_i(t+1) = m_i - \alpha(t)(Wx(t) - m_j(t)); \\ \omega_k^k(t+1) = \omega_k^k(t) - \beta(t)x^k(t)(m_i^k(t) - m_j^k(t)) \quad fork = 1...n; \\ m_k(t+1) = m_k(t) \quad for\ the\ rest\ of\ codewords. \end{cases} \tag{1.65}$$

Where $\alpha(t)$ and $\beta(t)$ usually are chosen as positive reals decreasing within each iteration t; and the codewords could be initialized with a clustering algorithm.
We will use this model as a start point for our proposed adaptive weighting SOM, to weight the variables and the samples/observations during the learning of the map.

1.8.2 The *w*-k-means algorithm

In order to detect the relevant variables during the learning of the k-means algorithm, Huang and al. [65] proposed a k-means type algorithm which automatically compute the variable weights.

Let $W = [w_1, w_2, ..., w_m]$ be the weights for m variables and β be a parameter for the variable weight w_j, the classical objective function of the k-means algorithm is modified as:

$$R_{w-kmeans}(U, Z, W) = \sum_{l=1}^{k} \sum_{i=1}^{n} \sum_{j=1}^{m} u_{i,l} w_j^{\beta} d(x_{i,j}, z_{l,j}) \tag{1.66}$$

subject to

$$\begin{cases} \sum_{l=1}^{k} u_{i,l} = 1, \ \ 1 \leq i \leq n \\ u_{i,l} \in 0, 1, \ \ 1 \leq i \leq n, \ \ 1 \leq l \leq k \\ \sum_{j=1}^{m} w_j = 1, \ \ 0 \leq w_j \leq 1 \end{cases} \tag{1.67}$$

Similar to solving the minimization of the objective function for the classical k-means algorithm, the minimization of the equation 1.66 is done by iteratively solving the following 3 minimization problems:

1) Problem P1 : Fix $Z = \hat{Z}$ and $W = \hat{W}$, solve the reduced problem $R_{w-k-means}(U, \hat{Z}, \hat{W})$;

2) Problem P2 : Fix $U = \hat{U}$ and $W = \hat{W}$, solve the reduced problem $R_{w-k-means}(\hat{U}, Z, \hat{W})$;

3) Problem P3 : Fix $U = \hat{U}$ and $Z = \hat{Z}$, solve the reduced problem $R_{w-k-means}(\hat{U}, \hat{Z}, W)$.

Problem P1 is solved using the next expression:

$$\begin{cases} u_{i,l} = 1 \ \ if \ \ \sum_{j=1}^{m} w_j^{\beta} d(x_{i,j}, z_{l,j}) \leq \sum_{j=1}^{m} w_j^{\beta} d(x_{i,j}, z_{t,j}) \ \ for \ \ 1 \leq t \leq k \\ u_{i,t} = 0 \ \ for \ \ t \neq l \end{cases} \tag{1.68}$$

Problem P2 is solved in the same manner like for the classical k-means algorithm.

The solution of the problem P3 is finding using the next theorem:

Theorem2.1. Let $U = \hat{U}$ and $Z = \hat{Z}$ be fixed.

- 1. When $\beta > 1$ or $\beta \leq 0$, $R_{w-k-means}(\hat{U}, \hat{Z}, W)$ is minimized if:

$$\hat{w}_j = \begin{cases} 0 & if D_j = 0 \\ \sum_{t=1}^{h} [\frac{D_j}{D_t}]^{\frac{1}{\beta-1}} \end{cases} \tag{1.69}$$

  where

$$D_j = \sum_{l=1}^{k} \sum_{i=1}^{n} \hat{u}_{i,l} d(x_{i,j}, z_{l,j}) \tag{1.70}$$

  and h is the number of variables where $D_j \neq 0$.

- 2. When $\beta = 1$, $R_{w-k-means}(\hat{U}, \hat{Z}, W)$ is minimized if:
  $\hat{w}_{j'} = 1$ and $\hat{w}_j = 0$, $j \neq j'$, where $D_{j'} \leq D_j$ for all j.

The complexity of the *w*-k-means increase by adding the weighting step during the learning process. For one iteration, the computational time is $O(mnk)$, where k is the number of clusters, m is the number of variables, and n is the number of observations.

1.8.3 SCAD

Frigui and al. proposed to weight the k-means algorithm using global and local weights. The authors propose two types of weighting approaches: SCAD1 and SCAD2 which are very similar to the w-k-means algorithm [42].

1.8.3.1 SCAD1

The SCAD1 algorithm minimizes the following objective function:

$$R_{SCAD1}(C, U, V; \chi) = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^{m} \sum_{k=1}^{n} v_{ik} d_{ijk}^{2} + \sum_{i=1}^{C} \sigma_i \sum_{k=1}^{n} v_{ik}^{2} \tag{1.71}$$

subject to the constraints:

$$u_{ij} \in [0, 1] \forall i; \quad 0 < \sum_{j=1}^{N} u_{ij} < N \ \ \forall i, j; \quad \sum_{i=1}^{C} = 1 \ \forall j \tag{1.72}$$

$$v_{ik} \in [0, 1] \forall i, k; \quad and \ \sum_{k=1}^{n} v_{ik} = 1, \ \ \forall i. \tag{1.73}$$

and

$$d_{ijk} = |x_{jk} - c_{ik}| \tag{1.74}$$

where x_{jk} is the value of the feature k of data point x_j, and c_{ik} is the k^{th} component of the center vector i of the cluster. Intuitively, d_{ijk} represent the projection of the displacement vector between x_i and the class center c_i along the k^{th} dimension.
The optimization of this objective function is doing exactly as for $w-k$-means.

1.8.3.2 SCAD2

The SCAD2 algorithm uses a regularization term in the objective function that results in a tunable feature discrimination parameter. So, the objective function with the discriminant exponent q is the following:

$$R_{SCAD2}(B, U, V; \chi) = \sum_{i=1}^{C} \sum_{j=1}^{N} (u_{ij})^m \sum_{k=1}^{n} (v_{ik})^q d_{ijk}^2 \tag{1.75}$$

subject to 1.73.

1.8.4 Analytical global weighting SOM (w-SOM)

The algorithm $w-SOM$ [119] is an approach of weighting features which is based on the optimization of a weighting function which makes it possible to evaluate the relevance of features in dataset. Obtained weights makes possible to order the variables according to their relevance and these weights can be used like criterion of evaluation in an feature selection [53],[119].

The approach $w-SOM$ is based on $w-k$-means algorithm proposed by Huang [65] which optimizes the following cost function:

$$R_{w-SOM}(U, Z, W) = \sum_{i=1}^{N} \sum_{j=1}^{n} \sum_{k=1}^{C} u_{ik}\, w_j^{\beta}\, (x_{ij} - z_{kj})^2 \tag{1.76}$$

- $U = (u_{ik})$ is a binary matrix with $\begin{cases} 0, & si\ x_i \notin C_k \\ 1, & si\ x_i \in C_k \end{cases}$

- $W = (w_1, w_2, ..., w_n)$ is the vector which gathers the weight of the various attributes,

- $Z = z_k \in R^n : k = 1, ..., C$ is the whole of the centers.

The extension of the w-k-means to the self-organizing maps provides to optimize the following cost function :

$$R_{w-SOM}(U, Z, W) = \sum_{k=1}^{C} \sum_{i=1}^{N} \sum_{j=1}^{n} u_{ik}\, w_j^{\beta} \sum_{l=1}^{C} h_{kl}\, (x_{ij} - z_{lj})^2 \tag{1.77}$$

$$D_j = \sum_{k=1}^{C} \sum_{j=1}^{n} \hat{u}_{ik} \sum_{l=1}^{C} h_{kl}\, (x_{ij} - \hat{z}_{lj})^2 \tag{1.78}$$

where h_{kl} is the the neighborhood function between the prototypes.
This weighted algorithm allows weighting the SOM map (prototypes) in a global mode, and respectively can't provide a local feature information.

1.9 Conclusion

This chapter presented an introduction to the clustering research field and presented the most used types of clustering approaches. One of the important conclusion is that there is no clustering algorithm which solves all problems for all the data types. Also, usually the clustering methods must be able to adapt itself to the learning data by using some weighting or transformation techniques as GDBSCAN, weighted Condorcet criterion, etc.
In the context of clustering, the proposed approaches in the literature show:

- Variables weighting allows for better classification results;
- Local weighting of variables is more efficient than global weighting;
- Adaptive weights are more effective than analytic weights.

An important problem that clustering methods need to deal with is the stability and adaptation dilemma. A learning schema should have the capability to maintain stable cluster structure during the entire clustering/learning process. Neural network–based clustering is tightly related to the concept of competitive learning. In the next chapters we will introduce some extensions of the classical SOM which will take into consideration the lose of the information during the learning process.
Another aspect of the clustering problem is the visualization which is not possible for the most presented techniques. This is why, we give a big attention to the SOM algorithm which allow to do the clustering, dimensionality reduction and the visualization at the same time. The related work shown that introducing the topology assumptions during the learning process allow to attempts a clustering visualization. An interest and new family of the clustering methods is the Relational Analaysis type clustering which are well-adapted to the binary data and resolve problems with a very low complexity and a high quality. One of the disadvantage of this method is that it doesn't allow the clustering visualization, because, contrarily to the SOM approach, RA doesn't take into the consideration any topological criterion. In the chapter 3 we propose an adaptation of the RA to the problem of the topological learning.
We therefore wish to develop new algorithms which will be able to treat all these problems using topological clustering technique.

Chapter 2

Memory based Weighted Topological Clustering

2.1 Introduction

As discused in the chapter 1, we are interested in the unsupervised learning problem, and more precisely in the clustering analysis. We use Kohonen's Self-Organizing Maps [75] as basic model for our approaches, which is one of the most used connectionism methods among the unsupervised learning methods. This network has given rise to numerous practical applications in order to visualize and perform the information reduction.
At the end of this topographic learning, the "similar" data will be collect in clusters, which correspond to the sets of similar patterns. These clusters can be represented by a more concise information than the brutal listing of their patterns, such as their gravity center or different statistical moments. As expected, this information is easier to manipulate than the original data points.
In this unsupervised learning algorithm, the process of self-organization helps to focus the adaptation of the connections weights on the most "active" map's region. This activity area is chosen as the area associated with the cell whose state is most active. During the competition, the criterion for selecting the most active unit is to attempt the one whose weight vector is closest from the sample at a given moment using Euclidean distance. This choice does not take into account the competition history of each unit during learning. This is a criterion, which is currently used in several variations of the topological maps algorithm. The approach proposed in this work proposes to introduce a memory effect of each unit during the learning process. This procedure is used in the process of unsupervised learning, at the competition stage. This memory effect will take into account the competition history of each unit during

the learning process. This new strategy uses information from various interactions of units with learning examples.
Our approach, called vm-SOM[1] increases the interaction between two concepts: memory and learning, so closely linked that often confuses the two. These two concepts, however, have reference to different phenomena. Learning refers to a process that will change a later behavior. Memory is the ability to remember past experiences. Moreover, not only the memory depends on learning, but learning also depends on memory. Indeed, the stored knowledges form a frame on which is joined the new knowledges.
In this chapter, we propose a new learning strategy (vm-SOM) for classification algorithms based on topographical self-organizing maps. This new strategy consists in the introduction of a memory process into the competition phase by calculating a voting matrix at each learning iteration. This model allows us to minimize the loss of information in the previous iterations of the neighborhood results; especially when one have a large-scale learning with several hundreds of iterations and a large dataset.

Another important aspect of the clustering analysis/characterization is the features selection is difficult in a high dimensional space.
One of drawbacks of the traditional SOM algorithms is that they treat all variables equally. This is not desirable in many applications of clustering where observations are described with a large number of variables. A cluster provided by SOM is often characterized by a subset of variables rather to entire variables set. Hence other variables can obscure the discovery of the specific cluster structure associated to each cell. The relevance of each variable change from cluster to another. Thus, *how to compute the variables relevances or weights automatically during SOM learning process ?* Variables weighting is an extension of the variables selection procedure where the variables are assigned continuous weights which can be considering as degrees of relevance [2].

The proposed approaches to perform SOM clustering and variables weighting is designed to search for the optimal prototypes, and the optimal set of features weights, simultaneously. Each prototype $\mathbf{w}_j = (w_j^1, w_j^2, ..., w_j^d)$ associated to cell j is allowed to have its own set of local features weights $\pi_j = (\pi_j^1, \pi_j^2, ..., \pi_j^d)$. We denote by $\Pi = \{\pi_j, \pi_j \in \Re^d\}_{j=1}^{|\Pi|}$ the set of weight vectors ($|\Pi| = |W|$).

[1] voting memory based Self-Organizing Map

2.1.1 In time activation during Toplogical Clustering

Chappell and Taylor [23] proposed a topological clustering model called the Temporal Kohonen Map (TKM) changing the classical SOM using a LIN-type[2] STM[3] mechanism. The TKM maintains the activation history of each neuron i by means of variable $z_i(t)$, called the leaky integrator potential:

$$z_i(t) = \lambda z_i(t-1) - (\frac{1}{2})\|x(t) - w_i(t)\|^2 \tag{2.1}$$

where $0 \leq \lambda \leq 1$ is the memory depth constant. An output neuron in the TKM stores activity at time t, after which this activity fades at a rate given by the time constant $|\frac{1}{\lambda}|$ [23]. The wining neuron is chosen according to the following adaptation step:

$$z_{bmus}(t) = max z_i(t) \tag{2.2}$$

The update step for the weights $w_i(t)$ and of its neighbors is the same as in the classical SOM algorithm.

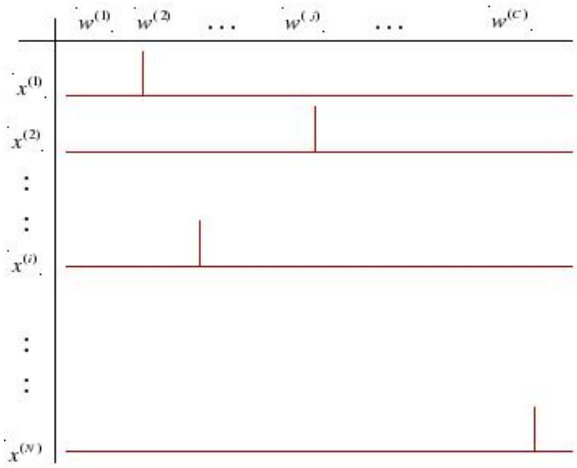
The TKM was used to detect the temporal (sequences) data, where the use of classical SOM will favor erroneous results because contrarily to the TKM, the determination of the winning neuron in the classical SOM depends mainly on the most recent inputs. In the following section we propose to use another schema for preserving the activation behavior in the topological clustering algorithms, and we propose an adaptation of it to the SOM called voting memory self-organizing map (vm-SOM).

2.2 vm-SOM : Topographic map with Voting Memory

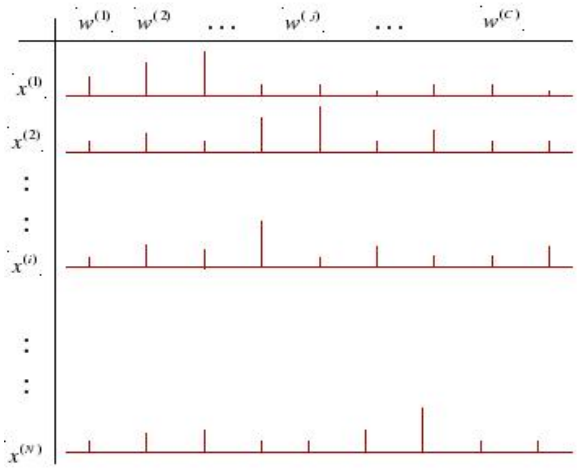
The use of the neighborhood concept introduces topological constraints in the final geometry of the map. Topological quality of the final map depends strongly on the construction mechanism. It is therefore important to enhance this mechanism by introducing a competition memory effect from various units of the map. This memory can take into account the reactions history of each map's cell with learning examples. The concept of voting memory will be introduced in the learning algorithm as a voting matrix VM (Voting Memory).

[2]LIN - Leaky Integrators Neurons
[3]STM - Short-term memory

This principle is based on a method proposed by Dimitrios et al. [43] for a clustering fusion scheme. This new voting matrix will give the possibility to weight the most active samples during the learning process taking into account the adaptation history.



(a) SOM



(b) vm-SOM

Figure 2.1: Voting matrix (on the top - classical SOM; vm-SOM on the bottom)

The stochastic version of the learning algorithm vm-SOM takes place in three phases: **the initialization phase**, where random values are assigned to the connections weight (referents or prototypes) of each map unit - initialization of voting memory matrix VM size NxC, where each learning instance is associated with a unit of the map: one whose weight vector is closest using the Euclidean distance; **the competition phase** during which, a unit of the SOM map is selected as a winner, for all input patterns which is one of the highest vote val-

ues (matrix VM); **the adaptation phase** where the weight of each cell in the map is updated according to adaptation rules and votes updating. This adjustment process is repeated until stabilization of self-organization.

The vm-SOM algorithm is therefore as follows:

1) Initialization phase :

   1.1) Define the SOM map structure; initialize randomly the prototypes $w^{(j)}$.

   1.2) Initialize the voting memory matrix VM (NxC):
   $for \quad k = 1, ..., N$
   $for j = 1, ..., C$

$$VM_{t=1}(k, j) = \frac{e^{-\|x^{(k)} - w_t^{(j)}\|^2}}{\sum_{i=1}^{C} e^{-\|x^{(k)} - w_t^{(j)}\|^2}} \tag{2.3}$$

2) Competition phase :

   2.1) $t = 2$;

   2.2) Provide a learning example $x^{(k)}$ to SOM;

   2.3) Select the cell whose prototype is the one with the highest vote's value in VM matrix (BVU : Best Vote Unit):

$$\phi(x^{(k)}) = \arg \max_{1 \leq j \leq C} (VM_{t-1}(k, j)) \tag{2.4}$$

   where ϕ assigns each example $x^{(k)}$ to a single cell of the SOM with the highest vote value in VM.

3) Adaptation phase :

   3.1) Updating prototypes using the rule :

$$w_t^{(j)} = w_{t-1}^{(j)} - \varepsilon_{t-1} K_{j\phi(x^{(k)})} \left(x^{(k)} - w_{t-1}^{(j)} \right) \tag{2.5}$$

   where K_{ij} is defined as following:

$$K_{ij} = \frac{1}{\lambda(t)} exp\left(-\frac{d_1^2(i, j)}{\lambda^2(t)} \right) \tag{2.6}$$

3.2) Update of voting matrix VM :

$$\begin{cases} VM_t(k, \phi(x^k)) = \dfrac{(t-1)}{t} VM_{t-1}\left(k, \phi(x^{(k)})\right) + \frac{1}{t} \\ \\ VM_t(k, j) = \dfrac{(t-1)}{t} VM_{t-1}(k, j),\ j = 1, ..., C,\ j \neq \phi(x^{(k)}) \end{cases}$$

4) Repeat steps 2 and 3 until stabilization of self-organization or up to $t = t_{max}$, where t_{max} is the maximum number of iterations.

The updating of the Best Vote Unit allows us to give more importance to the samples which were active during the learning process ($\frac{1}{i}$) and less importance if the samples are passives during the learning. This technique allows us to construct a map accordingly to the history of all the neurons adaptation. This process establishes an assignment principle, during which the size of voting matrix VM is updated during learning where $VM_t(i, j)$ represents the membership degree of the observation i in unit j at time t. At the end of learning, this VM matrix can be seen as a new data encoded in the prototypes space. Each learning example is thus, encoded as a vote's vector of various units of the map. This is a data vector quantization in vector of votes. Figure 2.1 shows the voting matrix in the case of classical SOM (binary vector for each example) and in the case of vm-SOM (vector with multiple real values).

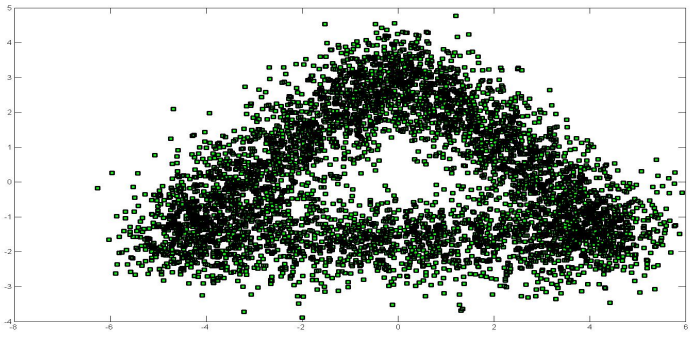
2.2.1 Experimental protocol

To validate our approach, we used two indicators: the quantization error (distortion) and topographic error. We tested our method with several maps of different sizes. To prove the importance and influence of competitive strategy proposed by our approach for final map segmentation, we calculated various indices of quality, as the purity index and the Rand index [68]. To check the stability of the proposed method, we conducted 15 experiments on several datasets: weveform, wdbc, isolet, madelon and spambase. More details for these datasets are given in the Appendix A.1.

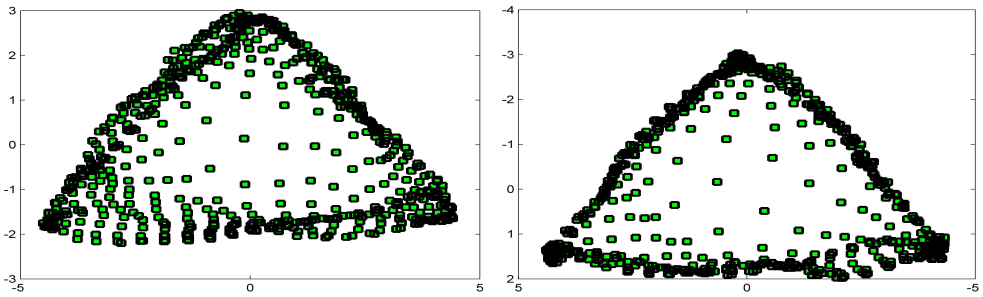
One of the most used criteria to assess the quality of a topological map is the quantization error. This error measures the average distance between each data vector and its winning neuron (BMU: Best Match Unit). One can also measure the quality of the map by estimating the topological preservation. The most used index to quantify this information is the topographic error, which was proposed by Kohonen ([75], [74], [8], [112]). This error measures the proportion of all examples for which the first and second winning neuron (BMU) are not adjacent vectors. The idea of this index is to show how the granularity SOM map can preserve the topology of the data. To evaluate the quality of the map segmentation we used the Rand and purity indices.

2.2.1.1 Illustration of the vm-SOM approach on an example: waveform

We use this dataset to show the topological structure of the dataset and of the map with the classical SOM approach and the proposed vm-SOM approach. Learning a map size 25x25 for all observations can provide a map composed of 625 neurons (units). Then, we use principal component analysis (PCA) on the two obtained maps to compare visually the structure of these two maps (SOM and vm-SOM). Part (a) of Figure 2.2 shows the data distribution, part (b) of the figure illustrates the map structure after the classical SOM algorithm, and part (c) of the same figure represents the distribution of neurons using the vm-SOM (voting memory).



(a) Data distribution of waveform dataset



(b) Neurons distribution using classical SOM (c) Neurons distribution using vm-SOM

Figure 2.2: Map structure using classical SOM algorithm and vm-SOM

On part (b) of this figure, we can observe that the neurons of the SOM map are mainly scattered at the intersection of classes. Opposite by using learning with memory vm-SOM, the structure of neurons (Fig.2.2 (c)) is more concentrated. Indeed, with vm-SOM, the topographic error decline strongly, as well as the quantization error. By better preserving the topology of the data, our approach also allows a better segmentation of the map compared to classical SOM. Generally, the use of topological maps is followed by a clustering of prototypes. Often, this segmentation is limited to the use of hierarchical classification algorithm

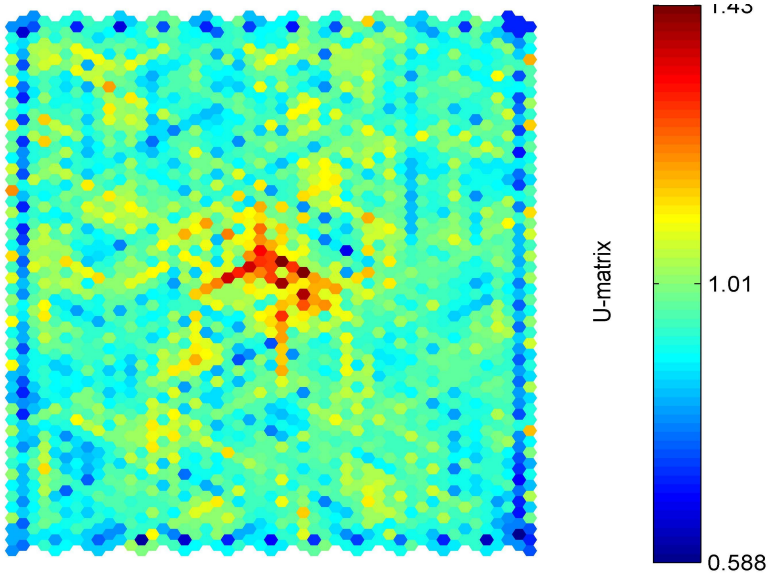
or *k*-means combined with a quality index to determine the size of the ideal map partition. In our case, we used the *k*-means algorithm combined with the Davies-Bouldin internal quality index [131], and hierarchical classification (AHC) [38]. For this map segmentation (form groups of cells) should be used the similarities between the cells. To do this, we calculate the unified distance matrix (U-matrix) which computes the distances between neighboring cells of the map.
In Figure 3 the different shades of colors indicate the distances between the different prototypes of the map, the more red (dark) - prototypes are more distant. By observing the structure of the waveform dataset, there is a better distribution of the prototypes on the unified distances matrix provided by our approach vm-SOM (figure 2.3 (b)) compared to classical results of SOM (figure 2.3 (a)).

To prove the stability of the vm-SOM approach, we calculated the quantization errors and topological errors on 15 maps of varying sizes (from 10x10 to 25x25) for the waveform dataset. Figure 2.4 shows the standard deviations of errors with the classical SOM algorithm and vm-SOM. From all experiments, we observe a strong decrease in the topographic error in the case of vm-SOM compared to classical SOM with an improved distortion. We note that for the both methods one fix the same parameters (map size, number of iterations,...).
To quantify the quality of clustering, we use Rand and purity indexes. Furthermore, during the learning process and the segmentation, the labels are not used. In figure 2.5 we can observe an improvement in the segmentation of the map with our approach vm-SOM through better conservation of the topology obtained after learning.

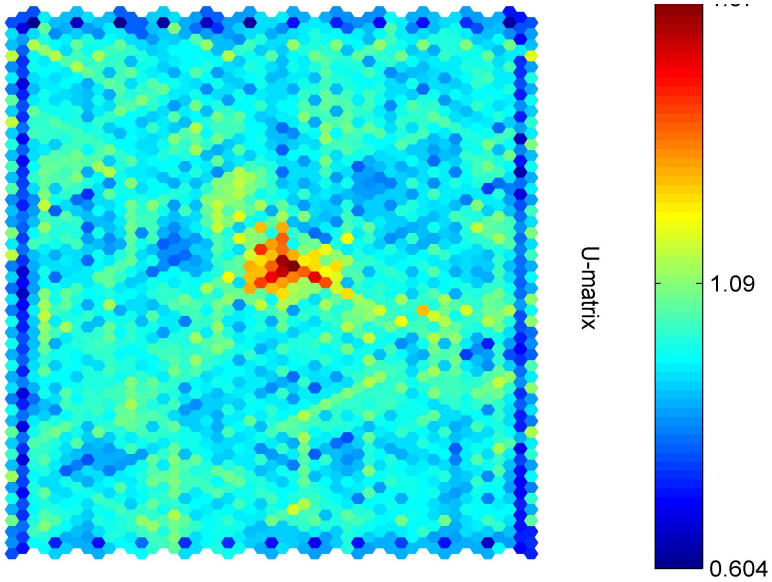
2.2.1.2 Validation of vm-SOM approach on other datasets

We used the experimental protocol for other datasets and all indices with standard deviation that are presented in Table 2.1. On all datasets, we can observe a significant decrease in the topographic error (Table 2.1). This shows that we obtain more relevant and meaningful maps with our vm-SOM approach. Indeed, the segmentation of the maps obtained with vm-SOM provides better results due to the affectation strategy of employment-based voting memory during learning.

The analysis of these experiments results shows some characteristics of vm-SOM and some comparisons between the classical SOM and vm-SOM:
- The voting memory used during the learning process at the competition phase will keep all information about the neighborhood and various interactions between the learning examples and prototypes; and promotes the preservation of the maps topology. Indeed, unlike the classical SOM algorithm, which only takes into account the previous iteration, vm-SOM enables us to take into account the competition history of each prototype during learning.



(a) SOM



(b) vm-SOM

Figure 2.3: U-matrices: classical SOM (on the left) and vm-SOM (on the right)

- The voting memory matrix VM can be used as a new encoding data in a conceptual space: the prototypes space.
- This memory can be easily integrated with other unsupervised learning approaches.
- Analyzing the standart deviation (Figure 2.4) we can see that the proposed method increase the stability of the results.

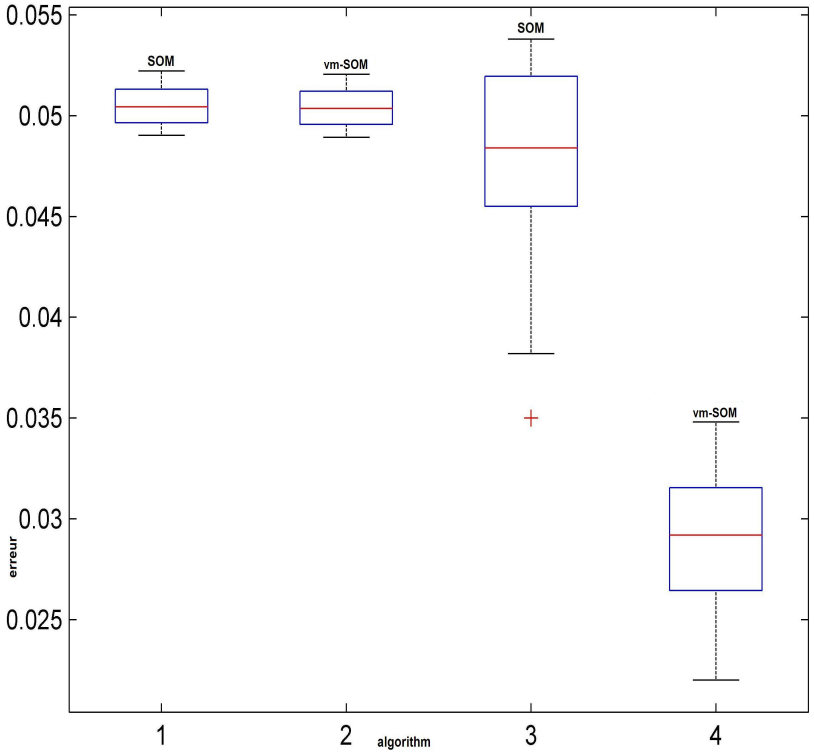


Figure 2.4: The quantization and topological errors on the waveform dataset

Table 2.1: Validation results of SOM and vm-SOM on different datasets

Index	Method	Dataset			
		Wdbc	Isolet	Madel.	Spam
Quantiz. error	SOM	2.607	17.444	21.746	4.128
	vm-SOM	2.593	17.308	21.672	3.923
Topolog. error	SOM	0.029	0.031	0.009	0.079
	vm-SOM	**0.015**	**0.007**	**0.002**	**0.063**
Rand kmeans	SOM	0.561	0.922	0.523	0.535
	vm-SOM	0.565	0.934	0.527	0.547
Rand AHC	SOM	0.648	0.894	0.521	0.532
	vm-SOM	0.720	0.909	0.549	0.571
Purity	SOM	0.884	0.873	0.595	0.610
	vm-SOM	0.905	0.890	0.602	0.631

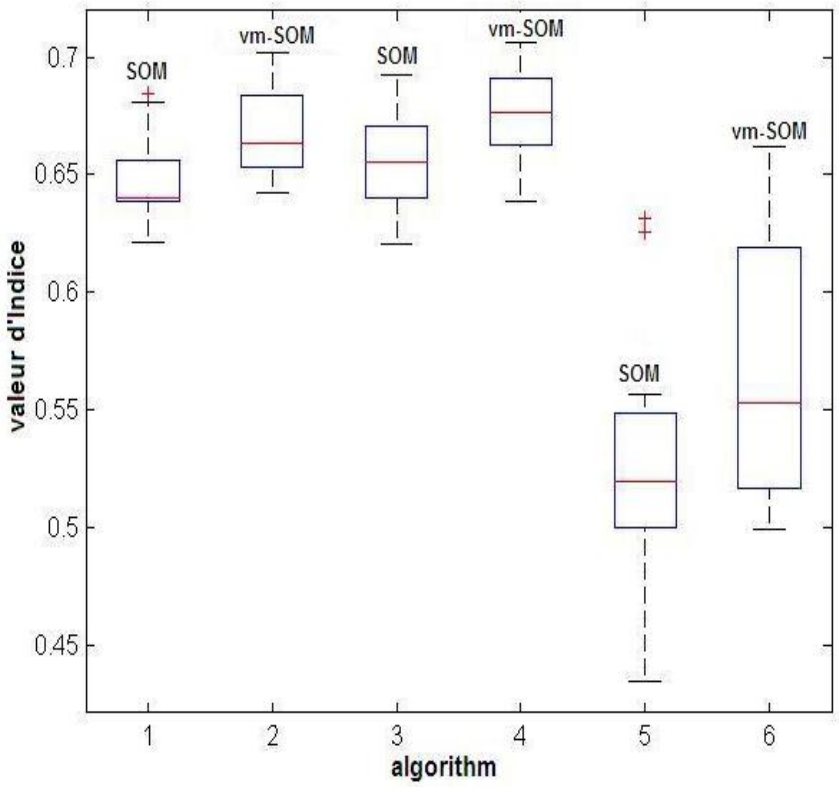


Figure 2.5: Map segmentation: Rand and purity indexes on the waveform dataset

2.3 Unsupervised local weighting for feature selection : Proposed approaches

Feature selection for clustering or unsupervised feature selection aims at identifying the feature subsets so that a model describing accurately the clusters can be obtained from unsupervised learning. This improves the interpretability of the induced model, as only relevant features are involved in it, without degrading its descriptive accuracy. Additionally, the identification of relevant and irrelevant variables with SOM learning provides valuable insight into the nature of the group-structure.
Features selection or variables selection for clustering is difficult because, unlike supervised learning, there are no class labels for the dataset and no obvious criteria to guide the search. The important issue of feature selection in clustering is to provide variables which give the

"best" homogeneous clustering. Therefore, we use the weight and prototype vector $\pi^{[j]}$ and $w^{[j]}$ provided by our proposed weighting approaches to cluster the map and to characterize each cluster with relevance variable.
For map clustering we use traditional hierarchical clustering coupled to Davies-Bouldin index [131] to choose the optimal partition. Thus, to select variables, we use an original method which is based on statistical criteria named *Scree Test* to select most important variables for cell and cluster of cells [21].

In the following, we present two versions of local variables weighting using SOM: weighting observations and weighting the distances.

Firstly, we propose an analytical weighting approach which were inspired from the $w-k$-means and ω-SOM algorithms.

The objective SOM function presented in the chapter 1 can be minimized based on two methods:

- Analytical method which allows us to obtain the SOM batch version;
- Adaptive method using the three steps of the minimization based on gradient descent: SOM stochastic version.

We will present our proposed feature weighted approaches for the both analytical and adaptive SOM versions : observations weighting, distance weighting and double weighting.

2.4 Analytical Weighting Features SOM approach

2.4.1 Local weighting distance : lwd-SOM

This approach introduces a vector of values in the SOM objective function by weighting the distance. The specific algorithm in this case is an extension of w-k-means and SCAD, [42, 65]. Despite the global weighting SOM (w-SOM), a local features weighting consists of weighting the distance between observations and prototypes. The set of parameter has to be updated from dataset X iteratively by minimizing a cost function defined as follows :

$$R_{lwd-SOM}(\chi, \mathcal{W}, \Pi) = \sum_{i=1}^{N} \sum_{j=1}^{|w|} \mathcal{K}_{j,\chi(\mathbf{x}_i)} \left(\pi_j\right)^{\beta} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \tag{2.7}$$

We fix a constraint $\sum_{k=1}^{d} \pi_k = 1$ and we incorporate β as discriminant exponent.

The minimization of $R_{lwd}(\chi, \mathcal{W}, \Pi)$ is run by iteratively performing three steps until stabilization. At each step all observations are assigned the best match unit.

1) Minimize $R_{lwd-SOM}(\chi, \hat{\mathcal{W}}, \hat{\Pi})$ with respect to χ by fixing $\mathcal{W}$ and Π. The expression is defined as follows:

$$\chi(\mathbf{x}_i) = \arg\min_j \left(\left(\pi_j\right)^{\beta} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \right) \tag{2.8}$$

2) Minimize $R_{lwd-SOM}(\hat{\chi}, \mathcal{W}, \hat{\Pi})$ with respect to $\mathcal{W}$ by fixing χ and Π. The expression is the same as in [119]:

$$\mathbf{w}_j = \frac{\sum\limits_{\mathbf{x}_i \in E} \mathcal{K}_{j,\chi(\mathbf{x}_i)} \mathbf{x}_i}{\sum\limits_{\mathbf{x}_i \in E} \mathcal{K}_{j,\chi(\mathbf{x}_i)}}, \tag{2.9}$$

3) Minimize $R_{lwd-SOM}(\hat{\chi}, \hat{\mathcal{W}}, \Pi)$ with respect to Π by fixing χ and $\mathcal{W}$. Here, we use the approach suggested in [42] and adapted to our SOM model. We denote by a vector $\mathbf{D}_j = (D_j^1, ..., D_j^d)$ the distortion associated to cell j and written as follows :

$$\mathbf{D}_j = \sum_{i=1}^{N} K_{j,\chi(\mathbf{x}_i)} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \tag{2.10}$$

$$\tag{2.11}$$

The minimization of the subproblem 3 is doing by respecting the following proposed theorem:

Theorem 2.2. Let $\chi = \hat{\chi}$ and $W = \hat{W}$ be fixed. For $\beta > 1$ or $\beta < 0$, the problem $R_{lwd}(\hat{\chi}, \hat{\mathcal{W}}, \Pi)$ is minimized if:

Each component of the weight vector $\pi_j = (\pi_j^1, \pi_j^2, ..., \pi_j^k, ..., \pi_j^d)$ associated to cell j is computed as follows:

$$\pi_j^k = \begin{cases} 0, \text{ if } D_j^k = 0 \\ \dfrac{1}{\left(\sum\limits_t \left[\dfrac{D_j^k}{D_j^t} \right] \right)^{\frac{1}{\beta-1}}}, \text{for } t \neq l, \quad otherwise \end{cases} \tag{2.12}$$

The demonstration of this theorem is given in the next section of this chapter. The learning algorithm described above allows us to estimate the parameters maximizing the cost function for a fixed neighborhood T. As in the traditional SOM algorithm, we decrease the value of T between two values T_{max} and T_{min} to control the size of the neighborhood influencing a given cell on the map. In the procedure 11 we show the analytical weighting learning process *lwd*-SOM .

Algorithme 11 : Analytical *lwd*-SOM learning algorithm

Input: Data set X; Iter - number of iterations
Initialization Phase:
Randomly initialize the prototype matrix $\mathcal{W}$;
Randomly initialize the weight matrix Π;
for $t = 1$ to $Iter$ **do**
- Assuming that Π and $\mathcal{W}$ are fixed we have to optimize the analytic $R_{lwd-SOM}(\hat{\chi}, \hat{\mathcal{W}}, \hat{\Pi})$, optimize $R_{lwd-SOM}(\hat{\chi}, \hat{\mathcal{W}}, \hat{\Pi})$ with respect to $\hat{\chi}$. This leads us to assign each observation $x \in \mathcal{X}$ to the best match cell using the weighted distance defined in expression 2.34.
- Assuming that $\hat{\chi}$ and Π are fixed, we have to optimize $R_{lwd-SOM}(\hat{\chi}, \mathcal{W},)$. This leads to use the expression (2.9) defined above.
- Assuming that $\hat{\chi}$ and $\mathcal{W}$ are fixed, we have to optimize $R_{lwd-SOM}(\hat{\chi}, \hat{\mathcal{W}}, \Pi)$. This leads to use the expression 2.12.
end for

2.4.1.1 Complexity of the analytical *lwd*-SOM

As we introduced another phase during the learning process, obviously, the complexity time will increase with computing the weights. The complexity time for computing the weights will be $O(|w| \times n \times m)$ where n is the size of the observations, m - the variables size and $|w|$ the number of cells, the same complexity as for the classical affectation phase. Also the computing of the distance will change, it will be $O(n+\pi)$. As π is a value and the complexity for computing the weights is linear, the total complexity of the analytical *lwd*-SOM learning process is $O(|w| \times n \times m + \pi)$. So, without taking into account the constant values (the value of the π for distance computing), the complexity remains linear for the proposed weighting approach and is $O(|w| \times n \times m)$ for one iteration. Therefore, the *lwd*-SOM algorithm deserves good scalability.

2.4.1.2 Minimization of the cost function for lwd-SOM

After introducing the weights ($\pi^{(j)}$) in the classical cost function of SOM, we obtain:

$$R_{lwd-SOM}(\chi, \mathcal{W}, \Pi) = \sum_{i=1}^{N} \sum_{j=1}^{|w|} \mathcal{K}_{j,\chi(\mathbf{x}_i)} \left(\pi_j\right)^{\beta} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \tag{2.13}$$

We note by D_j the distortion of the jth prototype:

$$D_j = \sum_{i=1}^{N} \mathcal{K}_{j,\chi(\mathbf{x}_i)} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \tag{2.14}$$

and for computing the weights we fix $\widehat{\chi}, \widehat{\mathcal{W}}$ and we obtain:

$$R_{lwd-SOM}(\widehat{\chi}, \widehat{\mathcal{W}}, \Pi) = \sum_{j=1}^{|w|} \left(\pi_j\right)^{\beta} \sum_{i=1}^{N} \mathcal{K}_{j,\chi(\mathbf{x}_i)} \|\mathbf{x}_i - \mathbf{w}_j\|^2 = \sum_{j=1}^{|w|} D_j \tag{2.15}$$

We have two situations for the respective distortion:

1) If $D_j = 0 \Rightarrow \widehat{\pi_j} = 0$
2) If $D_j \neq 0$ and forgot that $\sum_{j=1}^{|w|} \pi_j = 1$

In the 2nd case we use the Lagrangian to optimize the cost function.

Lets α - the multiplier and $\psi(\pi, \alpha)$ - the Lagrangian.

We obtain:

$$\psi(\pi, \alpha) = \sum_{j=1}^{|h|} (\pi_j)^{\beta} D_j + \alpha \left(\sum_{j=1}^{|h|} \pi_j - 1) \right) \tag{2.16}$$

where $|h|$ is number of variables ($\leq |w|$) for which $D_j \neq 0$

So, we have:

$$\frac{\partial\psi(\widehat{\pi},\widehat{\alpha})}{\partial\widehat{\pi}^{(j)}} = \beta * (\pi_j)^{\beta-1} D_j + \widehat{\alpha} = 0 \;\; for \;\; all \;\; 1 \leq j \leq h \tag{2.17}$$

$$\frac{\partial\psi(\widehat{\pi},\widehat{\alpha})}{\partial\widehat{\alpha}} = \sum_{j=1}^{|h|} \pi_j - 1 = 0 \tag{2.18}$$

From 2.18 we obtain:

$\beta * (\pi_j)^{\beta-1} D_j + \widehat{\alpha} = 0$

$\beta * (\pi_j)^{\beta-1} D_j = -\widehat{\alpha}$, and $(\pi^{(j)})^{\beta-1} D_j = \dfrac{-\widehat{\alpha}}{\beta * D_j}$

We can express the π_j :

$$\pi_j = (\frac{-\widehat{\alpha}}{\beta * D_j})^{\frac{1}{\beta-1}} \tag{2.19}$$

We substitute 2.19 in 2.18 and we obtain:

$$\sum_{t=1}^{|h|} (\frac{-\widehat{\alpha}}{\beta * D_j})^{\frac{1}{\beta-1}} = 1 \tag{2.20}$$

$$(-\widehat{\alpha})^{\frac{-1}{\beta-1}} = \frac{1}{\left[\sum_{t=1}^{|h|} (\frac{1}{\beta * D_j})^{\frac{1}{\beta-1}}\right]} \tag{2.21}$$

From 2.21 in 2.19 we obtain the expression of weights ($\pi^{(j)}$):

$$\pi_j = \frac{-\widehat{\alpha}^{\frac{1}{\beta-1}}}{(\beta * D_j)^{\frac{1}{\beta-1}}} = \frac{\frac{1}{\left[\sum_{t=1}^{|h|} (\frac{1}{\beta * D_t})^{\frac{1}{\beta-1}}\right]}}{(\beta * D_j)^{\frac{1}{\beta-1}}} = \frac{1}{\left[\sum_{t=1}^{|h|} (\frac{1}{\beta * D_t})^{\frac{1}{\beta-1}}\right] * (\beta * D_j)^{\frac{1}{\beta-1}}} =$$

$$= \frac{1}{\sum_{t=1}^{h} (\frac{\beta * D_j}{\beta * D_t})^{\frac{1}{\beta-1}}} = \frac{1}{\sum_{t=1}^{h} (\frac{D_j}{D_t})^{\frac{1}{\beta-1}}}.$$

This expression allows us to optimize variable weights in order to obtain the best clustering by minimizing the ratio of the average within-cluster distortion over the average between-cluster distortion.

2.4.2 Experimentations and Validation of the analytical *lwd*-SOM

The analytical algorithm *lwd*-SOM described in the section 2.4.1 allows us to obtain on the one hand, a two-dimensional projection data and on the other hand, a weighting of variables specific to each region of the map. Vesanto and Alhoniemi (2000) [131] have proposed to segment a topological map by combining the k-means algorithm and Davies-Bouldin index which allows to automatically determine the size of the partition after segmentation. We have applied this approach on referents and on the weights.
Using the Iris dataset, we evaluated the clustering using the traditional approach and using the k-means on weights vectors. Figure 2.6 shows a segmentation of the 6x4 map in two clusters, using only prototypes of the map.

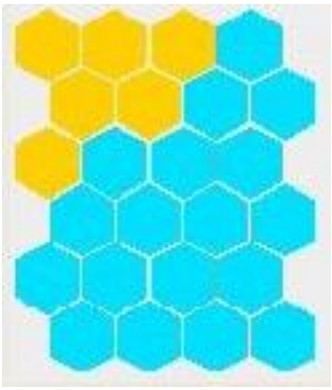


Figure 2.6: Map Segmentation (6x4) with k-means (classical approach - using referents). The DB index = 0.0776

Figure 2.7 shows a segmentation of the same map, but using in this case the weight vectors that take into account the local importance of each variables provided by the vector π of dimension m. There is clearly, in Figure 2.7, that this clustering provides three clusters corresponding to the three basic classes of Iris. This improvement is confirmed by the quality index (Davies Bouldin), which rose from 0.0776 to 0.0556.

We will now compare the different clusterings of the map using the k-means obtained on the waveform dataset. Figures 2.8 and 2.9 show the result of segmentation after learning the topology map with *lwd*-SOM method.

Figure 2.8 shows a segmentation of the map using the traditional approach on prototypes.

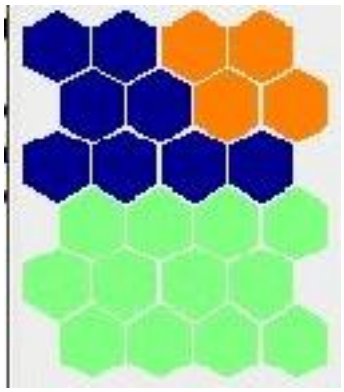


Figure 2.7: Map segmentation (6x4 - Iris dataset) with k-means using weights vector. The DB index = 0.0556

Figure 2.8: Map segmentation (waveform dataset) using k-means on referents vector. The DB index = 0.49

It provides a segmentation of the map in 3 clusters with a quality index equals to 0.49. Figure 2.9 shows a segmentation of the map using the k-means with the local weights. It improves the quality of the partition in finding a partition with 3 clusters with a quality index equals to 0.27.

By observing the results obtained using these two datasets, it is clear that the use of local variables weights makes easier the segmentation of the topological map.

Figures 2.10 and 2.11 represent the different weights of variables associated with prototypes of the topological map obtained with the analytical *lwd*-SOM, in the form of signal. Ana-

Figure 2.9: Map segmentation (waveform dataset) using lwd-SOM with k-means algorithm on the weights of neuron prototypes. The DB index = 0.27

lyzing the maps, it is clear that visually, we can segment the map by combining the referents which have close weights. Unlike the global weighting *w*-SOM technique, the analytical *lwd*-SOM algorithm can characterize each cluster using weighted referents of the map indicating the relevance of each variable. Hence, the variables that have close weights can identifies close referents in order to do a better segmentation.
On the Figure 2.10 which represent the local weights of the Iris dataset, we can detect a subset of prototypes on the top left corner of the map which is characterized by the second and fourth variables associated with very heavy weights. Looking also at the Figure 2.11, the local weights which represent the waveform dataset, it is easily to detect three sets of referents with varying weights. We remind that noise variables are represented in the right part of the signal. For example, in the bottom right corner of the map there are weights which represent more noise than the variables.

The existed weighted approaches pointed out that adaptive weighted allow a better learning than analytical weighted methods. This is why, in the next sections of this chapter we will introduce new adaptive local and global weighted topological approaches in order to attempts a clustering characterization.

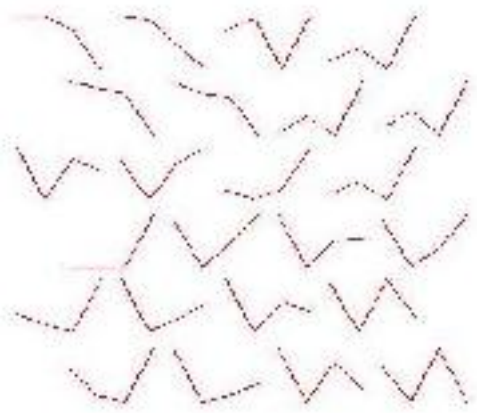


Figure 2.10: The features weights of Iris dataset

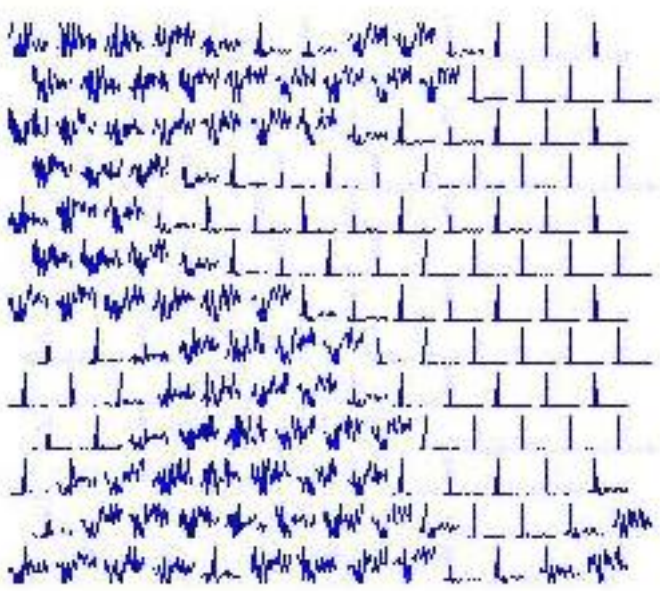


Figure 2.11: The weights of waveform dataset

2.5 Adaptive Weighting SOM

We proposed to use the principle of the *w*LVQ2 weighting technique and to addapt it to the Self-Organizing Maps using the stochastic version of the SOM algorithm. The minimiza-

tion of the objective function is doing using the gradient descent techniques looking for a local minimum. This type of approaches is more efficient than the analytical type weighting, and this is because we uses the adaptive weighting for the clustering characterization. We propose five types of adaptive approaches: local weighting observations; local weighting distance; global weighting observations; global weighting distance and finally the double local weighting. The table 2.2 summarizes the adaptive weighting SOM approaches proposed in this work.

weighting method	adaptive	
	global $\|\pi\| = 1$ vector	local $\|\pi\| = \|W\|$
distance weighting	*gwd*-SOM	*lwd*-SOM
observations weighting	*gwo*-SOM	*lwo*-SOM
double weighting	*dlw*-SOM	

Table 2.2: Notations of the adaptive proposed approaches

2.5.1 Weighting observations

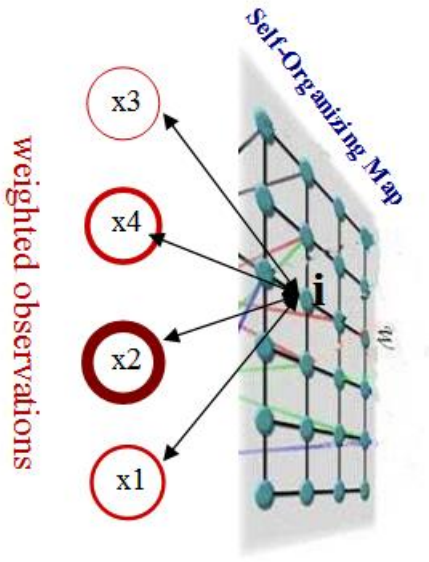


Figure 2.12: Local weighting observations

Weighting the observations during the learning process is a technique which allows to give more importance to the relevant features of the weighted observation. Consider the dataset $X = \{x_1, x_2, x_3, x_4\}$ as it is shown in the figure 2.12 and suppose that the observation x_2 has a bigger relevance in the X. In this case the weighting approach must be able to assign a highest weight value to this one comparing to others three observations. For this type of approach we propose the both local and global weighting described in the next sections.

2.5.2 Local Weighting Observations : lwo-SOM

We based our method on initial work describing the supervised model *w*-LVQ2 [136]. This approach adapts weights to filter the observation during the learning process. Using this model, we weighted observations $\mathbf{x}$ using weight vectors π before computing the distance. So, the weight matrix will be considered like a filtering process for the observations. In the figure 2.13 this filter is the matrix Π which filters the observations before presenting it to the Map.

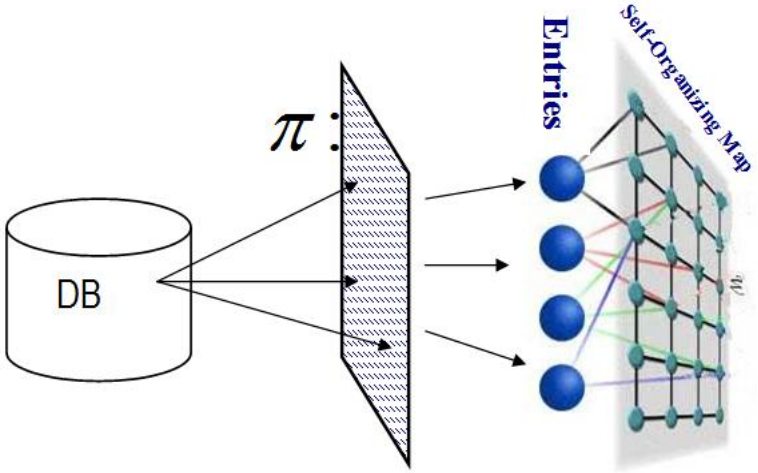


Figure 2.13: Local weighting observations filtering process

The objective function was rewritten as follows:

$$R_{lwo-SOM}(\chi, \mathcal{W}, \Pi) = \sum_{i=1}^{|N|} \sum_{j=1}^{|\mathcal{W}|} \mathcal{K}_{j,\chi(\mathbf{x}_i)} \|\pi_j \mathbf{x}_i - \mathbf{w}_j\|^2 \tag{2.22}$$

Minimization of $R_{lwo}(\chi, \mathcal{W}, \Pi)$ was performed by iterative repetition of the following three steps until stabilization. The initialization step determines the prototype set $\mathcal{W}$ and the set

of associated weights Π, at each training step $(t + 1)$. An observation $\mathbf{x}_i$ is then randomly chosen from the input dataset and the following operations are repeated:

- Minimize $R_{lwo}(\chi, \hat{\mathcal{W}}, \hat{\Pi})$ with respect to χ by fixing $\mathcal{W}$ and Π. Each weighted observation $(\pi_j \mathbf{x}_i)$ is assigned to the closest prototype $\mathbf{w}_j$ using the assignment function, defined as follows:

$$\chi(\mathbf{x}_i) = \arg\min_j \left(\|\pi_j \mathbf{x}_i - \mathbf{w}_j\|^2 \right) \tag{2.23}$$

- Minimize $R_{lwo}(\hat{\chi}, \mathcal{W}, \hat{\Pi})$ with respect to $\mathcal{W}$ by fixing χ and Π. The prototype vectors are updated using the gradient stochastic expression :

$$\mathbf{w}_j(t + 1) = \mathbf{w}_j(t) + \epsilon(t) \mathcal{K}_{j,\chi(\mathbf{x}_i)} \left(\pi_j \mathbf{x}_i - \mathbf{w}_j(t) \right) \tag{2.24}$$

- Minimize $R_{lwo}(\hat{\chi}, \hat{\mathcal{W}}, \Pi)$ with respect to Π by fixing χ and $\mathcal{W}$. The update rule for the feature weight vector $\pi_j(t + 1)$ is :

$$\pi_j(t + 1) = \pi_j(t) + \epsilon(t) \mathcal{K}_{j,\chi(\mathbf{x}_i)} \mathbf{x}_i \left(\pi_j(t) \mathbf{x}_i - \mathbf{w}_j(t) \right) \tag{2.25}$$

As in the traditional stochastic learning algorithm of Kohonen, we denote the learning rate at time t by $\epsilon(t)$. The training is usually performed in two phases. In the first phase, a high initial learning rate $\epsilon(0)$ and a large neighborhood radius T_{max} are used . In the second phase, a low learning rate and small neighborhood radius are used from the beginning. So, for a map there is a associated matrix of weights trained during the learning algorithm.
If we want to obtain a global weighting observations algorithm (*gwo*-SOM) one change this matrix in a vector. In this case we do not take care of the importance of each variable for assigned to each observations. The relevance vector do not depends on the cell and variables, but it depends on the map C: $\pi = (\pi^1, ..., \pi^m)$. We show the adaptive local process in the procedure 12.

2.5.2.1 The complexity for the *lwo*-SOM

Compared to classical SOM, for the proposed weighting algorithm, where is a new phase that appears during the learning process: the weighting phase. Obviously, the complexity of the *lwo*-SOM will increase with the computing of the weight matrix. Due to the local weighting vector, computing the complexity of the weighting phase will be the same like for the affectation phase, but due to the matrix multiplication $\pi_j x_i$, the complexity will increase: $O(|\mathcal{W}| \times (\setminus + \pi))$. Also, the affectation phase will take into account the weights and the

Algorithme 12 : The *lwo*-SOM learning algorithm

Input: Data set X; Iter - number of iterations
Initialization Phase:
Randomly initialize the prototype matrix W;
Randomly initialize the weight matrix Π;
for $t = 1$ to $Iter$ **do**
  Learning Phase:
  Present a learning example x and find the BMU computing the Euclidian distance;
  Updating Phase:
  Compute the new prototypes w using the expression 2.24;
  Compute the weights π using the expression 2.25.
end for

computing the BMU will be more expensive, more exactly : $O(|\mathcal{W}| \times \setminus \times + \pi)$. As π has the size of $|\mathcal{W}|$, the total computation complexity for the local weighting observation is: $O(2|\mathcal{W}| \times \setminus \times)$. Therefore, *lwo*-SOM algorithm deserves good scalability as the complexity remains linear.

2.5.2.2 Minimization of the *lwo*-SOM objective function

The objective function to optimize is the following:

$$R_{lwo-SOM}(\chi, \mathcal{W}, \Pi) = \sum_{i=1}^{|N|} \sum_{j=1}^{|\mathcal{W}|} \mathcal{K}_{j,\chi(\mathbf{x}_i)} \|\pi_j \mathbf{x}_i - \mathbf{w}_j\|^2 \tag{2.26}$$

where $\pi^{(j)}$ is the weighting coefficients matrix.

Learning Rules

We use the gradient descent to optimize the objective function **??**. Firstly, we compute the gradient with respect to $\mathcal{W}$:

$$\nabla_w = \frac{\partial R_{lwo}(\chi, \mathcal{W}, \pi)}{\partial \mathbf{w}_j} = \partial[\mathcal{K}_{j,\chi(\mathbf{x}_i)} \|\pi_j \mathbf{x}_i - \mathbf{w}_j\|^2] / \partial \mathbf{w}_j = \mathcal{K}_{j,\chi(\mathbf{x}_i)} \times 2 \times (\pi_j \mathbf{x}_i - \mathbf{w}_j).$$

So, the adaptation for the prototypes will be:

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) - \epsilon(t) \times \nabla_w = \mathbf{w}_j(t) - \epsilon(t)\mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j \mathbf{x}_i - \mathbf{w}_j) \tag{2.27}$$

We compute now the gradient descent with respect to Π:

$$\nabla_\pi = \frac{\partial R(W,\Pi)}{\partial \pi_j} = \partial[\mathcal{K}_{j,\chi(\mathbf{x}_i)}\|\pi_j x_i - w_j\|^2]/\partial \pi_j = \tag{2.28}$$

$$\mathcal{K}_{j,\chi(\mathbf{x}_i)} \times 2 \times x_i(\pi_j x_i - w_j(t)) \tag{2.29}$$

and, respectively, the assignment step for observation's weights will be defined as follows:

$$\pi_j(t+1) = \pi_j(t) - \epsilon(t)\mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j \mathbf{x}_i - \mathbf{w}_j(t)) \tag{2.30}$$

We use this adaptation rules iteratively for a fixed *Iter* number of iterations.

2.5.3 Global Weighting Observations: gwo-SOM

Weighting the clustering algorithm in a global way will allow us to weight the entire map with the same vector of weight. This is useful when we do not see to find the relevant features for each cluster, but we want to detect them for entire dataset or for the obtained map.

The objective function is the same like for the adaptive learning *lwo*-SOM, changing only the weight matrix π in a vector of weights:

$$R_{gwo}(\chi, \mathcal{W}, \Pi) = \sum_{i=1}^{N}\sum_{j=1}^{|w|} \mathcal{K}_{j,\chi(\mathbf{x}_i)}\|\pi \mathbf{x}_i - \mathbf{w}_j\|^2 \tag{2.31}$$

For each feature we have a corresponding numerical weight, where:

$$\pi(t+1) = \pi(t) - \epsilon(t)\mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi \mathbf{x}_i - \mathbf{w}_j(t)) \tag{2.32}$$

The complexity of the *gwo*-SOM is $O(2|w| \times m \times n)$. Compared to the adaptive $lwo - SOM$ the complexity is the same, the only difference is in the computation of the weighted distance, where the weigh vector will be a value, but as it is a constant value for the time computation, the complexity will not change considerably.

2.5.4 Weighting the distance

Despite the weighting observations approaches, a variable weighting consists of weighting the distance between observations and prototypes. Let X be a dataset with four samples : $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$ and assume that we are looking to compute the distances between observations and a cell i as it is shown in the figure 2.14. Assuming that observations $\mathbf{x}_1$ and $\mathbf{x}_4$ are the most closest to the cell i in the means of the Euclidian distance, thus the weighting procedure will provide more importance to these two samples: the value of the weight which is multiplied to the distance $d_{4,i}$ will be bigger than $d_{3,i}$ or $d_{5,i}$. This is the weighting phase which is doing iteratively during the learning of the map.

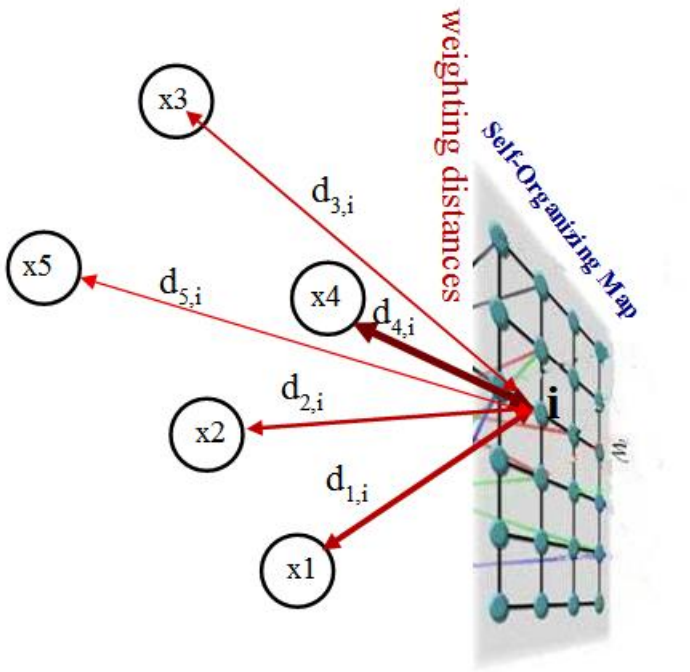


Figure 2.14: Local weighting distance process

2.5.5 Local weighting distance: *lwd*-SOM

Unlike *lwo*-SOM, the local distance weighting involves weighting the distance between observations and prototypes. We propose to minimize the following objective function:

$$R_{lwd}(\chi, \mathcal{W}, \Pi) = \sum_{i=1}^{N} \sum_{j=1}^{|\mathcal{W}|} \mathcal{K}_{j,\chi(\mathbf{x}_i)} (\pi_j)^{\beta} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \tag{2.33}$$

where β is the discrimination coefficient.
The *lwd*-SOM cost function is minimized in three steps:

1) Minimize $R_{lwd}(\chi, \hat{\mathcal{W}}, \hat{\Pi})$ with respect to χ by fixing $\mathcal{W}$ and Π. The equation is defined as follows:

$$\chi(\mathbf{x}_i) = \arg\min_j \left(\left(\pi_j\right)^{\beta} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \right) \tag{2.34}$$

2) Minimize $R_{lwd}(\hat{\chi}, \mathcal{W}, \hat{\Pi})$ with respect to $\mathcal{W}$ by fixing χ and Π. The prototype's vectors are updated using the following equation:

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \epsilon(t) \mathcal{K}_{j,\chi(\mathbf{x}_i)} (\pi_j)^{\beta} \left(\mathbf{x}_i - \mathbf{w}_j(t) \right) \tag{2.35}$$

3) Minimize $R_{lwd}(\hat{\chi}, \hat{\mathcal{W}}, \Pi)$ with respect to Π by fixing χ and $\mathcal{W}$. A weighting vector $\pi_j(t+1)$ is updated according to the following equation:

$$\pi_j(t+1) = \pi_j(t) + \epsilon(t) \mathcal{K}_{j,\chi(\mathbf{x}_i)} \beta (\pi_j(t))^{\beta-1} \|\mathbf{x}_i - \mathbf{w}_j(t)\|^2 \tag{2.36}$$

In addition, the parameter β needs to be provisionally fixed. As with the *lwo*-SOM algorithm, we start with a large initial value for the learning radius which decreases during the learning process allowing quantization of the prototypes. At the end of the learning phase, the *lwd*-SOM model represents a *k*-means model with simultaneously weighted features (SCAD [42] or *w-k*-means [65]). The general procedure of the *lwd*-SOM learning algorithm is given in the Algorithm 13.

2.5.6 Global Weighting Distance : *gwd*-SOM

As for the *gwo*-SOM technique, in this case we can extend the algorithm to a global weighting distance using adaptive (stochastic) learning SOM and we obtain the *gwd*-SOM which uses a weighting vector of distances (not a matrix) which depends only on the map. The objective function for the global weighting distance will be rewritten as following:

Algorithme 13 : The adaptive *lwd*-SOM learning algorithm

Input: Data set X; Iter - number of iterations
Initialization Phase:
Randomly initialize the prototype matrix w;
Randomly initialize the weight matrix π;
for $t = 1$ to $Iter$ **do**
  Learning Phase:
  Present a learning example x and find the BMU computing the weighted (*lwd*) squared Euclidian distance;
  Updating Phase:
  Compute the new prototypes w using the expression 2.37;
  Compute the distance weights π using the expression 2.38.
end for

$$R_{gwd-SOM}(\chi, \mathcal{W}, \Pi) = \sum_{i=1}^{N} \sum_{j=1}^{|\mathcal{W}|} \mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi)^{\beta} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \tag{2.37}$$

2.5.6.1 Complexity of the adaptive l/gwd-SOM technique

Even the l/gwd-SOM compute the distance weighting matrix contrarily to l/gwo-SOM which compute the weighting observation matrix, the computational time for the l/gwd-SOM is the same as for the l/gwo-SOM due to the same size of the computed matrix (a weight vector for a map's cell). So, the computational time for the l/gwd-SOM is $O(2|\mathcal{W}| \times n \times m)$. The both local and global distance weighting approaches has a good scalability depending on the number of iterations and the map size.

2.5.6.2 Minimization of the weighted objective function $R_{lwd-SOM}$

We introduce the weights (π_j) in the classical cost function of SOM, and we obtain:

$$R_{lwd}(\chi, \mathcal{W}, \Pi) = \sum_{i=1}^{N} \sum_{j=1}^{|w|} \mathcal{K}_{j,\chi(\mathbf{x}_i)} \left(\pi_j\right)^{\beta} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \tag{2.38}$$

We can note:

$$\nabla_w = \frac{\partial R_{lwd}(\chi, \mathcal{W}, \pi)}{\partial \mathbf{w}_j} = \partial \mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j)^\beta [\|\mathbf{x}_i - \mathbf{w}_j\|^2] / \partial \mathbf{w}_j = \mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j)^\beta (\mathbf{x}_i - \mathbf{w}_j)$$

and, respectively, the adaptation phase for the observations weights will be:

$$\mathbf{w}_j(t+1) = w_j(t) - \epsilon(t)\mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j)^\beta (x_i - \mathbf{w}_j(t)) \tag{2.39}$$

To find the expression for the weight matrix adaptation we use the gradient descend by respect to Π:

$$\nabla_\pi = \frac{\partial R_{lwd}(\chi, \mathcal{W}, \pi)}{\partial \pi_j} = \partial [\mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j)^\beta \|\mathbf{x}_i - \mathbf{w}_j\|^2] / \partial \pi_j = \mathcal{K}_{j,\chi(\mathbf{x}_i)} \beta (\pi_j)^{\beta-1} (x_i - w_j)^2$$

and, finally, the adaptation for the distance weights is:

$$\pi_j(t+1) = \pi_j(t) - \varepsilon(t)\mathcal{K}_{j,\chi(\mathbf{x}_i)} \beta (\pi_j)^{\beta-1} (x_i - w_j)^2 \tag{2.40}$$

Usually the β parameter is fixed on the beginning and has value from 1 to 10. This parameter doesn't have a big influence on the learning process, but it is used as a smoothing parameter.

2.5.7 Double local weighting SOM

The both adaptive local weighting approaches *lwo*-SOM and *lwd*-SOM depends on the initial data, if one make confidence to the observations we will weight the observations using the *lwo*-SOM, contrarily we will look on the data distribution when weighting the distance using the *lwd*-SOM. It is difficult to extract this information from a dataset, and sometimes weighting the observations can give better results then weighting the distances and vice versa, relatives to the dataset. Therefore, we introduce another weighting approach, which integrates both adaptive local weighting methods called *dlw*-SOM (double local weighting Self-Organizing Map).

Let $X = \{x_1, x_2, x_3, x_4, x_5\}$ a set of samples which should be placed on the map which is shown in the figure 2.15. The double weighting approach detects that x_2 is more important, but x_4 is the nearest observations to the cell i. So, the algorithm *dlw*-SOM will affect a

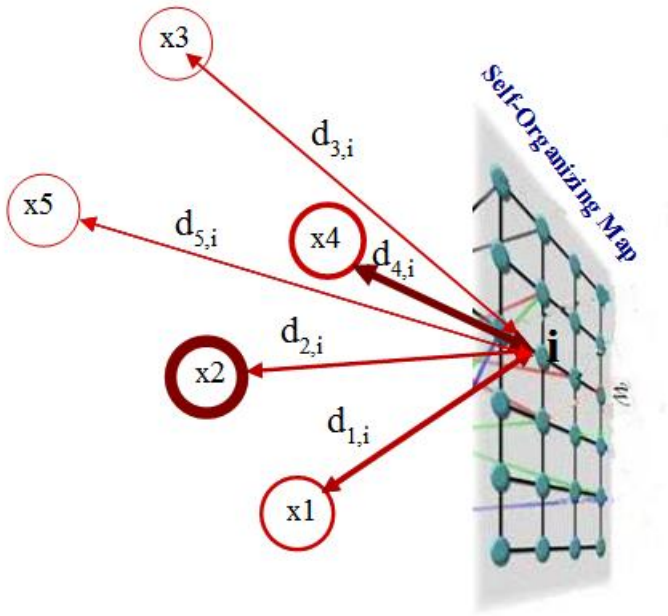


Figure 2.15: Double local weighting process

highest weight to the observations x_2 and x_4. Moreover, computing the weights of the observation take into account the distance weight matrix and vice versa. It means that the double weighting procedure allows us to weight simultaneously the observation and the distance.

For the double local weighting process, we introduce the both weights in the SOM objective function, and we obtain:

$$R_{dlw-SOM}(\chi, \mathcal{W}, \Pi^{(d)}, \Pi^{(o)}) = \sum_{i=1}^{|N|} \sum_{j=1}^{|\mathcal{W}|} \mathcal{K}_{j,\chi(\mathbf{x}_i)} (\pi_j^{(d)})^{\beta} \|\pi_j^{(o)} \mathbf{x}_i - \mathbf{w}_j\|^2 \tag{2.41}$$

where $\Pi^{(d)}$ are the distance weights and $\Pi^{(o)}$ - the observations weights.

As we combined two types of the weighting techniques, contrarily to precedent weighting approaches, the minimization of the $R_{dlw-SOM}$ objective function is doing in four steps:

1) Minimize $R_{dlw}(\chi, \hat{\mathcal{W}}, \hat{\Pi^{(d)}}, \hat{\Pi^{(o)}})$ with respect to χ by fixing $\hat{\mathcal{W}}$, $\hat{\Pi^{(d)}}$ and $\hat{\Pi^{(o)}}$. The expression is defined as follows:

$$\chi(\mathbf{x}_i) = \arg\min_j \left((\pi_j^{(d)})^{\beta} \|\pi_j^{(o)} \mathbf{x}_i - \mathbf{w}_j\|^2 \right) \tag{2.42}$$

2) Minimize $R_{dlw-SOM}(\hat{\chi}, W, \hat{\Pi^d}, \hat{\Pi^o})$ with respect to W by fixing $\hat{\chi}$, $\hat{\Pi^{(d)}}$ and $\hat{\Pi^{(o)}}$. The prototype's vectors are updated using the following expression:

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \epsilon(t)\mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j^{(d)})^{\beta}\left(\pi_j^{(o)}\mathbf{x}_i - \mathbf{w}_j(t)\right) \tag{2.43}$$

3) Minimize $R_{dlw-SOM}(\hat{\chi}, \hat{\mathcal{W}}, \hat{\Pi^{(d)}}, \Pi^{(o)})$ with respect to $\Pi^{(o)}$ by fixing $\hat{\chi}$, $\hat{\mathcal{W}}$ and $\hat{\Pi^{(d)}}$. The update of the observation weights vectors ${\pi^{(o)}}_j(t+1)$ are made using the following expression:

$$\pi_j^{(o)}(t+1) = \pi_j^{(o)}(t) + \epsilon(t)\mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j^{(d)}(t))^{\beta}\mathbf{x}_i\left(\pi_j^{(o)}(t)\mathbf{x}_i - \mathbf{w}_j(t)\right) \tag{2.44}$$

4) Minimize $R_{dlw-SOM}(\hat{\chi}, \hat{\mathcal{W}}, \Pi^{(d)}, \hat{\Pi^{(o)}})$ with respect to $\Pi^{(d)}$ by fixing $\hat{\chi}$, $\hat{\mathcal{W}}$ and $\hat{\Pi^{(o)}}$. The update of the distance weights vectors ${\pi^{(d)}}_j(t+1)$ are made using the following expression:

$$\pi_{(d)\,j}(t+1) = \pi_j(t) + \epsilon(t)\mathcal{K}_{j,\chi(\mathbf{x}_i)}\beta(\pi_{(d)j}(t))^{\beta-1}\left(\pi_j^{(o)}(t)\mathbf{x}_i - \mathbf{w}_j(t)\right) \tag{2.45}$$

The algorithm is presented in the following procedural schema:

Algorithme 14 : The *dlw*-SOM learning algorithm

Input: Data set X; Iter - number of iterations
Initialization Phase:
Randomly initialize the prototype matrix W;
Randomly initialize the observation weight matrix $\Pi^{(o)}$;
Randomly initialize the distance weight matrix $\Pi^{(d)}$;
for $t = 1$ to $Iter$ **do**
  Learning Phase:
  Present a learning example x and find the BMU computing the double weighted Euclidian distance (expression 2.42);
  Updating Phase:
  Compute the new prototypes w using the expression 2.43;
  Compute the observations weights $\pi^{(o)}$ using the expression 2.44
  Compute the distance weights $\pi^{(d)}$ using the expression 2.45
end for

2.5.7.1 Complexity of the *dlw*-SOM

As the *dlw*-SOM has four optimization steps it is clearly that we increase the computational complexity compared to the both *lwo*-SOM and *lwd*-SOM weighting approaches. Analyzing the computational costs for all phases we can extract:

- Finding the best matching unit will take into account the weights (double weighting distance) and it will be : $O(\pi^{(d)} + (\pi^{(o)} + n \times m \times |\mathcal{W}|))$
- The affectation phase will use the both matrix weights, and the complexity cost is: $O(\pi^{(d)} + (\pi^{(o)} + n \times m \times |\mathcal{W}|))$;
- For the observations weights computing, the cost is: $O(\pi^{(d)} + (\pi^{(o)} + m \times n \times |\mathcal{W}|))$;
- Finally, the computational cost for the distance weights is: $O(\pi^{(d)} + \pi^{(o)} + n \times m \times |\mathcal{W}|))$

As the size of the weights matrix are the same for the map size, the total computational cost for the *dlw*-SOM is $O(2|W| + |W| \times n \times m)$. Even *dlw*-SOM increase with two phases the learning of the SOM map, the algorithm is still efficient due to its linear complexity and improves a good scalability. As we can see, the complexity depends in a bigger part on the map size (number of $|\mathcal{W}|$) because for each cell, the algorithm will compute the prototype and the both weights, but, thanks to the usually small number of cells, the complexity time does not grow significantly.

2.5.7.2 Minimization of the *dlw*-SOM objective function

In order to minimize the $R_{dlw-SOM}$ objective function (eq. 2.41) we use the technique of descent gradient for the four following phases:

1) To minimize the $R_{lwd-SOM}$ with respect to W we obtain:

$$\frac{\nabla R_{dlw-SOM}}{\nabla w} = (\mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j^{(d)})^{\beta}\|\pi_j^{(o)}\mathbf{x}_i - \mathbf{w}_j\|^2)' = \mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j^{(d)})^{\beta} * 2(\pi_j^{(o)}\mathbf{x}_i - \mathbf{w}_j);$$

The prototypes assignment is doing using the next expression:

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \epsilon(t)\mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j^{(d)})^{\beta}\left(\pi_j^{(o)}\mathbf{x}_i - \mathbf{w}_j(t)\right)$$

2) The minimization of the objective function by respect to the distance weights $\Pi^{(d)}$) is doing as following:

$$\frac{\nabla R_{dlw-SOM}}{\nabla \pi_j^{(d)}} = \|\pi_j^{(o)}\mathbf{x}_i - \mathbf{w}_j\|^2 \mathcal{K}_{j,\chi(\mathbf{x}_i)}[\pi_j^{(d)})^{\beta}]' = \|\pi_j^{(o)}\mathbf{x}_i - \mathbf{w}_j\|^2 \mathcal{K}_{j,\chi(\mathbf{x}_i)}\beta[\pi_j^{(d)})^{(\beta-1)}];$$

Consequently, the assignment of the distance weights is:

$$\pi_j^{(d)}(t+1) = \pi_j^{(d)}(t) + \epsilon(t)\mathcal{K}_{j,\chi(\mathbf{x}_i)}\beta(\pi_j^{(d)}(t))^{\beta-1}\left(\pi_j^{(o)}(t)\mathbf{x}_i - \mathbf{w}_j(t)\right)$$

3) The minimization of the $R_{dlw-SOM}$ by respect to ($\Pi^{(o)}$) is the following:

$$\frac{\nabla R_{dlw-SOM}}{\nabla \pi_j^{(o)}} = \mathcal{K}_{j,\chi(\mathbf{x}_i)}\pi_j^{(d)})^{\beta}[\|\pi_j^{(o)}\mathbf{x}_i - \mathbf{w}_j\|^2]' = \mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j^{(d)})^{\beta}2\mathbf{x}_i(\pi_j^{(o)}\mathbf{x}_i - \mathbf{w}_j);$$

So, the expression for computing the observations weight matrix is:

$$\pi_j^{(o)}(t+1) = \pi_j^{(o)}(t) + \epsilon(t)\mathcal{K}_{j,\chi(\mathbf{x}_i)}(\pi_j^{(d)}(t))^{\beta}\mathbf{x}_i\left(\pi_j^{(o)}(t)\mathbf{x}_i - \mathbf{w}_j(t)\right)$$

4) Finally, the minimization with the respect to χ is done by using the expression:

$$\chi(\mathbf{x}_i) = \arg\min_j \left((\pi_j^{(d)})^{\beta}\|\pi_j^{(o)}\mathbf{x}_i - \mathbf{w}_j\|^2\right)$$

2.6 Feature selection and Cluster characterization

In this section we propose to use an automatic procedure to select the relevant features using the weights obtained during the learning process. After obtaining the map we cluster it using a k-means or HCA algorithm on the map prototypes. Finally, on the map clusters we apply the variables selection procedure to detect the relevant features for each cluster separately.

2.6.1 Automatic Variables Selection : Catell Scree Test

We propose to use an established statistical method, *scree test*, to select the most important features [21].

This statistical test was initially developed to provide a visual technique to select eigenvalues for principal components analysis [21]. The basic idea is to generate a curve associated with eigenvalues, allowing random behavior to be identified. The number of components retained is equal to the number of values preceding this 'scree'. Often the 'scree' appears where the slope of the graph changes radically. We therefore needed to identify the point of maximum deceleration in the curve.

Figure 2.16 shows an example of a curve generated using a weight vector. We observed the scree on the 19th feature which means that the irrelevant features have index values lying in the range [20 – 40]. We used an automated process to apply this technique to each weight vector $\pi_j = (\pi_j^1, \pi_j^2, ..., \pi_j^d)$.

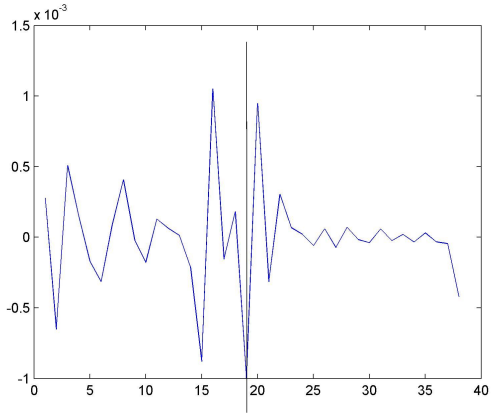


Figure 2.16: An example of the automatic scree test using a particular weight vector. The axes X and Y correspond to features and weights, respectively. The scree is indicated by the vertical bar.

Thus we have to process the following steps presented in the procedure 15.

2.6.1.1 Complexity of the Scree Test procedure

The Scree Test acceleration procedure has four steps until finding the scree in the vector. We will analyze all these steps:

Algorithme 15 : The Scree Test Acceleration Factor

Input: weights vector π_j size m

for $i = 1$ to m **do**

Sort the weights in descending order $\pi^{[j]}$.

Thus we obtain a new order $\pi^{[j]} = (\pi_{\cdot}^{[j],1}, \pi_{\cdot}^{[j],2}, ..., \pi_{\cdot}^{[j],i}, ..., \pi_{\cdot}^{[j],d})$; where i indicates the index order.

end for

for $j = 1$ to m (on the sorted vector) **do**

Compute the first difference $df_i = \pi_{\cdot}^{[j],i} - \pi_{\cdot}^{[j],i+1}$ and we obtain the vector $\pi_{df1}^{[j]}$

end for

for $k = 1$ to m (on the $\pi_{df1}^{[j]}$ vector) **do**

Compute the second difference (acceleration) $acc_i = df_i - df_{i+1}$ obtaining the vector $\pi_{df2}^{[j]}$

end for

for $l = 1$ to m (on the $\pi_{df2}^{[j]}$ vector) **do**

Find the scree: $\max_i (abs(acc_i) + abs(acc_{i+1}))$

end for

OUTPUT:

Retain all the features displayed before the scree (we used the initial index values of features before sorting).

- To made the sort of the weight vector we are using the Merge sort procedure which has an algorithmic complexity: $O(m \log m)$;
- The complexity for the first difference df_i is the $O(m)$;
- For the second difference the complexity is the same as previously: $O(m)$;
- Even if the scree is in the beginning of the vector, the algorithm must look over entire weight vector to see if there is no another bigger scree, and the complexity is $O(m)$.

As there is no nested loops, the total computational time for the Scree Test acceleration algorithm is the sum of the complexity of the four steps, and for a weight vector it is $O(m \log m + 3m)$.

2.6.2 Automatic cluster characterization trough features selection

Feature selection for clustering or unsupervised feature selection is used to identify the feature subsets that accurately describe the clusters. This improves the interpretability of the induced model, as only relevant features are involved in it, without degrading its descriptive accuracy. Additionally, the identification of relevant and irrelevant features with SOM learning provides valuable insight into the nature of the cluster-structure.

Feature selection for clustering analysis is difficult because, unlike supervised learning, there are no class labels for the dataset and no obvious criteria to guide the search [133]. Feature selection in clustering must provide features that describe the "best" homogeneous cluster. Here, we used the weight set Π and prototype set $\mathcal{W}$ provided by the adaptive *lwo*-SOM, *lwd*-SOM and *dlw*-SOM. We then clustered the map and used features selection to characterize the resulting clusters associated with cells and group of cells. For map clustering we used traditional hierarchical clustering combined with the Davies-Bouldin index to choose the optimal partition [131]. Thus, to select the relevant features, we use the Scree Test Acceleration Factor (algorithm 15).

Algorithme 16 : The Clustering Characterization Procedure

Input: Dataset X size $n \times m$
for $i = 1$ to n **do**
  Build a weighting map size $|\mathcal{W}|$ using one of the weighting approaches (*lwd*-SOM or *lwo*-SOM)
end for
for $j = 1$ to C **do**
  Cluster the map using a clustering method (k-means or HCA) obtaining k numbers of clusters
end for
for $l = 1$ to k (for each cluster) **do**
  for $j = 1$ to $|\mathcal{W}|$ (for each prototype) **do**
    Find the relevant subset of features using the ScreeTest procedure (for each cell of the cluster)
  end for
end for
OUTPUT: The relevant subset of variables characterizing the k clusters of the map $|\mathcal{W}|$

To attempts the clustering characterization, we integrate the weighting SOM proposed approaches, the clustering of the map and variables selection schema (Scree Test) in one procedure which is presented in the algorithm 16.

2.6.2.1 The complexity of the Clustering Characterization Procedure

Let n be the number of observations; m -the size of variables and C - the size of the map, the clustering characterization procedure is composed from three phases:

1) Weighting. If we use *lwo*-SOM weighting approach, the complexity for this phase is $O(2|\mathcal{W}| \times \setminus \times)$;

2) Clustering. The complexity of the k-means algorithm is $O(|\mathcal{W}| \times \times)$ which cluster the map size C in k clusters;

3) Selection. The computational time of the Scree acceleration Test procedure for the k clusters is : $O(m \log m \times k)$.

So, the total complexity time for the proposed clustering characterization technique is $O(2|\mathcal{W}| \times \setminus \times + |\mathcal{W}| \times \times + \log \times)$. This complexity depends on the size of variables which is the case for all the variables selection algorithms, and on the size of the map, because the proposed methods are using the map prototypes to weight and to cluster the dataset.

2.6.3 Experimental Results for the Cluster Characterization (using the adaptives approaches: l/gwo-SOM and l/gwd-SOM)

We have performed several experiments on five known problems from the UCI Repository of machine learning databases : waveform, spambase, madelon, isolet and Wisconson cancer database [9]. For more details on the datasets, the lecture can find out in the Appendix A.1.

To evaluate the quality of clustering, we compared results to a "ground truth". We used the clustering accuracy for measuring the clustering results. In general, the results of clustering are usually assessed on the basis of some external knowledge about how clusters should be structured. The only way to assess the usefulness of a clustering result is indirect validation, whereby clusters are applied to the solution of a problem and the correctness is evaluated against objective external knowledge. This procedure is defined by [69] as "validating clustering by extrinsic classification", and has been used in many other studies. To use this approach we therefore need labeled datasets, where the external (extrinsic) knowledge is the class information provided by labels. Thus, the identification of significant clusters in the data, by *lwo*-SOM, *lwd*-SOM or *dlw*-SOM will be reflected by the distribution of classes. A purity score can thus be expressed as the percentage of elements in a cluster that have been assigned a particular class.

We also validated our approaches in supervised case learning paradigms. We used the K-fold cross validation technique, repeated s times for $s = 5$ and $K = 3$, to estimate the performance of g/lwo-SOM, g/lwd-SOM and dlw-SOM. For each run, the dataset was split into three disjoint subsets of equal size (15 runs for each dataset). We used two subsets for training and then tested the model on the remaining subset using all features and selected features (selected on the cells or on clusters). The labels generated were compared to the real labels of the test set for each run.

We used the purity index to evaluate the quality of map segmentation. This index shows the correspondence between the class of data and cluster label, which is computed using the majority vote rule. A high value for this measure indicates a high level of homogeneous clustering. A purity index value close to 0 is indicative of poor clustering, whereas an index value close to 1 is indicative of a good clustering result.

2.6.4 Results on the waveform dataset for g/lwo-SOM and g/lwd-SOM

We used this dataset to show a good level of performance for both algorithms (lwd-SOM and lwo-SOM) for simultaneous clustering and feature weighting. All observations were used to generate a map with 26×14 cells dimension. Both learning algorithms provided two vectors for each cell: the referent vector $\mathbf{w}_j = (w_j^1, w_j^2, ..., w_j^d)$ and weight vector $\pi_j = (\pi_j^1, \pi_j^2, ..., \pi_j^d)$, where $d = 40$. Preparing data for clustering requires some preprocessing, such as normalization or standardization. In the first experimentation step, we normalized the initial dataset (Figure 2.18(a)) to obtain more homogeneous data (Figure 2.18(b)). We used variance normalization, representing a linear transformation that scales the values such that their variance is equal to 1.

We created 3D representations of the referent vector and weight vector provided by classical SOM and by our methods (g/lwd-SOM and g/lwo-SOM). The axes X and Y indicate the features and the referent indexes, respectively. The amplitude indicates the mean value of each component. Examination of the three graphs ($2.18(c), 2.19(b), 2.20(b)$) shows that the noise represented by features 19 to 40 may be clearly detected with low amplitudes. This visual analysis of the results clearly shows that lwo-SOM algorithm provides the best results. Both graphs of weights Π and prototypes $\mathcal{W}$ show that features associated to noise is irrelevant with low amplitude. Visual analysis of both weight vectors (figure 2.19(d) and figure 2.20(d)) showed the weight vectors obtained with lwo-SOM to give a more accurate representation of the data structure (features relevance) than the weight vectors obtained with lwd-SOM.

The lwo-SOM algorithm provides good results because the weight vectors work as a filter for observations and estimates the referents that result from this filtering. We applied the

selection task to all parameters of the map before and after map clustering to check that it was possible to automatically select the features using our algorithms. This task involves detecting major changes for each input vector represented as a signal graph. We used hierarchical classification [131] for clustering the map.

After *lwo*-SOM map clustering, we obtained three clusters with a purity index equal to 0.7076. Using *lwd*-SOM resulted in six clusters. However, in *lwd*-SOM map, clustering with the product $\Pi\mathcal{W}$ led to the generation of three clusters (purity index equal to 0.6803), which were significant in our example. This demonstrates that when there is no cluster (labels) information, feature weighting can be used to find and characterize homogeneous clusters.
The importance of this index is that it can give us information about each clusters in a visual mode. Using only referents of lwd-SOM we find 5 clusters and analyzing the figure 2.17 we detect some errors on 4th and 5th clusters because the waveform dataset has only 3 clusters. The plot founded on the left part of the figure shows the wrong labeled observations.

In the case of both global weighting algorithms, we can see that some noise features has a high value, and even for the *gwo*-SOM the first features (1-20) do not describe well the waves. This disadvantage compared to local weighted approaches is because the global weighting technique uses only a vector of weights for all the data, and respectively each sample vector will be weighted with the same vector of weights. The weights obtained with *gwo*-SOM (Figure 2.19(a)) gives better results than using *gwd*-SOM (Figure 2.20(a)) because that the observations in this dataset is more significant that the distances computed during the learning process.

After *lwo*-SOM map clustering with the referents $\mathcal{W}$, which are already weighted, we obtain 3 clusters. Using *lwd*-SOM, the clustering of the map with the referents $\mathcal{W}$ provide 6 clusters, but the clustering of the map using the product $\Pi\mathcal{W}$ leads to 3 clusters which is significant for our example. This means that when where is no cluster (labels) knowledge, the variable weighting helps to find the pureness clusters, and to characterize them.

The characterization of clusters with the "Scree Test" algorithm is provided in Table 2.3. For each algorithm, we present the features selected for each cluster. Both techniques (*lwo*-SOM, *lwd*-SOM) provided three clusters characterized by different features. By contrast, segmentation of the map using classical SOM provided six clusters with a purity index value of 0.662. Map segmentation was performed using hierarchical clustering with all the features. For clusters cl_1, cl_2 and cl_3, the features selected using *lwd*-SOM were also selected using *lwo*-SOM. We found that both algorithms *lwo*-SOM and *lwd*-SOM identified relevant and informative features, giving more accurate results than classical SOM. The new and classical methods were compared after segmentation of the map. We investigated the effect of

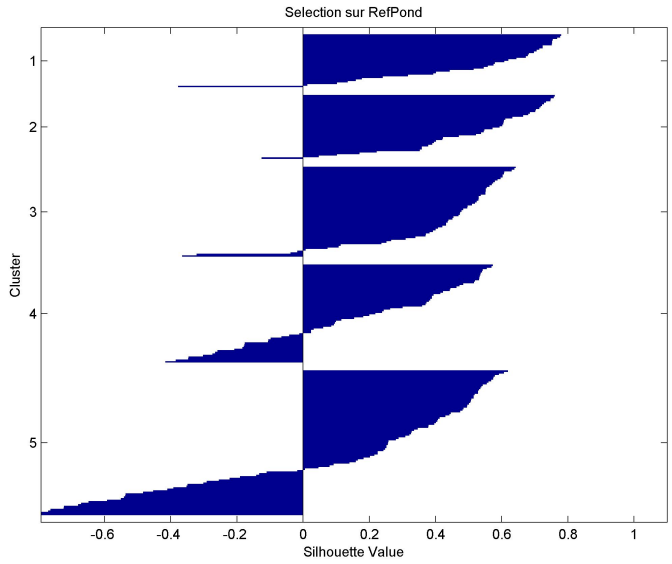
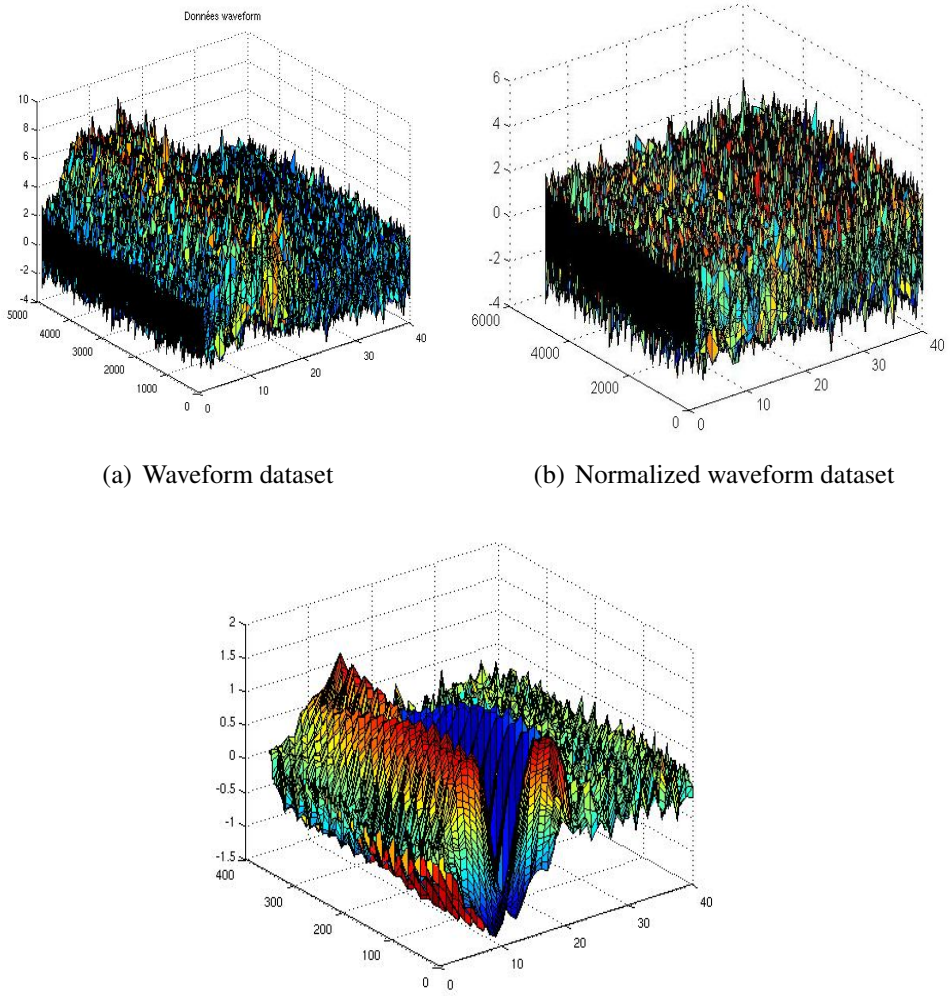


Figure 2.17: Visualization of silhouette index using lwd-SOM

selected features before and after, or without segmentation by testing this selection process in the supervised paradigm and computing the accuracy index for each method.

In the case of global weighting approaches (*gwd*/*gwo*-SOM) we are not able to characterize each cluster because the weight vector are the same for all the prototypes, but we can detect the relevant features for the whole map (dataset). We can see that the set of selected features using these global weighting algorithms (Table 2.3) represent the union of relevant features obtained with the local weighting approach for all the clusters.

In order to evaluate the relevance of variable selected, we compute purity score by running a 3-fold cross-validation five times. Figure 2.23 shows the box plot indicating the purity scores calculated for each run with learning dataset, using SOM, *gwo*-SOM, *lwo*-SOM, *gwd*-SOM and *lwd*-SOM. We show also the result after clustering the corresponding map using hierarchical clustering. We observe that the *lwo*-SOM method has significantly better score compared to traditional SOM and the second proposed method *lwd*-SOM. The score is degraded after clustering the map, but *lwo*-SOM and *lwd*-SOM are still significantly better than traditional SOM. In the second time we evaluate the variable selected by assigning the dataset (test part) during the cross validation task. Figure 2.24 shows the purity score using traditional SOM, *lwd*-SOM and *lwo*-SOM. The classification task are tested using all variables, the variables selected by cell and the variables selected after clustering map. We observe that using *lwo*-SOM and *lwd*-SOM improve the performance and reduce the vari-



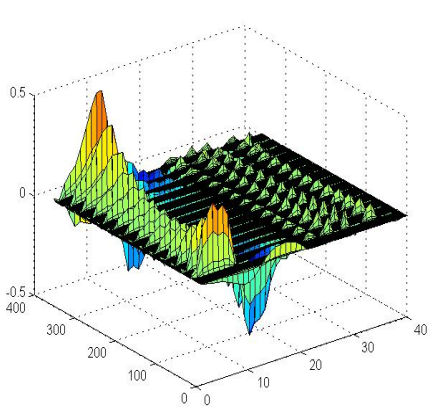
(a) Waveform dataset

(b) Normalized waveform dataset

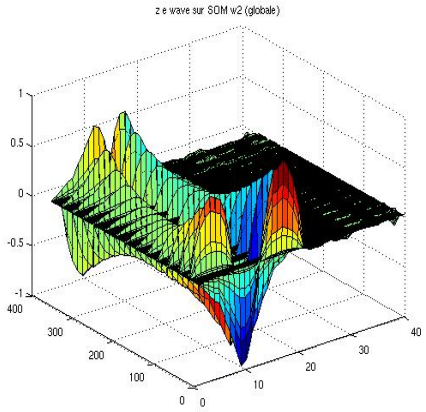
(c) (*W* provided by SOM)

Figure 2.18: Waveform dataset

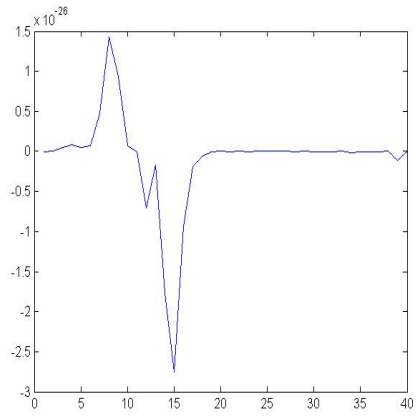
ance of the results. Among the performed tests, *lwo*-SOM had the best performance (high level of accuracy and low variance) in the three cases.



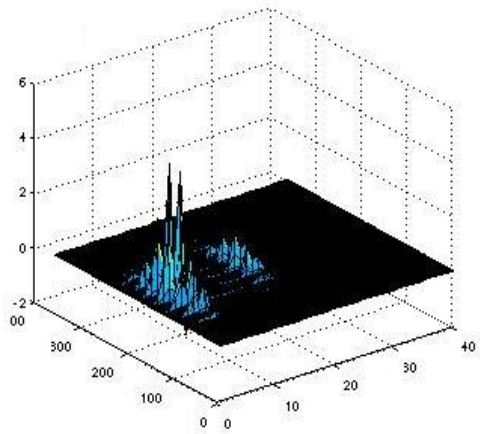
(a) (W provided by *gwo*-SOM)



(b) (W provided by *lwo*-SOM)



(c) Π provided by *gwo*-SOM

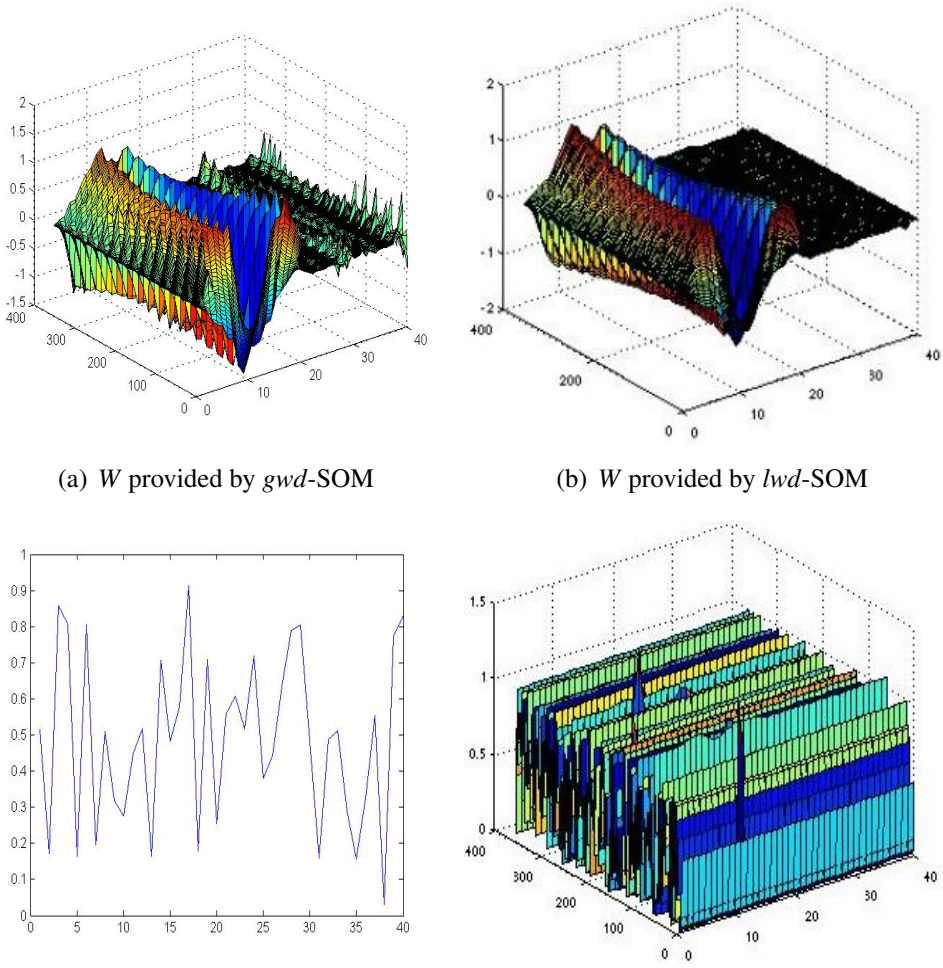


(d) Π provided by *lwo*-SOM

Figure 2.19: 3D visualization of the referent vector and weight vector. The axes X and Y indicate features and the referent index values, respectively. The amplitude indicates the mean value of each component of map 26×14 (364 cells).

Table 2.3: Comparison of the selected variables using traditional and our approaches (g/lwo-SOM, g/lwd-SOM). $[i-j]$ indicates the set of selected variables

Db	# real cluster	*gwd*-SOM	*lwd*-SOM $\Pi\mathcal{W}$	$gwo-SOM$	*lwo*-SOM $\mathcal{W}$
wave-form	3	[4-19]	cl_1: [6-15] cl_2: [4-10] cl_3: [7-19]	[3-20]	cl_1: [3-8; 11-16] cl_2: [8-11; 14-19] cl_3: [3-20]



(a) W provided by *gwd*-SOM

(b) W provided by *lwd*-SOM

(c) Π provided by *gwd*-SOM

(d) Π provided by *lwd*-SOM

Figure 2.20: 3D visualization of the referent vector and weight vector of both local and global distance weighting SOM. The axes X and Y indicate respectively the variables and the referent indexes. The amplitude indicates the mean value of each component of map 26×14 (364 cells).

Feature selection for cluster characterization and visualization

In this section we use our model for feature selection. We propose here to use the statistical scree test presented above to select automatically the informative variables. In order to simplify this step we cluster the map provided by the both models. For this task we use the traditional hierarchical clustering [131]. With waveform dataset we obtain three clusters. Figure 2.21 display the 2D visualization of three clusters. Color graduation indicates the relevance of the features (red - important; blue - non important). We compare our results with

a visual cross-clustering technique presented in the figure 2.22. As we can see the 'cross-clustering' method obtain a 4 class clustering and the 4th cluster is represented by all the relevant features from the waveform dataset.

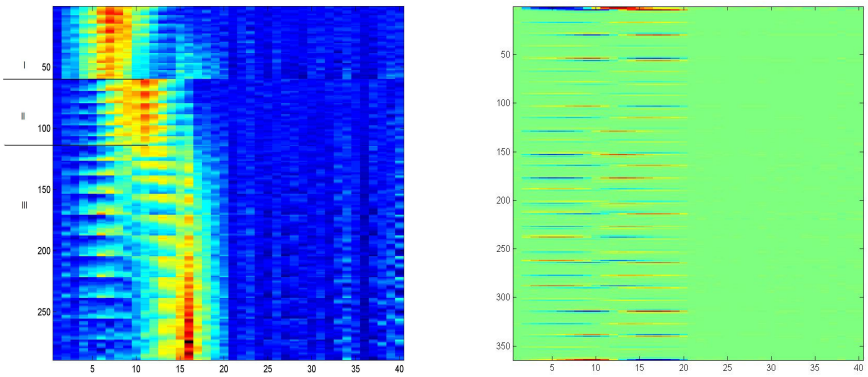


Figure 2.21: Cluster characterization of the map obtained using *lwo*-SOM and *lwd*-SOM after hierarchical clustering of waveform data set. Color graduation indicate the importance of the features (red - important; blue - non important). Left map : lwd-SOM, right map: lwo-SOM

After applying the scree test, we obtain the selected variables: first cluster: {4 5 6 7 8 9 10}; second cluster: {6 7 8 9 10 11 12 13 14 15}; third cluster: {7 8 9 10 11 12 13 14 15 16 17 18 19}. So, lwd/lwo-SOM combining to acceleration test select variables for all dataset the selected features: {4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19}. The same results are also founded with supervised technique HVS [137].

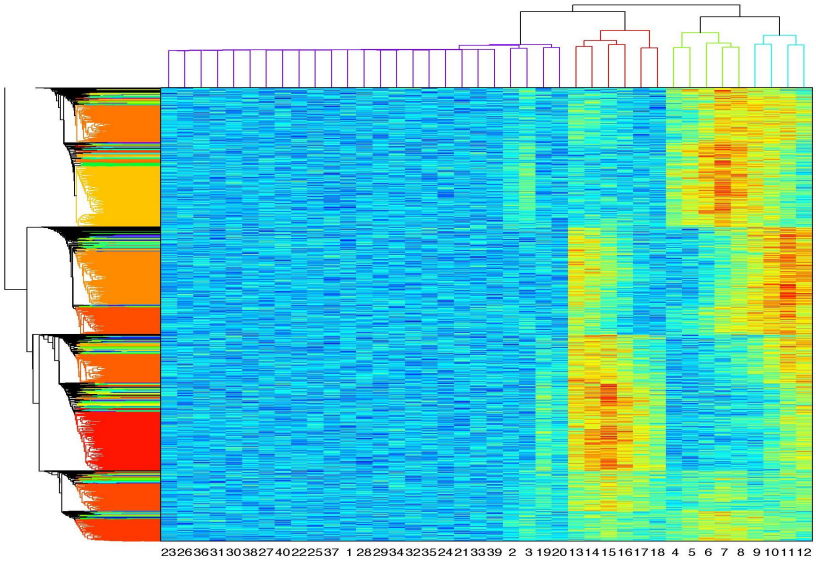


Figure 2.22: Cluster characterization of the map obtained using cross classification of waveform dataset.

2.6.5 Results on others datasets

We tested our algorithms on additional datasets with different characteristics. To demonstrate the potential benefits of simultaneous clustering and feature weighting, we used the referent and weight vector for map clustering. For both algorithms proposed, we showed the feature selection results obtained for the Isolet, Madelon, WDBC and SpamBase datasets.

For WDBC dataset, lwo/lwd - SOM algorithms select the features 4 et 24 like the most pertinent variables with a big importance for the first and 9th class (we find 9 clusters). The approach of cross classification (Bar-Joseph et al., 2001) make possibility the visual validation of this result obtaining the same features except a less importance for class 1 (Figure 2.25). For the clustering characterization and visualization we don't use the global weighted approaches, as these ones doesn't show the local importance of each variable.

For the Isolet dataset we found out an accord of the selection of non pertinent features. The algorithms *lwo*/*lwd*-SOM associated on statistical selection test found the features in interval 300 - 500 like non important variables. This result is visual confirmed by the visualization of the cross classification of this data set.

Table 2.5 shows the comparison between the results obtained from the four datasets: we

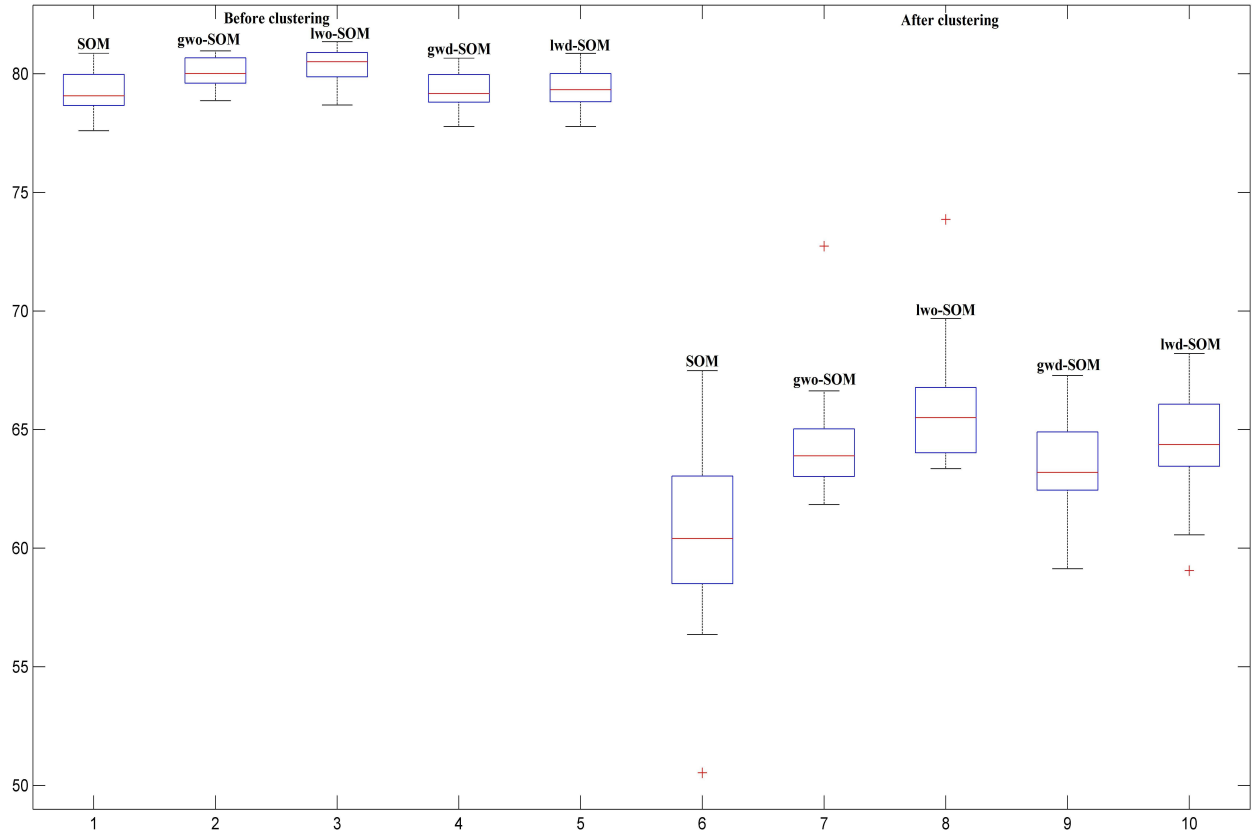


Figure 2.23: Comparison of purity score (classification accuracy with learning dataset) using SOM, *lwd*-SOM and *lwo*-SOM before and after clustering map

Table 2.4: Comparison of the selected variables using traditional and our approaches (g/lwo-SOM, g/lwd-SOM). $[i-j]$ indicates the set of selected variables

Db.	# clu-. ster	*gwd*-SOM	*lwd*-SOM using $\Pi\mathcal{W}$	*gwo*-SOM	*lwo*-SOM using $\mathcal{W}$
wdbc	2	[4;24]	cl_1-cl_9:[4;24]	[4;24]	cl_1-cl_9:[4;24]
Made-lon	2	[403-424]	cl_1:[1] cl_2: [91, 281, 403-424]	[242, 417-452]	cl_1:[1] cl_2: [242, 417-452]
Isolet	26	[1-330, 450-617]	cl_1-cl_{13}: [1-330, 450-617]	[5-302, 434-488]	cl_1-cl_{13}: [5-302, 434-488] [545-551, 586-593]
SpamB	2	[56,57]	cl_1: [56]; cl_2:[57]	[56,57]	cl_1:[56]; cl_2: [57]

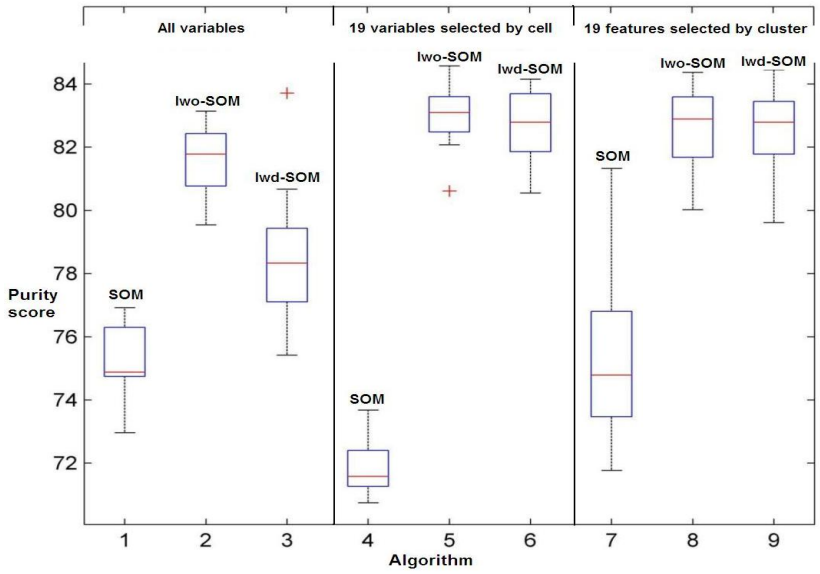


Figure 2.24: Comparison of purity score using SOM, g/lwd-SOM and g/lwo-SOM before and after clustering map

compared the characteristics of all clusters for each dataset and found that our two methods *lwo*-SOM and *lwd*-SOM provided similar results. We recall that *lwo*-SOM and *lwd*-SOM characterize clusters in an automated unsupervised manner. Validation of the results obtained was difficult because these methods used unsupervised learning. Therefore, we compared our methods using supervised validation techniques.

Table 2.6.5 provides a comparison of the accuracy of classification for various datasets after running a 3-fold cross-validation five times. We compared different situations in which the features were selected using our methods (*lwo*-SOM, *lwd*-SOM) or classical SOM. We found that our methods performed better than SOM for the various situations (using all features, features selected by cell and features selected by cluster). In all cases local weighting observation *lwo*-SOM gave a significantly higher classification accuracy than other algorithms. Means and standard deviation (SD) for the accuracy index values were computed for 15 independent runs.

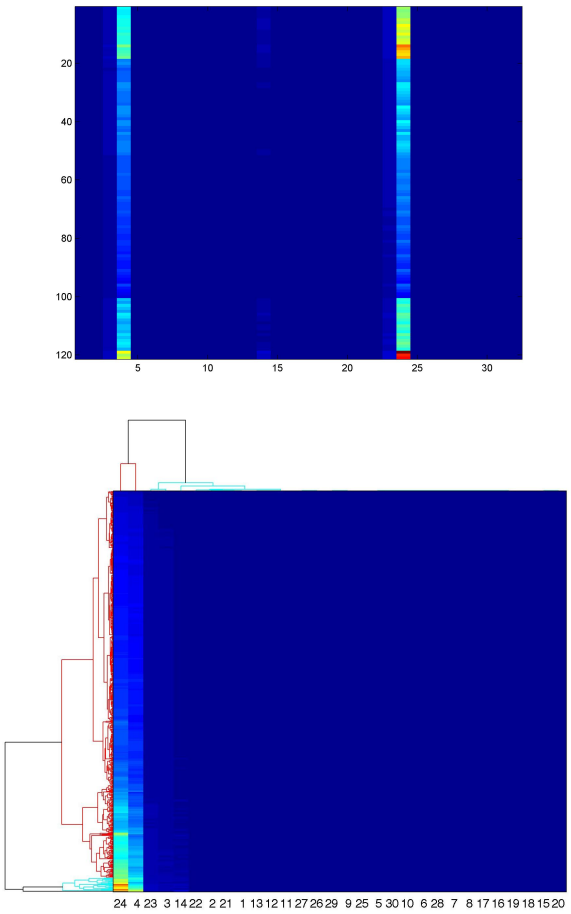


Figure 2.25: WDBC dataset: Cluster characterization of the map obtained using $lw(o/d)$-SOM after hierarchical classification (left figure) and using cross classification (right figure).

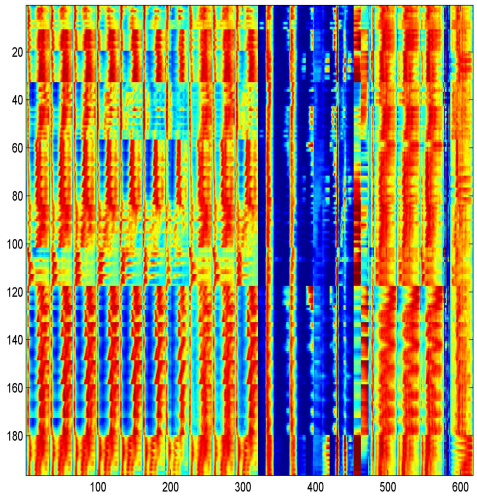


Figure 2.26: Cluster characterization of the map obtained using $lw(o/d)$-SOM after hierarchical classification on Isolet dataset

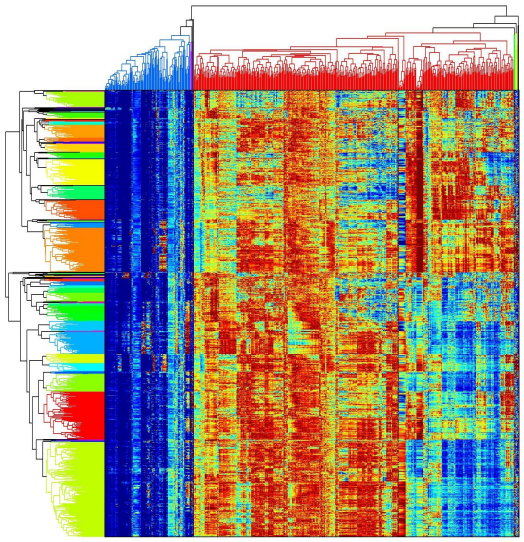


Figure 2.27: Cluster characterization of the map obtained using cross classification on Isolet dataset.

Table 2.5: Comparison of purity score with ±SD after running a 3-fold cross-validation five times (15 runs for each). b/a - before and after segmentation; Sel f. - selected variables by the cell; Sel cl. - selected variables by cluster

Db.	b/a sel/cl	method		
		SOM	*lwo*-SOM	*lwd*-SOM
Isolet	b.	0.779±0.05	0.802±0.04	0.78±0.047
	Sel f.	0.741±0.052	0.786±0.043	0.761±0.041
	Sel cl.	0.679±0.061	0.782±0.047	0.781±0.048
wdbc	b.	0.894±0.042	0.92±0.037	0.905±0.041
	Sel f.	0.892±0.047	0.915±0.04	0.902±0.043
	Sel cl.	0.891±0.046	0.915±0.041	0.901±0.042
Spam	b.	0.861±0.041	0.867±0.041	0.857±0.043
	Sel f.	0.858±0.039	0.875±0.04	0.873±0.043
	Sel cl.	0.618±0.044	0.856±0.041	0.853±0.042
made-lon	b.	0.654±0.041	0.68±0.04	0.676±0.039
	Sel f.	0.661±0.038	0.702±0.041	0.691±0.04
	Sel cl.	0.6524±0.052	0.716±0.042	0.709±0.047

Table 2.6: Global Weighting - Comparison of purity score with ±SD after running a 3-fold cross-validation five times (15 runs for each). b/a - before and after segmentation; Sel f. - selected variables by the cell; Sel cl. - selected variables by cluster

Db.	method		
	SOM	*gwo*-SOM	*gwd*-SOM
Isolet	0.779±0.05	0.785±0.04	0.762±0.042
wdbc	0.894±0.042	0.867±0.041	0.874±0.048
Spam	0.861±0.041	0.845±0.043	0.753±0.023
madelon	0.654±0.041	0.598±0.05	0.549±0.046

Table 2.7: Selected variables with *lwo*-SOM technique compared to the bi-clustering technique (clustergram)

Database	Nb. of clusters	bi-clustering	lwo-SOM
waveform	3	cl1: [3-12] cl2: [7-15] cl3: [10-18] cl4:[5-17]	cl1: [4-10] cl2: [6-15] cl3: [7-19]
Wisconsin D. Breast Cancer	2	cl1-5 [4;24]	cl1-9 [4;24]
Madelon	2	cl1: [445-450] cl2: [450-460]	cl1:1 cl2: [91, 281, 403, 424]
Isolet	26	cl1-15: [1-300, 450-620]	cl1-13: [1-330, 450-617]
SpamBase	2	cl1:56 cl2:57	cl1: 56 cl2: 57

2.6.6 Evaluation of the double local weighting SOM (*dlw*-SOM)

We show here the impact of the simultaneously observations and distance weighting (*dlw*-SOM) on the waveform dataset. The prototypes of the map changes and one can see that the variables [1 – 20] has a bigger importance compared to variables [21 – 40] that means that *dlw*-SOM detect easy the relevant variables. Also, compared to *lwo*-SOM prototypes and *lwd*-SOM prototypes for the same dataset (Figure 2.19(b) and 2.20(b)) in the figure 2.28 one can detect the impact of the distance weighting (form of the weights) and of the observations weighting (filtering the observations) because the highest values of the prototypes are not dense.

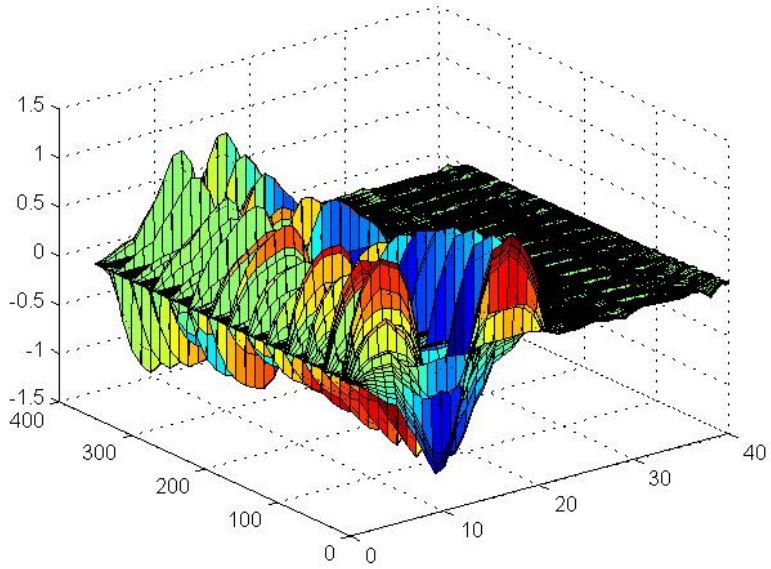


Figure 2.28: double local weighted SOM : prototypes of the map

Analyzing the distance weights (Figure 2.29) we can detect the influence of the observations weights (Figure 2.30) on the distance: the distance weights is bigger where observations weights are more important. The distance weights depend on the observations weights because the observations computation phase is doing before the assignment of the distance weights during the learning process.

In order to attempts the cluster characterization using the *dlw*-SOM map we apply the feature selection ScreeTest on the prototypes matrix. The test shows that the relevant features are [2 – 21] and irrelevant are 1 and [22 – 40] which are representatives for waveform dataset.

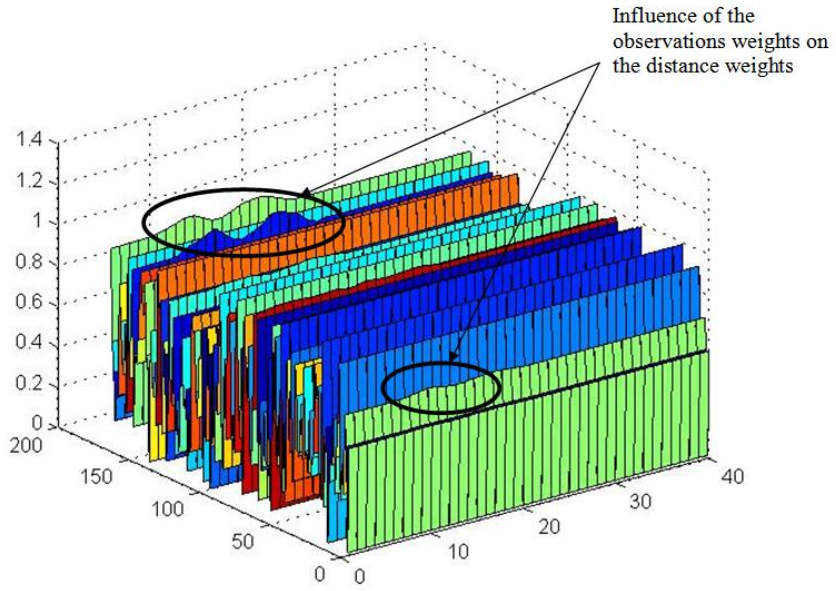


Figure 2.29: double local weighted SOM : distance weighting matrix

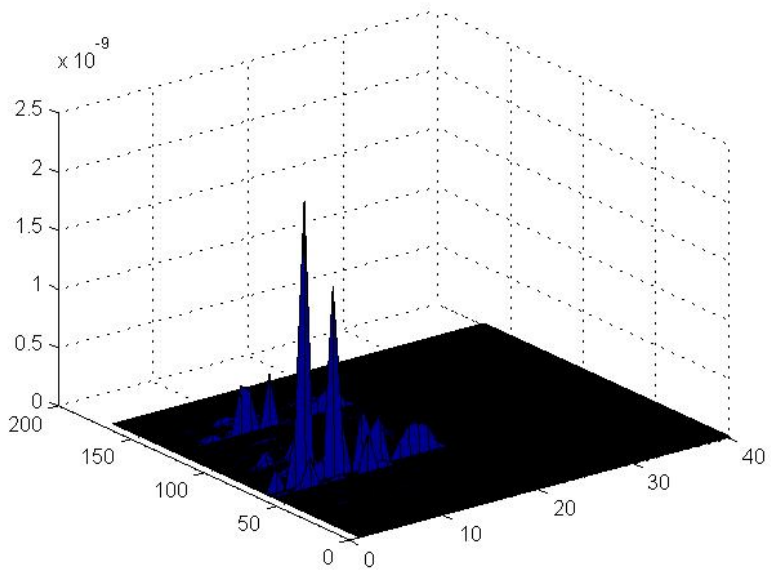


Figure 2.30: double local weighted SOM : observations weighting matrix

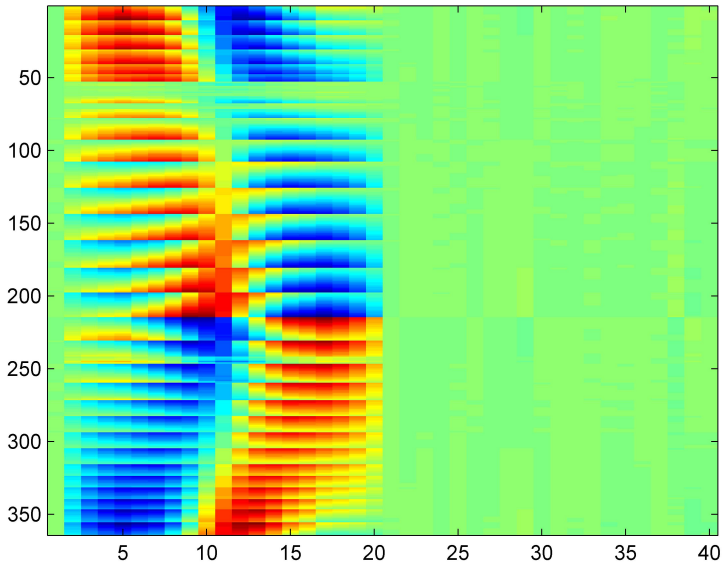


Figure 2.31: Cluster characterization using double weighting SOM

The figure 2.31 shows the relevance of each corresponding cluster of the map. Compared to *lwo*-SOM and *lwd*-SOM, we can detect easily the form of waves and the variable's relevance for each prototype a more clearly in this case. This is due to the combination of the both local weights.

2.7 Conclusion

In this chapter, we introduced new learning approaches that takes into account all neighborhood information during the competition phase and thus allows the memory learning. We used the SOM algorithm as a basic learning algorithm by introducing a voting matrix for taking into account the competition history. The different topographic errors that we have calculated on several maps show a strong improvement in the quality of the topological maps retaining a good distortion. Also, the segmentation of the maps with vm-SOM offers finer (homogeneous) clusters compared to conventional SOM.

Also, we have described a process for selecting relevant features in unsupervised learning paradigms using these new weighted approaches. These new methods are based on the SOM model and feature weighting. Both learning algorithms *lwo*-SOM and *lwd*-SOM provide cluster characterization by determining the feature weights within each cluster. We described extensive testing using a novel statistical method for unsupervised feature selection. Our approaches demonstrated the efficiency and effectiveness of this method in dealing with high dimensional data for simultaneous clustering and weighting. The models proposed in this chapter were tested on a wide variety of datasets, showing a better performance for the *lwo*-SOM algorithms than for the *lwd*-SOM or classical SOM algorithm. We also showed that through different means of visualization, *lwo*-SOM and *lwd*-SOM algorithms provide various pieces of information that could be used in practical applications.

The global weighted approaches are used in the case of analysis of the entire clustering result and not for each cluster separately. The last proposed weighted method is the double local weighted SOM (*dlw*-SOM) which allow us to weight the observations and the distances simultaneously and avoid the user to choose the confidence criteria for the weighted approach: observations or distances during the learning process.

This work offers several perspectives for future work. We can extend these models to take into account possible correlations between features and the robustness to noise. We can also, extend the algorithms to treat other types of features (categorical, mixed features) using an appropriate measure or distance. For the *wm*-SOM approach, one of the perspectives is to use the estimated vote matrix during the learning process, to build a neighborhood matrix that can serve as input to clustering algorithms.

Chapter 3

Modular and Hybrid Clustering

3.1 Introduction

As is mentioned in Chapter 1, in literature exist different clustering algorithms and new clustering algorithms continue to appear. But, many of clustering algorithms require additional user-specified parameters, such us the optimal number of clusters (k-means), stopping criteria, similarity parameters, smoothing parameter (β in *lwd*-SOM or *gwd*-SOM), and others parameters. There are also, some algorithms which use random initializations and due to this they give different results for each replication, so there is no clear indication for the best partition result. So, it is evident, that a combination scheme between clustering algorithms could give a better result: number of clusters, purity of segmentation, etc. To resolve this challenge, some approaches for Multi-Clustering Fusion Scheme were proposed in literature as [47, 123, 109]: "Consensus Models", "LSEC - Least Square Error Combination", "HCE - Heterogeneous clustering ensemble", etc.
In this chapter we propose a Multi-Clustering Fusion Scheme based on Relational Analysis [91] and *lwo*/*lwd*-SOM.

The aim of this chapter is to expose an original approach to merge different partitions, related to the same data set, which are obtained either by applying different clustering techniques or by the same clustering technique with different parameters. Fusing partitions has been broadly studied and has been given several names, depending on different scientific fields, like machine learning or bioinformatics [34, 95, 73]. Among these names we can quote: consensus clustering, clustering aggregation, clustering combination, fusion of clustering, ...etc. Several studies [125, 124, 44, 122, 93] have pioneered clustering data sets as a new branch of the conventional clustering methodology. In [124] the authors propose a

probabilistic formalism of clustering concensus using a finite mixture of multinomial distributions in a space of clustering. The approach proposed in [44] is designed for combining runs of clustering algorithms with the same number of clusters. In [122] the authors proposed combiners based on a hyper-graph model to solve the cluster fusion problem. The authors discuss two manners of consensus clustering: (1) Feature Distributed Clustering (FDC): a set of clustering are obtained from partial view of variables using all observations (2) Object-Distributed Clustering (ODC): with this technique the ensemble clustering has limited to subset of observation with access to all variables. The authors provides three techniques (CSPA[1], HGPA[2], MCLA[3]), but indicate that HGPA delivers poor scores for the both data sets used in this chapter. Our work is in FDC category. In [10] authors propose a modification of k-means for clustering a multiple runs of k-means. It's named intelligent k-means, which is especially defined for clustering ensembles. All these models assume that the correct number of clusters is given as parameter of model. In [49] the authors give a formulation of ensemble clustering titled clustering aggregation, which does not require a number of clusters. The authors give a nice review of algorithm dedicated to ensemble clustering.

In this chapter, we offer a representation of consensus clustering as a set of new variables characterizing the observations. This leads to a formulation of the fusion problem as categorical clustering problem. We propose to use *Relational Analysis (RA)* as consensus method for unsupervised learning. The consensus clustering is provided as solution of the minimization of the objective function for a given consensus clustering. The main idea, shared with other algorithm is : If many clustering algorithms assign two observations in the same cluster, it will not benefit to consensus clustering to split these observations [14].

There are several advantages of RA consensus function: first we have low computational complexity, and second ability to deal with huge data set. Another purpose of our algorithm is not to neglect the weak clustering result. Often in the ensemble/aggregation/fusion clustering we combine only the best results. Given observations and m clustering result proposed with categorical variables, the purpose is to produce a single clustering that agree as much as possible with all results of clustering algorithms. The algorithm we propose for the problem of consensus clustering take advantage of statistical formulation.

Another problem for clustering algorithms is the deal with multimodal datasets (images, text, video, sound) which are widely requested by users, and there is a strong need for effective ways to process and to manage it, respectively.
Concerning the images libraries, most of existed algorithms/frameworks are doing only im-

[1]Cluster-based Similarity Partitioning Algorithm
[2]HyperGraph Partitioning Algorithm
[3]Meta-Clustering Algorithm

ages annotations and the search is doing by this annotations, or combined with some clustering results, but they do not allow a rapid browse of these images. Even if the search is very quickly, but if the number of images is very large, the system must give the possibility to the user to browse this data. In the context of this problem, we propose in this chapter a multimodal fusion schema using a real images dataset (wikipedia).

As the Relational Analysis approach doesn't allow a topological view of the clustering results, for the hybrid clustering we propose an iterative algorithm associated with this model, which, contrarily to the SOM technique, the proposed RTM (Relational Topological Map) has the advantage of not setting a priori the number of neurons and helps to develop the topological map in automatically adjusting its size.

The rest of the chapter is structured as follows: In section 3.2 we describe in detail the proposed model for consensus clustering. In section 3.2.2 we present a special case of global fusion based on self-organizing map and we present experiments on public data set in the section 3.2.3. In section 3.3 we introduce the proposed RTM hybrid clustering method.

3.2 Modular Topological Clustering

As mentionned, it is well-know that there is no perfect clustering method, some clustering approaches could give better results than other clustering methods depending on the dataset's size, features distribution, correlation between the observations, and vice versa. For classification or regression problems, it has been shown that the gains, from using modular clustering approaches (ensemble methods), are directly related to the amount of diversity among the individual component models. Strehl et al. [121] shown that combining multiple clustering methods to find a consensus at the end of the learning can be used to introduce robustness, speed-up superlinear clustering algorithms, and dramatically improve 'sets of subspace clusterings' for a large variety of domains. In the context of this motivation we will adapt proposed weighted SOM approaches and Relational Analysis technique to a new clustering fusion schema.

Relational Analysis in the context of a modular clustering can be applied in various settings. Multiple runs of clustering algorithm, like self-organizing map, generate a new variables space, which is significantly better than pure or normalized variables space. Therefore, running a simple clustering algorithm on generated variables space can obtain the consensus cluster significantly better than pure observations. In this chapter we present another features of our framework:

- *Clustering categorical variable*: Consensus clustering provides a natural method for clustering categorical data.
- *Determining the correct number of clusters*: The formulation we propose don't require as parameter the number of clusters. The only parameter needed by RA is the similarity threshold.
- *Clustering mixed data* : The clustering fusion method can be particularly effective in the cases where data are defined over heterogeneous variables that contain incomparable values. We consider in this chapter a particular case with continuous and categorical variables. In such cases the data set can be divided vertically into sets of homogeneous variables. Thus we apply an appropriate clustering algorithm and then combine the individual clustering into single clustering using categorical data clustering method.
- *Clustering a multimodal dataset* (we uses an images dataset for experimentations).

3.2.1 Clustering Fusion Scheme for binary/mixed data

In order to improve the strong points of the RA technique for the clustering fusion, we introduce in this section a global fusion scheme for the mixed data. The proposed fusion schema (figure 3.1), initially, uses different clustering methods for the data adapted to them. If the dataset contains binary, numerical and mixed data, the schema will contain three different methods which are able to deal with each type of data respectively. After obtaining different clustering results for each part of the dataset (type of data) the RA technique is applied on these results and will find a consensus which will represent the clustering result for the entire dataset.

3.2.2 Example of a global fusion for mixed data

A specific SOM (Self-Organizing Map) model has been developed for mixed data using the similar cost function as the model presented in [76, 84]. The model dedicated to binary and continuous data is called MTM (Mixed Topological Map) proposed by Lebbah et al. in 2005 [84]. As with a traditional self-organizing map, we assume that the lattice $\mathcal{C}$ (map) has a discrete topology defined by an indirect graph. Usually, this graph is a regular grid in one or two dimensions. For each pair of cells (c,r) on the map, the distance $\delta(c, r)$ is defined as the length of the shortest chain linking cells r and c. Let $\mathcal{X} = \{x_i, i = 1..n\}$ the learning data set where each observation $x_i = (x_i^1, x_i^2, ..., x_i^k, ..., x_i^m)$ is made of two parts: continuous part

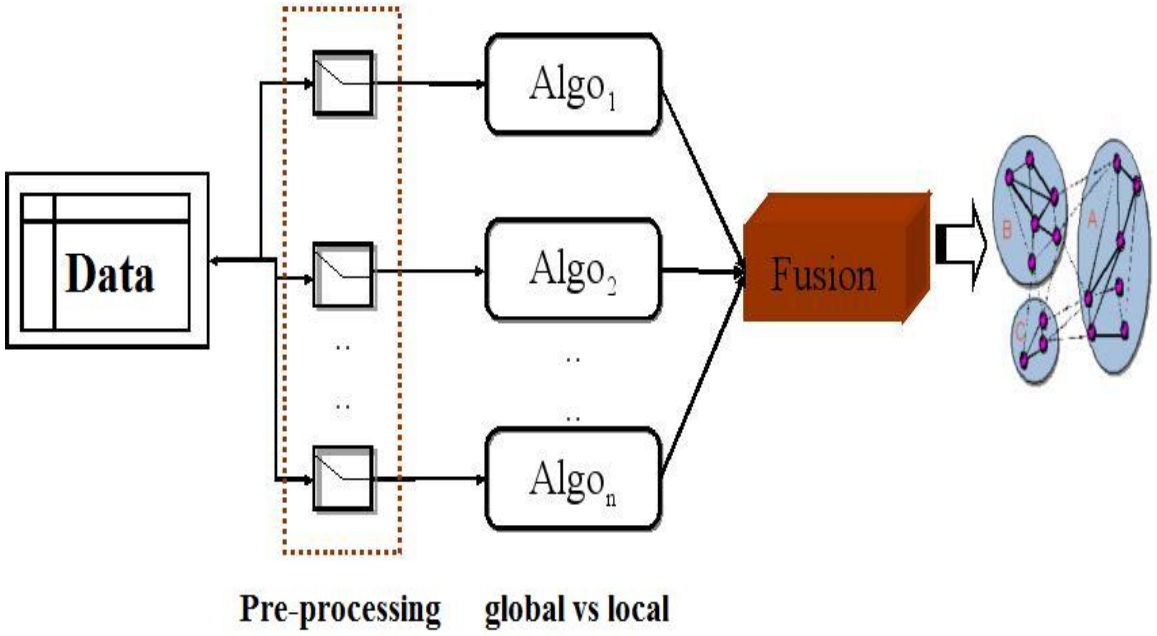


Figure 3.1: General Scheme of Fusion Clustering

$x_i^{r[.]} = (x_i^{r[1]}, x_i^{r[2]}, ..., x_i^{r[d_r]})$ ($x_i^{r[.]} \in \mathcal{R}^{d_r}$) and binary part $x_i^{b[.]} = (x_i^{b[1]}, x_i^{b[2]}, ..., x_i^{b[k]}, ..., x_i^{b[d_b]})$ where the kth component $x_i^{b[k]}$ is binary variable ($x_i^{b[k]} \in \beta = \{0, 1\}$) such as each observation x_i is thus, a realization of a random variable which belongs to $\mathcal{R}^{d_r} \times \beta^{d_b}$. With these notations a particular observation $x_i = (x_i^{r[.]}, x_i^{b[.]})$ is a mixed subvector (continuous and binary variables) of dimension $m = d_r + d_b$.
Since for binary vectors the Euclidian distance is no more than the Hamming distance $\mathcal{H}$, then the Euclidian distance can be rewritten by:

$$||x - \mathbf{w}_c||^2 = ||x^{r[.]} - \mathbf{w}_c^{r[.]}||^2 + \mathcal{H}(x^{b[.]}, \mathbf{w}_c^{b[.]})$$

where $\mathcal{H}(x^{b[.]}, \mathbf{w}_c^{b[.]})$ the complement of global similarity between a binary part of an observation x and referent $\mathbf{w}_c^{b[.]}$).

Using this expression, the cost function of the traditional SOM algorithm which is dedicated to mixed data can be expressed as:

$$\begin{aligned} \mathcal{G}(\phi, \mathcal{W}) &= \sum_{x_i \in \mathcal{P}} \sum_{r \in \mathcal{C}} \mathcal{K}(\delta(r, \phi(x_i))) ||x_i^{r[.]} - \mathbf{w}_r^{r[.]}||^2 \\ &+ \sum_{x_i \in \mathcal{P}} \sum_{r \in \mathcal{C}} \mathcal{K}(\delta(r, \phi(x_i))) \mathcal{H}(x_i^{b[.]}, \mathbf{w}_r^{b[.]}) \end{aligned} \tag{3.1}$$

Where ϕ assigns each observation x_i to a single cell in $\mathcal{C}$. $\mathcal{K}$ is a particular kernel function which is positive and symmetric ($\lim_{|y|\to\infty} \mathcal{K}(y) = 0$).

The first term is the classical cost function used by the Kohonen Batch algorithm [76], and the second term is the cost function used in BinBatch model [83]. The cost function (3.1), is minimized using an iterative process with two steps.

1) Assignment step, which leads to the use of the following assignment function:

$$\forall x,\ \phi(x) = \arg\min_{c} \left(\|x^{r[.]} - \mathbf{w}_c^{r[.]}\|^2 + \mathcal{H}(x^{b[.]}, \mathbf{w}_c^{b[.]}) \right)$$

2) Optimization step: It is easy to see that this two minimizations of two terms allow to define:

   - The continuous part $\mathbf{w}_c^{r[.]}$ of the referent vector $\mathbf{w}_c$ as the mean vector as:

$$\mathbf{w}_c^{r[.]} = \frac{\sum_{x_i \in \mathcal{P}} \mathcal{K}(\delta(c, \phi(x_i)))x_i^{r[.]}}{\sum_{x_i \in \mathcal{P}} \mathcal{K}(\delta(c, \phi(x_i)))},$$

   - The binary part $\mathbf{w}_c^{b[.]}$ of the referent vector $\mathbf{w}_c$ as the median center of the binary part of the observations $x_i^{b[.]} \in \mathcal{P}$ weighted by $\mathcal{K}(\delta(c, \phi(x_i)))$. Each component $\mathbf{w}_c^{b[.]} = (w_c^{b[1]}, ..., w_c^{b[k]}, ..., w_c^{b[d_b]})$ is then computed as follows:

$$w_c^{b[k]} = \begin{cases} 0 & \text{if } \left[\sum_{x_i \in \mathcal{P}} \mathcal{K}(\delta(c, \phi(x_i)))(1 - x_i^{b[k]})\right] \geq \\ & \left[\sum_{x_i \in \mathcal{P}} \mathcal{K}(\delta(c, \phi(x_i)))x_i^{b[k]}\right] \\ 1 & \text{otherwise} \end{cases},$$

3.2.3 Experimental evaluation

In the following, the RA is used as the clustering consensus/fusion based algorithm for categorical and mixed data. First, the original data set is divided into two sub-data sets: pure categorical data set and pure continuous data set. Next, existing well established clustering algorithms designed for different data types are employed to provide corresponding clusters. We can run many algorithms or the same with different parameter using the same data. Finally the clustering results are combined as categorical data set to provide a consensus single clustering.

As quality evaluation criterion we use purity index. However, when class labels are available for each observation, we can use purity measure to indicate the match between cluster labels and class labels. The purity assess clustering quality from 0 (worst) to 1 (best).

Artificial data sets for fusion

We illustrate the cluster consensus applications on two artificial data sets downloaded from *http://strehl.com/* and used by [122] : 2D2K and 8D5K.

For this experiment we take several clustering results provided by Strehl in his website *http://strehl.com/*. The authors give two simulations of clustering ensemble: (FDC, Exp1) Feature-distributed Clustering (ODC, Exp2): Object-distributed Clustering. Table 3.1 indicates different results provided by Strehl and Ghosh adding the result obtained with our consensus clustering technique RA in both experimentations. Our purpose through this comparison, is not to assert that our method is the best, but to show that RA method can obtain quite the same good results as the two previews ones, without making any arbitrary assumptions about the number of clusters to be found. Indeed, as shown in the table bellow, we can see that RA method give similar results and quite comparable to the ones obtained by both proposed techniques (FDC, ODC). The main difference between these three methods is that, unlike the two other methods, RA doses not require a priori knowledge of the number of clusters.

8D5K		RA
FDC (Exp1)	0.9970	0.9930
ODC (Exp2)	0.9480	0.9330

2D2K		RA
FDC (Exp1)	0.9440	0.9440
ODC(Exp2)	0.9680	0.9700

Table 3.1: Comparison of consensus clustering. FDC: Feature-Distributed Clustering; ODC: Object-Distributed Clustering; RA: Relational Analysis; Exp: Experimentation

In the first experiment we simulate such clustering result by running several clustering algorithms, each one having access to only a restricted categorical or continuous variables. Thus, each clusters has a partial view of the observations. The clusters are found using subspaces and adapted clustering technique. In the consensus clustering, cluster labels are clustered using RA technique. In order to compare our result, we cluster the data using the MTM method which provides a small cluster organized as map (see section 3.2.2). Often we use hierarchical clustering to reduce the number of the clusters [131]. The combining method is indicated by MTM+HC and the number of clusters between bracket.

The figures 3.2 and 3.3 show the comparative results in term of number of clusters and the purity index. As can be seen, the both figures indicate that RA provides the high scores when

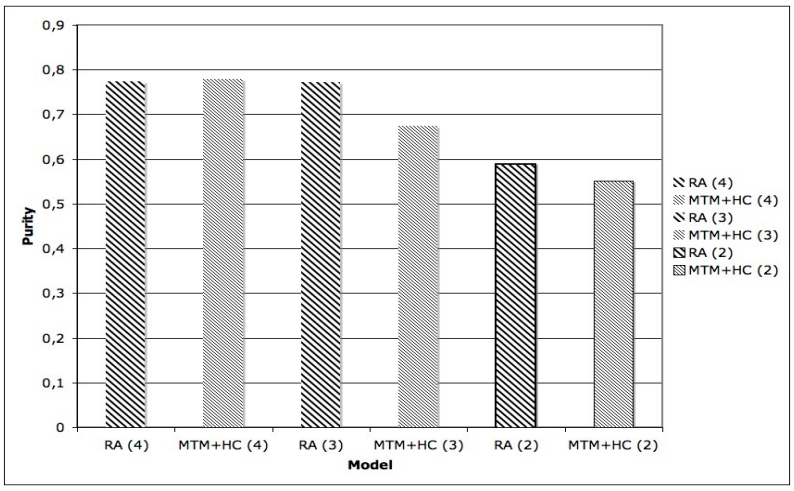


Figure 3.2: Credit data set. Purity scores for consensus clustering. RA : Relational Analysis; MTM: Mixed Topological Map. HC: Hierarchical Clustering. The number between brackets indicates the number of clusters provided automatically

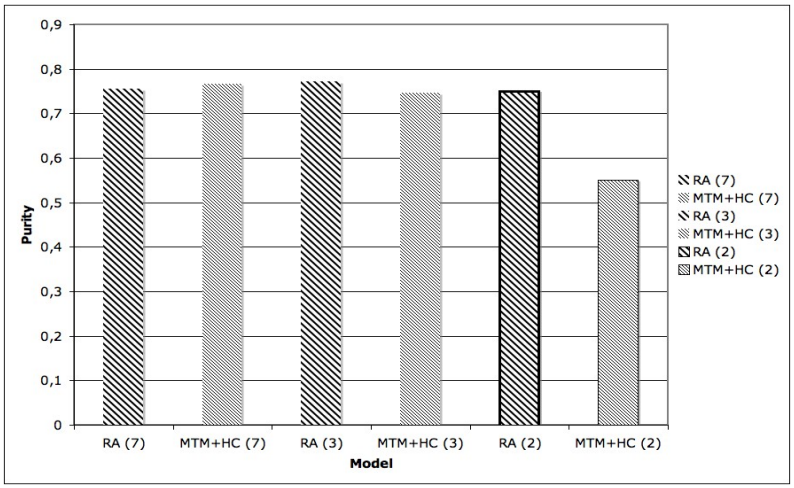


Figure 3.3: Heart disease data set. Purity scores for consensus clustering. RA : Relational Analysis; MTM: Mixed Topological Map. HC: Hierarchical Clustering.The number between brackets indicates the number of clusters provided automatically

compared for the same number of clusters. Note in this case for the both data set we have, a priori, two classes, and the RA (2) provides high purity for this case. We note also that RA don't require two steps of clustering, comparing to the MTM and other clustering ensemble algorithms found in the literature which needs an agglomerative clustering technique to reduce the number of clusters. The only parameter needed by RA is the similarity threshold.

In this second experiment we use Handwritten data set. The purpose is to use RA as consensus clustering of several runs of the same clustering algorithm. In this case we simulate

16 cluster results obtained with Self-organizing map dedicated to categorical data and hierarchical clustering, using different parameters [84, 131]. We use 5 cluster results with purity score lower than 0.4, and 4 results lower than 0.72, and the rest results are between 0.74 and 0.76. Thus the RA consensus clustering provide a stable purity with 0.76.

The figure 3.4 shows the distribution of each class of digit in all 15 consensus clusters. The figure 3.5 shows the best map obtained among the 16 maps used for consensus clustering. We visualize this figure in order to interpret the results of consensus. We note that RA grouped in a cluster numbered 12, 13, 15, the mix of digit 7, 9 and digit 5. It is clear to see on the map (Fig.3.5) that some figures such as "9" are written in the same way as "5" and "7". The same analysis could be done with the other clusters.

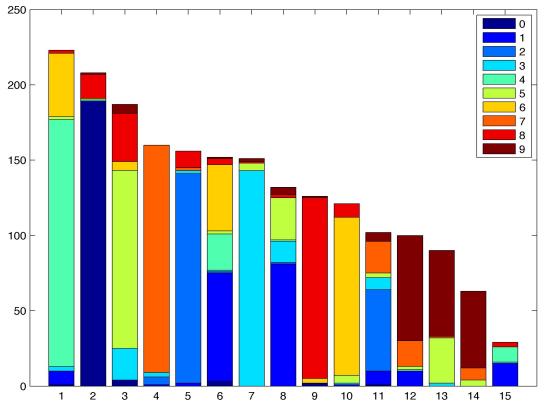


Figure 3.4: Consensus clustering with RA. Each bar shows the distribution of each cluster.

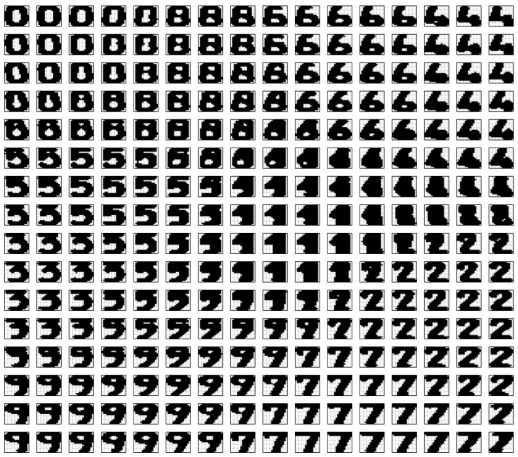


Figure 3.5: 16×16 map using MTM with only categorical data

3.2.4 The fusion technique for a multimodal dataset

The multimodal data has properties which made difficult it's exploitation using statistical classical methods:

- Hetrogenous data : some physical image characterization are associated to some words extracted from a text jointed to the image, or sound and image which came from a video, or sound and text derived from an audio record;
- The data are usually mixed : one can have at the same time quantitative data (images characterization/features), and binary data (the words which correspond to the images);
- The number of features to describe this data is usually very important/big and can be in order of some thousands features;

Images datasets are becoming more common and widely used as visual information is produced at a rapidly growing rate. Creating images and storing its became an easily and very used process for general use. Consequently, the digital visual libraries are growing and there is a strong need of adequate solutions to process this data and to extract relevant informations from it.

The traditional text-based approaches to image retrieval have proven out to be inadequate for many purposes. On some occasions, image databases have associated captions or other text describing the image content and these annotations can be used to greatly assist image search. Manually annotating large databases takes, however, a lot of effort and raises the possibility of different interpretations of the image content. As a result, content-based image retrieval (CBIR) has received considerable research and commercial interest in the recent years. One of the challenge is to automate the process of image retrieval and to make it separately from text annotation [77]. The most interesting and used technique for data reduction and visualization in machine learning are the Self-Organizing Maps (SOM). This approach was used for image retrieval system called PicSOM [80] which use the tree structured SOM (TS-SOM) [103].

We propose a novel technique which propose to use the *lwo*-SOM [102] [52] to attempt a 3D visualization and browsing of the dataset. Also, we incorporate an interactive learning approach based on the users/experts information and on the computing of the Euclidean distances matrix. This technique is similar to the annotation which was used in different way, like in [71] that compute mixture models for each image and uses the Mallows distance to construct a matrix to be used by the clustering algorithms.

Contrarily to the most of the feature extraction techniques where the main access to the images is made through query, we will use an autonomous approach which auto-organize the structure of the dataset using the learned map. The system is also capable to receive new data after the clustering and to place it in the computed space in the map.

In this work, we will study a specific image collection of 17812 images extracted from Wikipedia pages. The images are accompanied with keyword-type annotations which specify a subset of available keywords for each image.

The general schema for the fusion of images maps result is given in figure 3.6.

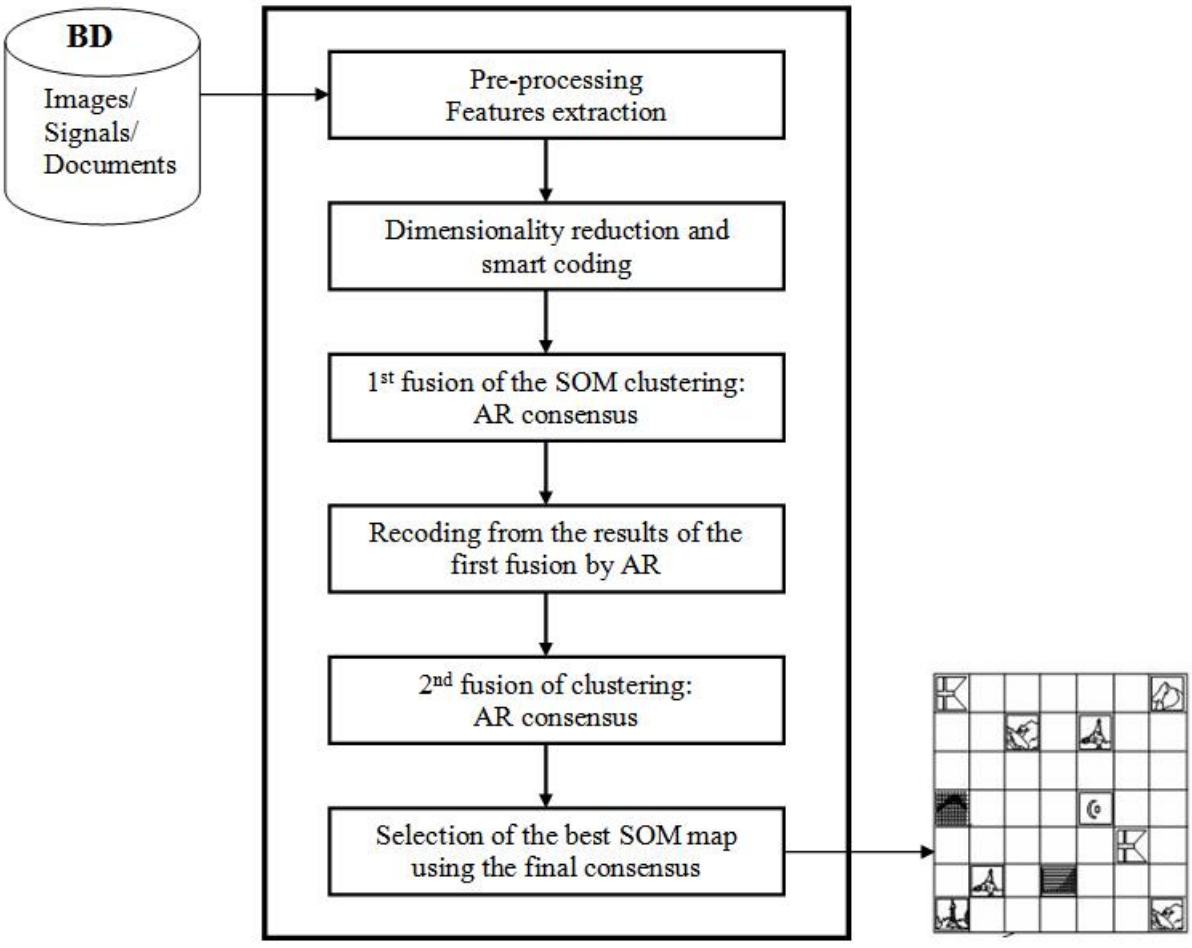


Figure 3.6: General Schema of the Fusion Procedure

3.2.5 Images and Features extraction

The images (17812) were extracted from the wikipedia web site from the tourism compartment by the Xerox Research Center [111] during the Infom@gic project. Each image has a web link and a set of words attached to it. The pre-processing technique is an important step for clustering a multimodal dataset. From a multimodal dataset we can extract different types of features as shown in the figure 3.7.

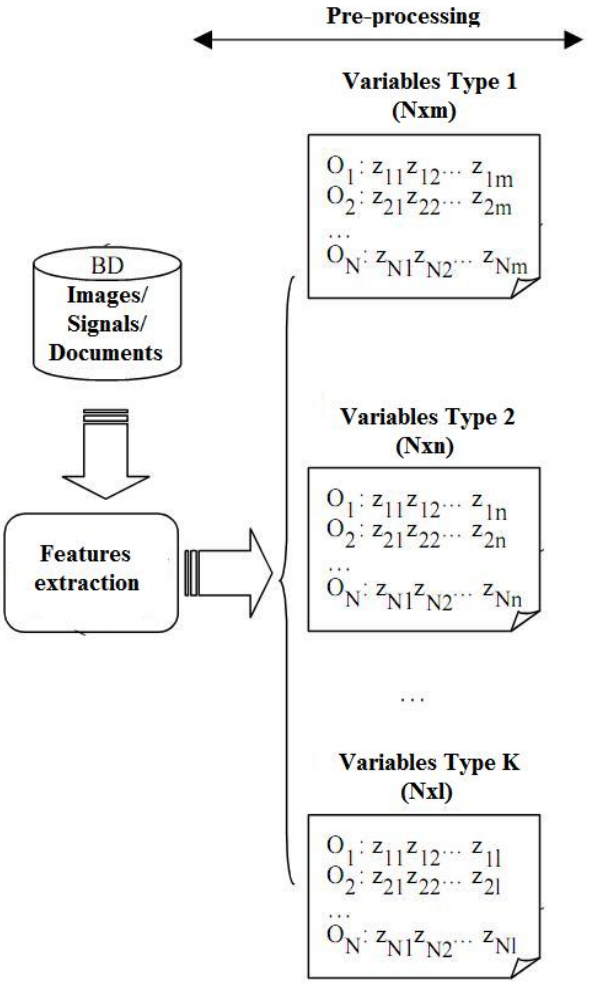


Figure 3.7: Pre-procession of the images dataset

The Fisher Kernels approach was used to obtain a numerical transformation of images. The used technique is an approach which was inspired by the bag-of-words used in text categorization referred to as the bag-of-keypatches or bag-of-visterms (BOV). Given a visual

vocabulary, the idea is to characterize an image with the number of occurrences of each visual word. The gradient of the log-likelihood transforms a variable length sample X into a fixed length vector whose size is only dependent on the number of parameters.
Perronnin F. and Dance C. proposed to apply Fisher kernels on visual vocabularies, where the vocabularies of visual words are represented by means of a GMM (Gaussian Mixture Models). All the images were resized to contain approximately the same number of pixels, so, the same number of features were extracted from all images (between 500 and 600 for each feature type). First features are based on local histograms of orientations and the seconds ones are simple local RGB statistics [111]. After obtaining these features vectors, the features dimension were reduced to 6400 using an GMM algorithm and the Fisher Kernel described bellow.
Fisher kernels have been introduced to combine the benefits of generative and discriminative approaches. Let p be a pdf whose parameters are denoted λ. Then one can characterize the samples $X = x_t, t = 1...T$ with the following gradient vector:

$$\nabla_\lambda \log p(X|\lambda) \tag{3.2}$$

We note that in this work we used the numerical data for each image, the transformation was made by the Xerox Research Center.

3.2.6 The Framework for images self-organizing map

In the next sections of this chapter concerning the modular topological learning we will use the wikipedia images dataset which will allow us to propose a new image retrieval system for browsing an images dataset or searching in it.

One of the well known browsing/search images techniques is the google search engine which uses the *ranking* method to classify the web pages containing the corresponding images. The rank of a page depends numerically on the pages that are pointing to it [46]. For a given document v from a set of documents V ($v \in V$), let $\Gamma^-(v)$ be the set of documents pointed to v, and $\Gamma^+(v)$ - the set of documents to which v is pointing respectively, the rank $\rho(p)$ of a page p is given by the following formula:

$$\rho(p) = K\left(\alpha \sum_{g \in \Gamma^-(p)} \frac{\rho(q)}{|\Gamma^+(q)|} + (1-\alpha)\rho(p)\right) \tag{3.3}$$

where K is a normalization constraint that assures that ρ is a probability, α is a weighting factor that allows to gives more importance to a page - the PageRank method. A sensitive

point of the PageRank is the importance that is given to the pages rank pointed to q, that is the value of the parameter α. Moreover, the result of a search/query will not be allways the best, because by searching an image containing a flag (*drapeau* in french) we can detect some images which doesn't correspond to the user query like some people's images which has the name '*Flag/Drapeau*'. So, making a test, searching the flags on search engine based on rankpage, we have 182000 result's images that means that the user need approximately 45 minutes to browse the resulted images which is not confortable. Based on these problems we will introduce a new original topological learning system for images datasets in order to attempt a faster browsing and a well-clustered topological map composed from images.

For each coded set of variables, all the different maps are learned (figure 3.8) in order to obtain a reduced matrix for each type of the coding technique respectively. We used the classical SOM and weighted SOM approaches presented in the chapter 2 in order to learn different maps of images (figure 3.8). So, from a dataset size 17812x6400 we reduced the dimensionality to 17812x10 where variables were recoded using cells number for each map.

After obtaining the *lwo*-SOM map, we construct the distance matrix $Nx|W|$, computing the Euclidean distance between each sample (image) $i = 1...N$ and each prototype $j = 1...|W|$ of the map:

$$\chi(D_m) = \left(||\pi_j \mathbf{x}_i - \mathbf{w}_j||^2\right)$$

The matrix D_m is sorted in order to have all the samples(images) structured by levels : from the best matching unit until the last unit. Using this sorted matrix, we can construct now the map and visualize images which correspond to these units. For each unit there is a corresponding image from the data set. Each map unit/cell has several images/samples/observations which were captured during the learning process. So we have two ways to manage this images dataset: to cluster it and to browse it, presented in the next sections.

3.2.6.1 Clustering the images

Firstly, we learn a map size 13×13 (169 cells), and secondly, for each cell we display the corresponding image from the dataset. Also, the framework give the possibility that when choosing the interest image, the map will show all others images which were captured by this cell/neuron (Figure 3.9).

On the obtained *lwo*-SOM map (Figure 3.9), we can detect 6 clusters: one in the top left corner which correspond to the clear blue images (like images which sky); a cluster

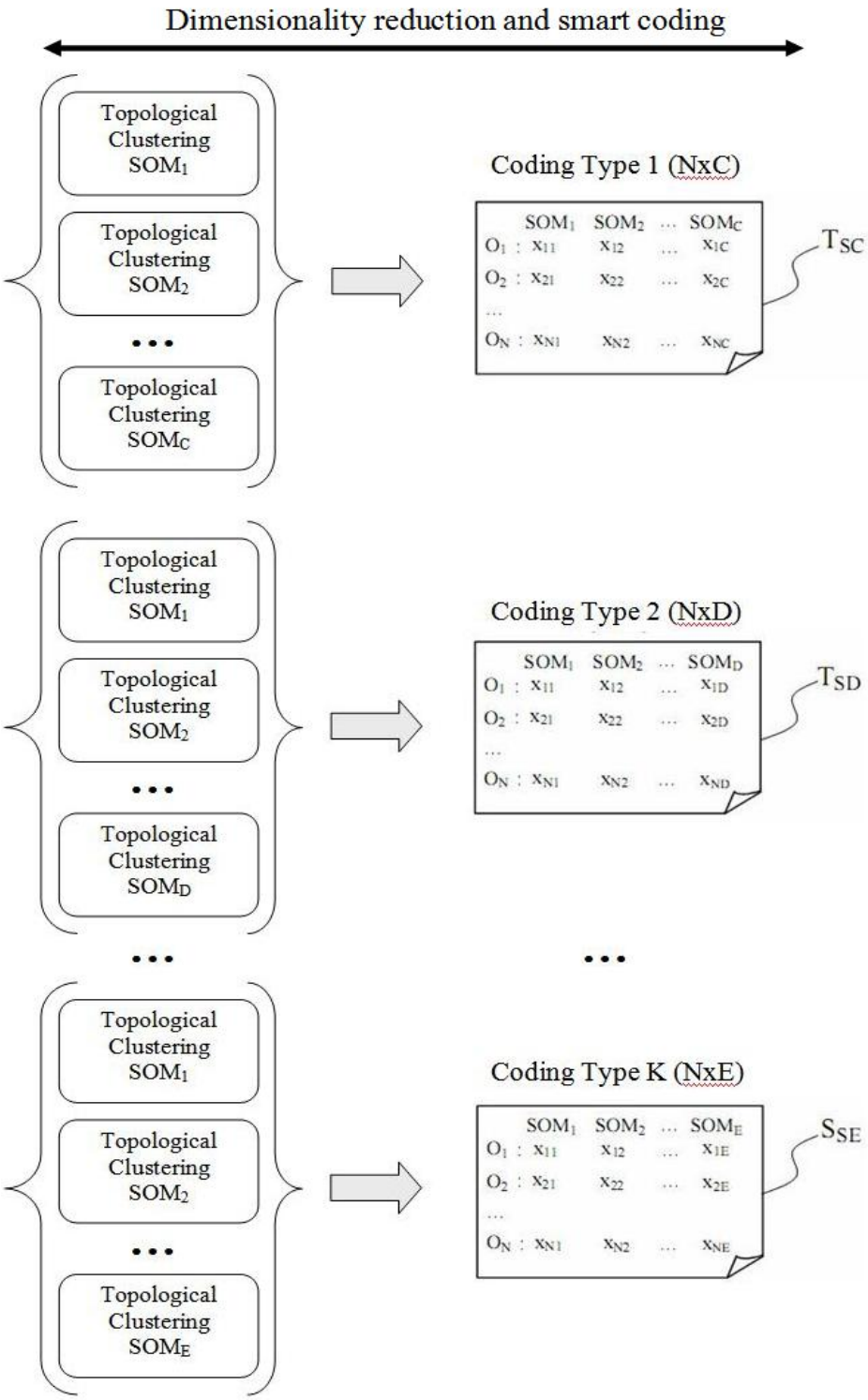


Figure 3.8: Dimensionality Reduction

Figure 3.9: Images SOM Map (13x13 size)

in the left bottom part of the map - images with the more darkly blue color. We observe that these two clusters are neighborhoods on the map because they are correlated (clear blue and dark blue). Another correlated cluster to these two is the cluster situated on the bottom which contains images with blue/black color. On the right bottom corner of the map there is a cluster which contains red/yellow images represented the flags. The right top cluster has more white images characterizing graphs and geographical maps; and the last cluster is situated in the middle of the map and is representing by brown color images. As we can see, the topology of the map is well defined and the neighborhoods cells on the map are correlated between them. This technique gives the possibility to the user to have a small-space representation for the entire dataset.

Now, we will analyze captured images by the cells 169, 91 and 32 as we can see on the figure 3.9. For the cell 169 (Figure 3.10) we can observe that all the captured images are

reds and represents flags (only one image are green). The 91th cell captured all white images representing the graphs presented in the figure 3.11.

Figure 3.10: Captured images by the 169th cell

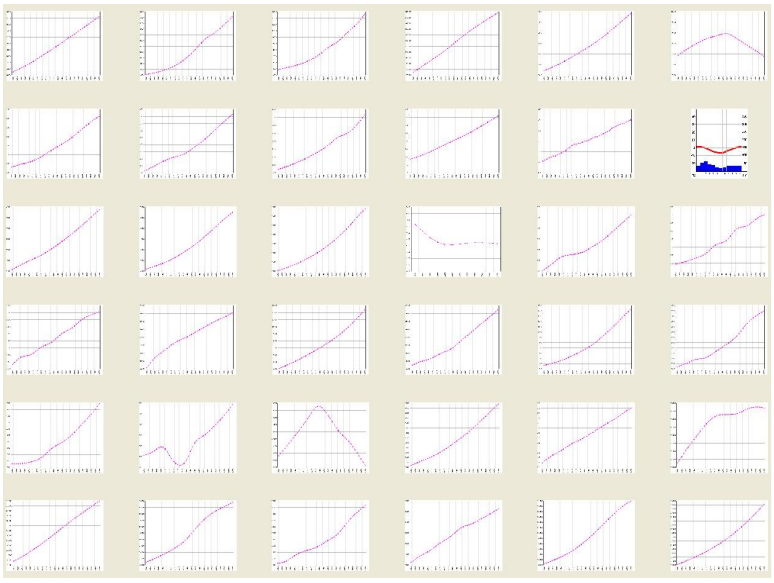


Figure 3.11: Captured images by the 91th cell

More difficult is to find the correlated images for the 32th cell there the images are not highly similar between them, this means that the expert annotation could not coincide with

our result. In this case, the system(framework) must be able to use the user/expert informations and to change its results after the learning process - the interactive learning.

Figure 3.12: Captured images by the 32th cell

3.2.7 Fusion

We use the Relational Analysis method to fusion the results, in order to obtain a consensus between all these maps.

The fusion schema (figure 3.13) is used for all types of coded variables matrix: color, texture, text or combined them as shown in the figure 3.13. The fusion's result of the obtained images maps gives a new clustering result (figure 3.14). On this new clustering result we can find out new formed clusters as black color images cluster located in the middle of the clustering. Another new cluster is the set of the green images situated on the left of the black images cluster. These new forms of clusters means that the clustering fusion technique found out new clusters, or in other words, founded the 'real' clustering result by combining multiple topological clustering approaches.

The figure 3.14 shows the clustering fusion results on the wikipedia maps. Visually, one can detect the well-defined images clustering: there are a cluster composed only from white flags, a cluster with blue images, with green images, etc.

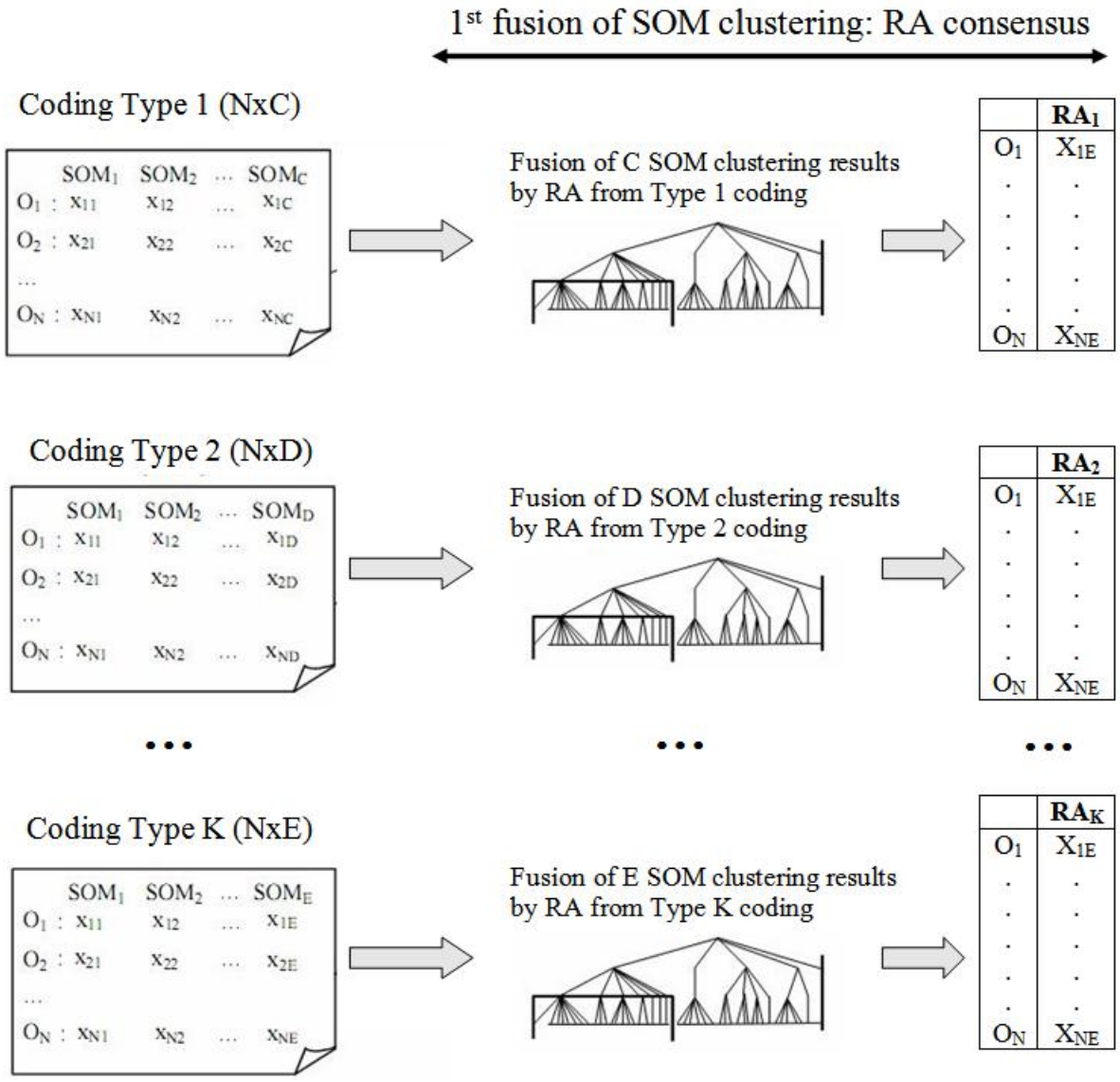


Figure 3.13: Fusion of the SOM maps

3.2.7.1 Browsing

After obtaining the RA consensus for each type of coded data, we process a second fusion presented in the figure 3.15 in order to find out the best map from the initial ones.

To browse the images dataset, we firstly visualize the map with the best matching units (the most representative images) and then, we can chose the next level to visualize (or to skip some levels) until we are satisfied of the result. This process is doing in a 3D visualization

Figure 3.14: Fusion of the wikipedia maps

by displaying the maps with the corresponding captured images step by step as shown in the figure 3.16. Most of the image retrieval systems do not support browsing, likely because it is difficult to define and implement. Rather, these systems force/ask the user to specify what they are looking for with a query. If the user's task is not compatible with the images annotations made by another users/experts, the system will not be able to help the user learn what kind of images can be found. So, the problem, in this case is the text annotations, and the no-organizations between images during this task. Our approach, is to automate the browsing task using not only the annotated text, but also the similar images founded during the unsupervised learning. The idea, is to present an images map to the user in order to detect not only the searched image, but also the similar images from the map (neighborhoods cells using the Euclidean distance) (Figure 3.16(a)). Furthermore, a cell from the map (the best matching unit) can be used to represent many others similar pictures, and will accurately

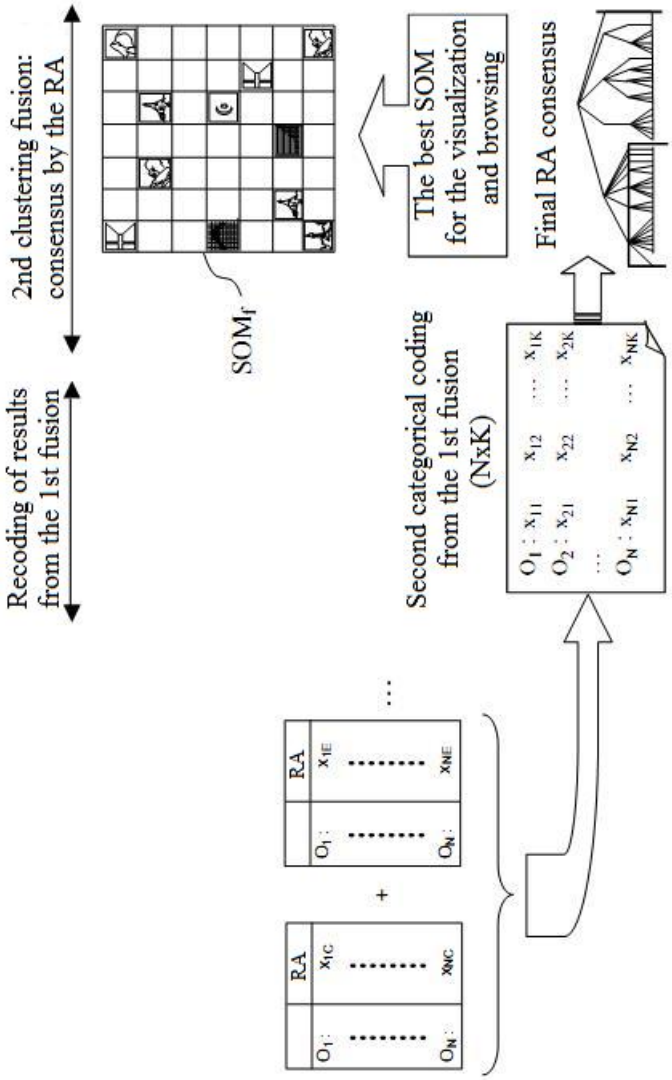
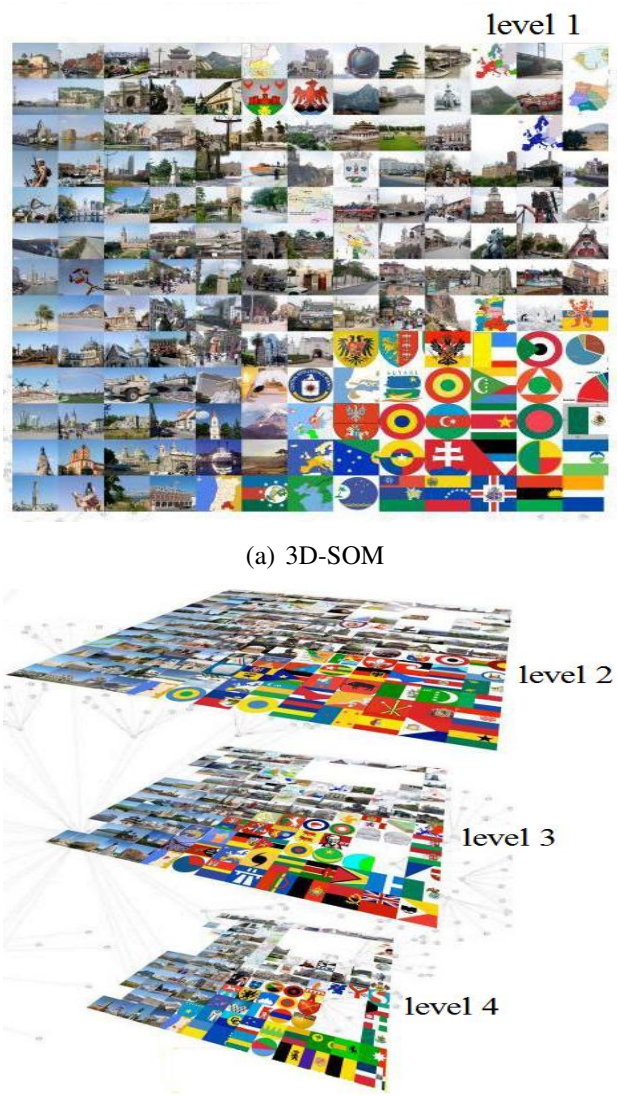


Figure 3.15: Fusion of the RA clustering results

suggest the kinds of pictures that will be found by exploring that cluster.
The figure 3.16 shows the map with the best matching units (first level), and the next 3 levels of the maps. For each map the neighborhoods displayed images are correlated between them, and one can detect also some cells which are empty, because there are cells which captured only 1, 2, or 3 images. So displaying the map level which is greater than the size of the captured images vector for a cell, the respective cell will display an empty (white) image to show that there are no more correlated images to the last one.

The figure 3.17 shows the best map of images (*lwo*-SOM map). The 3D blue bars show



(a) 3D-SOM

Figure 3.16: Images dataset browsing using *lwo*-SOM technique. Four levels of the 3D map.

the approximative dimension of each cell of the map. Images in each map's cell are sorted by means of the Euclidian distance in order to display foreword the most relevant images. Dropping on the others map's levels, the map could have empty cells as not all the cells captured the same number of observations (images) as is displayed in the figure 3.18 and 3.19.

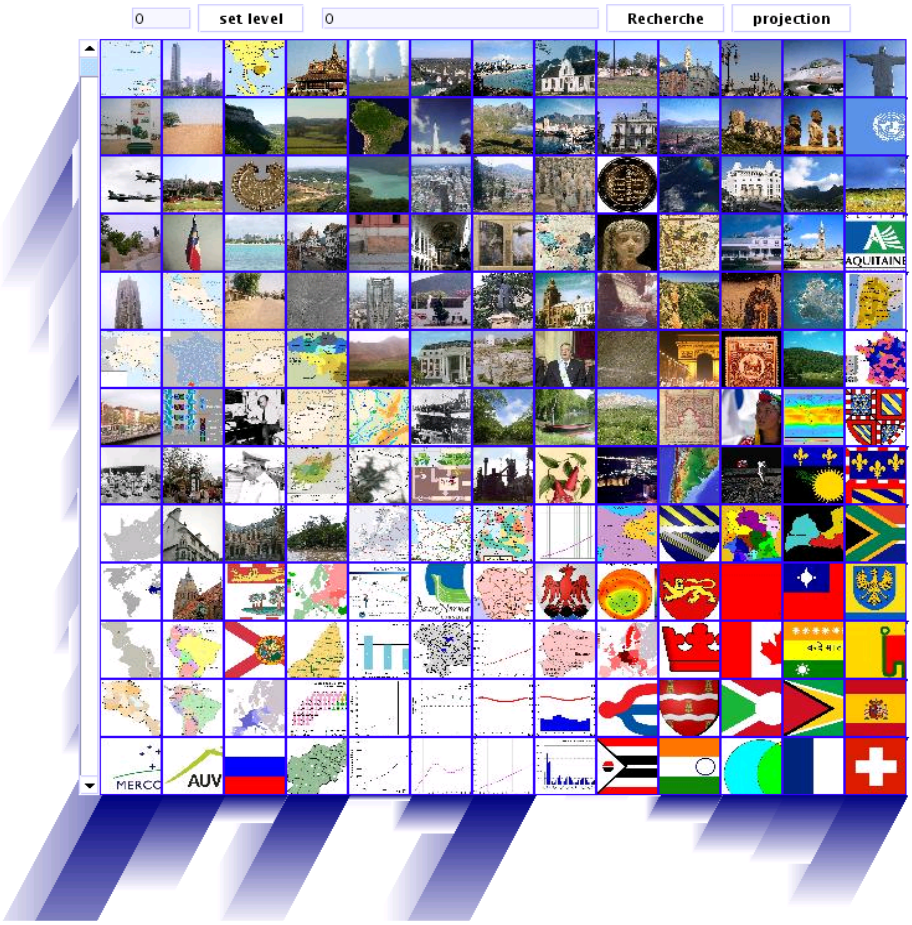


Figure 3.17: Browsing: level 0

Fusion of the weighting methods l/gwo-SOM and l/gwd-SOM

In order to find the best structured map obtained with the classical SOM, *lwd*-SOM and *lwo*-SOM on the wikipedia dataset we use the Relational Analysis technique which compute the contribution measure for each map and we obtain the following results:

- For the classical batch SOM the contribution is 0.874;
- For the *lwd*-SOM : 0.914;
- Finally, for the *lwo*-SOM this index increases to 0.979.

As we can see, the *lwo*-SOM approach provides a higher contribution measure, and respectively this map highly corresponds to the fusion results compared to classical SOM and to *lwd*-SOM. This results confirm the performance of the weighting process during the map learning.

Figure 3.18: Browsing: level 120

The figure 3.19 shows one of the last level of the *lwo*-SOM map and, as, we can see, even in this case there are two correlated images on the corresponding map's level. These images are situated in the middle of the map. The browsing process shows against the strong importance of the topological knowledge for the learning/clustering task.

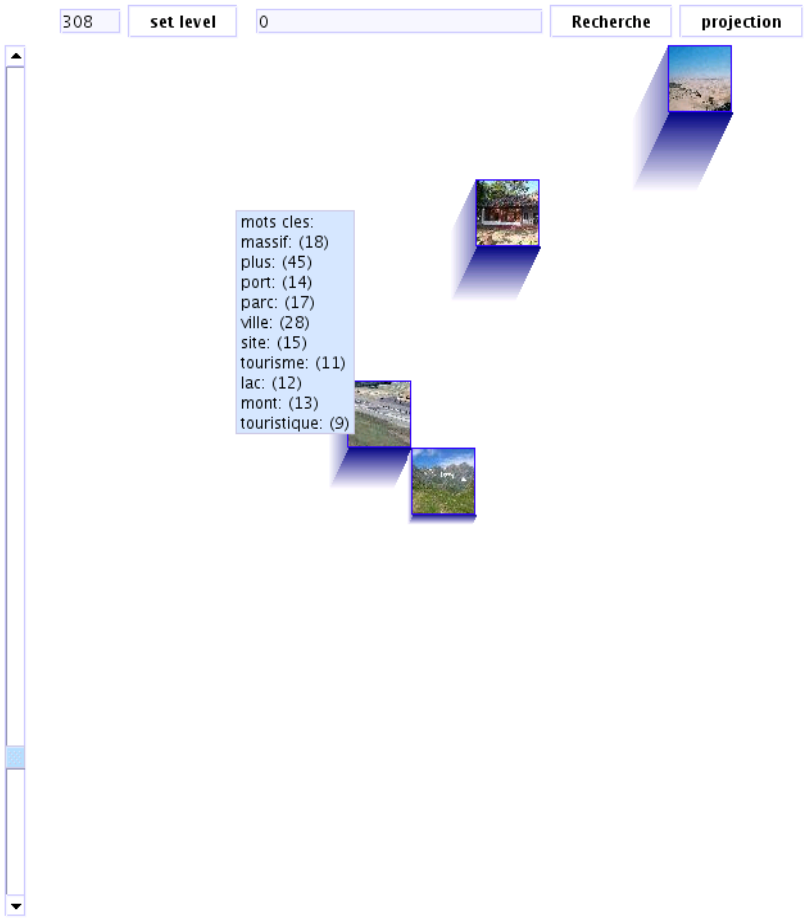


Figure 3.19: Browsing: level 308

3.2.7.2 Search

A second important facility for the images libraries is the images retrieval based on user queries. In literature exists many algorithms to resolve this problem, like computing the probability of each candidate image of emitting the query items, computing the images rank, and many others. Our goal is to use the attached images words, but instead of displaying all the images which has this word jointed, our system will display to the user, the level of the map where is situated the searched image. With this, the user will have the possibility to choose another correlated image even if it doesn't has the searching word corresponded to it.

The figure 3.20 shows the *lwo*-SOM map where each cell is bordered by a circle or a square in order to show cells which contains the searched word 'flag' ('drapeau' in french).

Figure 3.20: Searching : level 0

The border color indicate the frequency of the searched word in the corresponding cell: red color is representatively to the cell which have more images with the corresponding word; respectively, the blue color represent the less number of images within this word. The square border corresponds to the attached words and the circle form border - to the web link attached to each image. If the first map's level doesn't satisfy the user/expert, the level can be moved to other level and to search the corresponding image or to choose another cell if the image is correlated with the desired ones.

Figures 3.22 and 3.23 shows examples of the wikipedia web pages which correspond to the Los Angeles flag image and to the earth map image respectively, extracted from the obtained map.

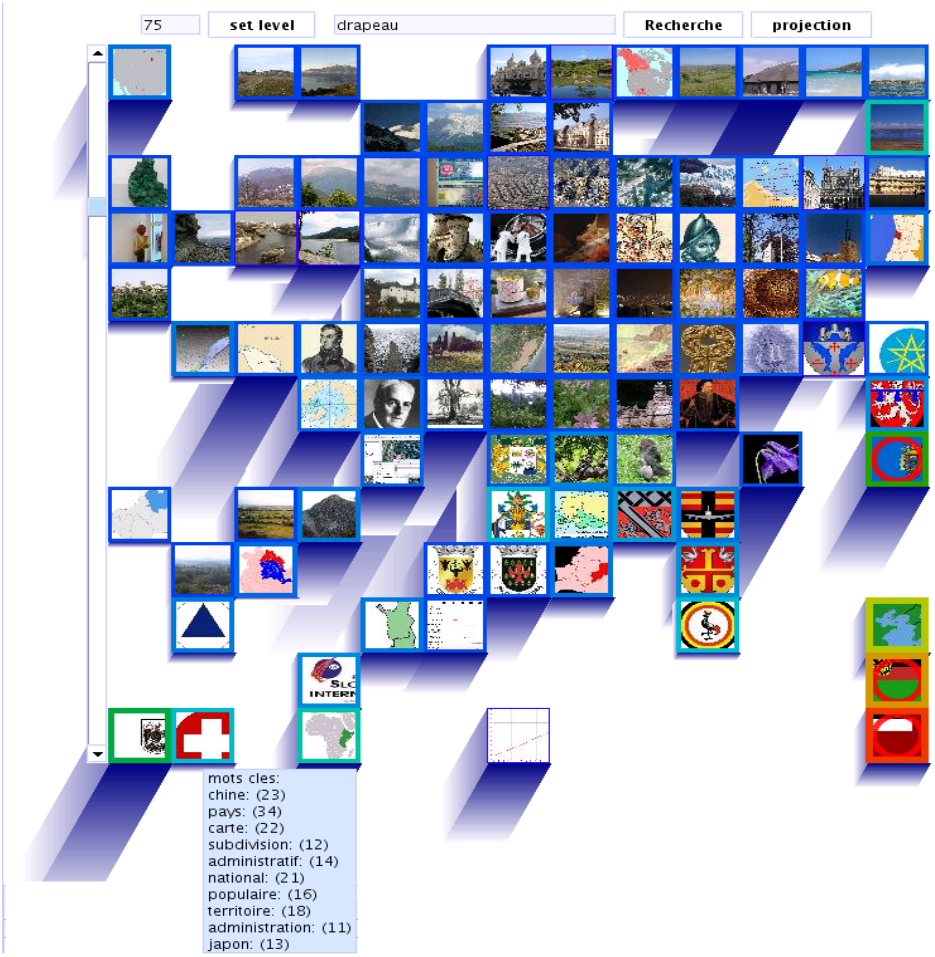


Figure 3.21: Searching: level 75

3.2.8 Interactive Learning

As already stated, sometimes we can have images that don't correspond to the user/expert criteria, and, in this case, the framework gives the possibility to the user to choose the class/-cell where the image should be placed. Our system will compute the minimum Euclidean distance between the image i and the corresponding cell/prototype j:

$$\chi(\mathbf{x}_i) = \arg\min\left(\|\pi_j \mathbf{x}_i - \mathbf{w}_j\|^2\right)$$

The new image is placed in the respective cell and to find it's place in the cluster j the corresponding vector of images is re-sorted.
This process is useful also to cluster the new images which arrives in a real-time to the

Figure 3.22: Web link of a image (1)

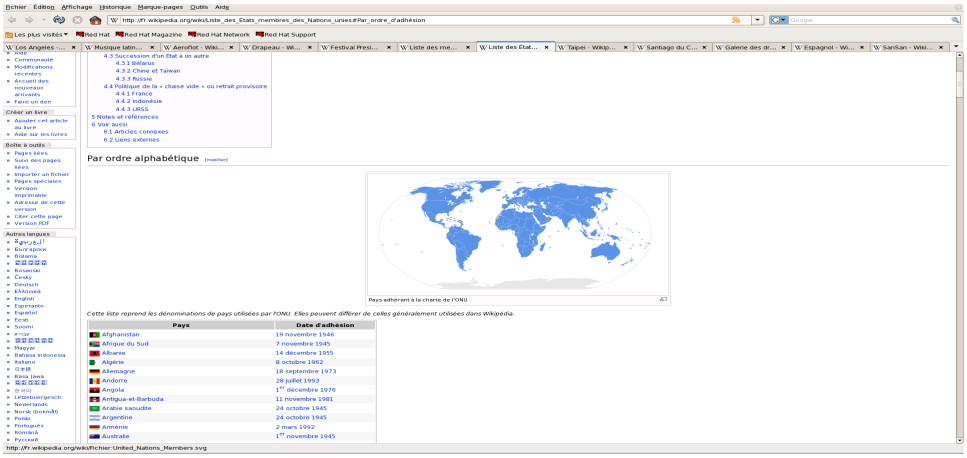


Figure 3.23: Web link of a image (2)

dataset/framework by computing the corresponding distance: firstly between it and all the prototypes/cells, and secondly - to compute the distance inside of the cell between it and all the images to find its place.

3.3 Hybrid topological clustering

3.3.1 Relational Topological Map: Batch algorithm

As the Relational Analysis based clustering techniques doesn't allow clustering visualization, we propose a relational formalism approach for the binary topological maps called RTM.

Using the batch version of the Self-Organizing Maps we will introduce the notion of the topology in the classical relational analysis objective function presented in the Chapter 1. In this case, the affectation phase of the learning algorithm will change, and will take into account the neighborhoods relations in computing the contributions(links):

$$\mathcal{R}^{T}_{cond}(\varphi, W) = \sum_{i=1}^{N} cont^{T}(X_i, W_{\varphi(i)}) \tag{3.4}$$

Suppose that during this phase, the set of prototypes is fixed and constant, so, one have to maximize the objective function $\mathcal{R}^{T}_{cond}(\varphi, W)$ with respect to φ, and we obtain:

$$\forall i;\ \varphi(i) = arg \max_{l}\ cont^{T}(X_i, W_l) \tag{3.5}$$

As for the traditional model of self-organizing map, for the RTM model we use a two layer neural network with a entry layer for the observations and an exit layer representing a topological map C. The proposed RTM model use the same vector quantization where each cell of the map which is the index of the required quantization's prototype will be represented by a vector of the same size of observations.

The quantization is done by means of the assignment function φ adapted to binary data. The choice of prototypes and the assignment function is done by maximizing the following objective function:

$$R^{T}_{cond}(\varphi,\chi) = \sum_{i=1}^{N}\sum_{i'=1}^{N}[c_{ii'} - \alpha(\frac{c_{ii} + c_{i'i'}}{2})]\sum_{r=1}^{L} K^{T}_{(\delta(\varphi(i),r))}\chi_{i'r} \tag{3.6}$$

The maximization of the R^{T}_{cond} objective function must preserve the topology of the map and carry out a partition of a dataset into homogeneous subsets.

By rewriting the objective function in terms of the observations X_i of each object and the prototype W_l of each neuron of the map C, the $R^T_{cond}(\varphi,\chi)$ has the following form:

$$R^T_{cond}(\varphi,\chi) = \sum_{i=1}^{N}\sum_{l=1}^{L} K^T_{(\delta(\varphi(i),l))} \sum_{i'=1}^{N}[c_{ii'} - \alpha(\frac{c_{ii} + c_{i'i'}}{2})]\chi_{i',l} \tag{3.7}$$

We note the contribution between an observation i and its prototype l by $cont(i,l)$ which is defined as following:

$$cont(i,l) = \sum_{i'=1}^{N}[c_{ii'} - \alpha(\frac{c_{ii} + c_{i'i'}}{2})]\chi_{i',l} \tag{3.8}$$

Changing the contribution $cont(i,l)$ which will be noted $cont(X_l, W_l)$ in the expression 3.7, we obtain the following objective function:

$$R^T_{cond}(\varphi, W) = \sum_{i=1}^{N}\sum_{l=1}^{L} K^T_{(\delta(\varphi(i),l))}\left[\langle X_i, W_l\rangle - \alpha\frac{|C_l|\langle X_i, X_i\rangle + \sum_{i'\in C_l}\langle X_{i'}, X_{i'}\rangle}{2}\right] \tag{3.9}$$

$$= \sum_{i=1}^{N} cont^T\left(X_i, W_{\varphi(i)}\right) \tag{3.10}$$

where $cont^T(X_i, W_{\varphi(i)})$ is the regularized contribution of the observation i to his winner neuron $\varphi(i)$ and is defined as following:

$$cont^T(X_i, W_{\varphi(i)}) = \sum_{l=1}^{L} K^T_{(\delta(\varphi(i),l))}\left[\langle X_i, W_l\rangle - \alpha\frac{|C_l|\langle X_i, X_i\rangle + \sum_{i'\in C_l}\langle X_{i'}, X_{i'}\rangle}{2}\right] \tag{3.11}$$

The regularized contribution of the observation i to $\varphi(l)$ (expression 3.11) represent a weighted sum of contributions of the i to all prototype vectors W_l taking into account the neighborhood of influence of the corresponding winner neuron.

In order to obtain the expression to compute the prototypes we will rewrite the contribution between an observation and a prototype in the following simplified form:

$$cont^T(X_i, W_{\varphi(i)}) = \left\langle X_i \sum_{l=1}^{L} K^T_{(\delta(\varphi(i),l))} W_l \right\rangle - \alpha \sum_{l=1}^{L} K^T_{(\delta(\varphi(i),l))} \frac{|C_l| \langle X_i, X_i \rangle + \sum_{i' \in C_l} \langle X_{i'}, X_{i'} \rangle}{2}$$

or

$$cont^T(X_i, W_{\varphi(i)}) = \left\langle X_i, W^T_{\varphi(i)} \right\rangle - \alpha \frac{\sum_{l=1}^{L} K^T_{(\delta(\varphi(i),l))} |C_l| \langle X_i, X_i \rangle + \sum_{l=1}^{L} K^T_{(\delta(\varphi(i),l))} \sum_{i' \in C_l} \langle X_{i'}, X_{i'} \rangle}{2}$$

$$cont^T(X_i, W_l) = \left\langle X_i, W^T_l \right\rangle - \alpha \frac{\sum_{r=1}^{L} K^T_{(\delta(l,r))} |C_r| \langle X_i, X_i \rangle + \sum_{r=1}^{L} K^T_{(\delta(l,r))} \sum_{i' \in C_r} \langle X_{i'}, X_{i'} \rangle}{2}$$

where $W^T_{\varphi(i)}$ is the regularized prototype of the winner neuron $\varphi(l)$ and is defined as following:

$$W^T_{\varphi(i)} = \sum_{l=1}^{L} K^T_{(\delta(\varphi(i),l))} W_l = \sum_{l=1}^{L} K^T_{(\delta(\varphi(i),l))} \sum_{i' \in C_l} X_{i'} \tag{3.12}$$

The maximization step consists in the maximization of the objective function with respect to W by fixing the affectation function φ, and we obtain:

$$cont^T(\varphi, W_l) = \sum_{i=1}^{N} K^T_{(\delta(\varphi(i),l))} cont(X_i, W_l) \tag{3.13}$$

The objective function is rewritten using the following contribution:

$$\mathcal{R}^T_{cond}(\varphi, W) = \sum_{l=1}^{L} cont^T(\varphi, W_l) \tag{3.14}$$

and

$$W_l = arg \max_{W} cont^T(\varphi, W) \tag{3.15}$$

This technique allows to find the profiles $|\mathcal{C}_l|$ from I which have bigger value of the $cont^T(\varphi, W_l)$. More explicitly, this process will update the prototypes for each t iterations based on the following expression:

$$\forall l; \quad W^T_l(t) = \sum_{r=1}^{L} K^T_t(\delta(r,l)) \sum_{i' \in \mathcal{C}^t_r} X_{i'} \tag{3.16}$$

3.3.1.1 RTM heuristics

For the proposed Relational Binary Topological Map (RTM), we consider the batch SOM where the update step of the learning consist to maximize the objective function 3.9 by considering all prototypes Z fixed; the representation step maximize the same function considering the set of clusters to be fixed (the assignment function φ fixed). So, after the initialization step, for a temperature T fixed, the minimization occurs in two alternating steps during the learning process: the assignment and maximization steps.

Algorithme 17 : Heuristic RTM algorithm

Step 1. Initialization: Initialize the greed C using the classical RA approach

for $t = 1$ to N_{iter} **do**

Step 2. Assignment: For an observation $i \in X$ compute its contribution for each neuron of the map C using the expression 3.11.
It is assumed at this stage that all prototypes are fixed and remain constant by maximizing the objective function $R^T_{cond}(\varphi, W)$ compared to φ, and it is easy to see that this maximum is reached for an assignment function defined by:

$$\forall i; \quad \varphi_{(i)}(t) = \arg\max_{l} cont^T(X_i, W_l(t-1)) \tag{3.17}$$

Step 3. Maximization: The maximization step consists in maximizing the objective function over W by fixing the assignment φ. In other words, maximization step consists in updating each regularized prototypes $X^T_l(t)$ of the neuron C_l at each iteration t according to:

$$\forall l; \quad W^T_l(t) = \sum_{r=1}^{L} K^T_{(\delta(r,l))}(t) \sum_{i' \in C_r(t)} X_{i'} \tag{3.18}$$

end for

The proposed algorithm combines the relational analysis for clustering a dataset X and the neighborhood function for the SOM model to construct the grid and to take into account the topological information during the map learning. The algorithm starts by building an empty grid of size C initially fixed. During the first iteration, filling the map will be done incrementally using the classical relational analysis without taking into account the topological information. Then, we apply the topological relational analysis which will allow a self-organization of clusters (map's cells) produced by the RA technique during the first RTM iteration.

In the procedure 17 the heuristic version of the RTM algorithm it is shown.

To computing the contributions between an observation X_i and a prototype W_l, the algorithm is presented in the procedure 18.

Algorithme 18 : PROCEDURE *ComputeContribution*

for $l = 1$ to L **do**

ComputeContribution() $cont^T_{t-1}(X_i, W_l) = :$

$$[< X_i, W^T_l > -\alpha \frac{\sum_{r=1}^{L} K^T_{(\delta(r,l))} |\mathcal{C}_r| < X_i, X_i > + \sum_{r=1}^{L} K^T_{(\delta(r,l))} \sum_{i' \in \mathcal{C}_r} < X_{i'}, X_{i'} >}{2}] \quad (3.19)$$

end for

More detailed, the RTM algorithm is presented in the procedure 19.

Algorithme 19 : RTM batch Algorithm

Inputs:

Initialization : $\mathcal{C}^0$= initial map with L cells. N_{iter}= number of iterations. N= observations number. α= similarity threshold. K^T= neighborhoods matrix

- Randomly choose an observation $i \in I$ to be the first cell element $\mathcal{C}_1$
- $l = 1$, where l is the corresponding cell number
- Initialize the profile of the first constant cell

$$W_l^T(0) = X_i \tag{3.20}$$

Consider $K^T = I_d$, for the iteration $t = 1$, I_d is the identity matrix.

for $t = 1$ to N_{iter} **do**
  for $i = 1$ to N (Affectation) **do**
    ComputeContribution()

$$l^* = arg\max cont_{t-1}^T(X_i, W_l) \tag{3.21}$$

    $cont_{t-1}^T(X_i, W_{l^*}) \leftarrow$ the computed contribution
    if $t \neq 1$ **then**
      Affect i to $\mathcal{C}_{l^*}$
    else
      if $cont_{t-1}^T(W_i, W_{l^*}) < 0$ and $l < L$ **then**
        Build a new cell where i is the first affected observation for this cell
        $l = l + 1$
        Compute $W_{\mathcal{C}_{l+1}}$: $W_{\mathcal{C}_{l+1}}^T(t) = \sum_{i' \in \mathcal{C}_{l+1}^t} X_{i'}$
        Randomly place the cell $\mathcal{C}_{l+1}$ on the map $\mathcal{C}$
      else
        Affect i to $\mathcal{C}_{l^*}$
      end if
    end if
  end for
  for $l = 1$ to L (Update of the profiles) **do**

$$W_l^T(t) = \sum_{r=1}^{L} K_{t,(\delta(r,l))}^T \sum_{i' \in \mathcal{C}_r^t} X_{i'} \tag{3.22}$$

  end for
end for

OUTPUT:

A map with L cells.

3.3.2 Experimentations and Validations

We use the zoo dataset to show the good performane of the RTM algorithm. This dataset contains 101 animals described with 16 qualitative variables: 15 of the variables are binary and one is numeric with 6 possible values. Each animal is labelled 1 to 7 according to its class. Using disjunctive coding for all variables with 6 possible values for the numerical variable, the data set consists of a 101x36 binary data matrix. All 101 animals are used for learning a map with the dimension 5x5 cells. At the end of the learning step, each observation, corresponding to an animal, is assigned to the cell with the highest contribution by taking into account the neighborhood relation.

	(1)	(7)		
(1)			(4)	
	(7)	(3)		
(1)	(3)		(6)	(2)
	(5)	(6)	(1)	

Figure 3.24: Initialization map using Relational Analysis algorithm

The RTM algorithm start with the initialization of the grid by distributing the observations using the classical relational analysis approach. The figure 3.24 shows the class of animals distributed after the initialization step of the RTM algorithm. We used the animals names used in original dataset. To visualize the coherence of the map with the labelling of animals, this figure shows the class number corresponding to each cell after the application of the majority rule in each cell. We remind that during this learning step, the neighborhood information is not considered (the neighborhood function K is not computed). On the initialization grid (figure 3.24) the observations are not well distributed, there is two set of observations labeled with 7 which are separated by 2 empty cells; we can find also four sets of animals labeled as 1 which are dispersed on the map: two sets on the left top corner, one

set is situated on the left bottom corner, and the last one - on the right bottom part of the map. This map demonstrate that the classical RA doesn't use a topological information during the clustering process which could allow a good distribution of the observations.

After the initialization step, the RTM algorithm will continue the learning process by taking into account the neighborhood relation between all the cells. Figure 3.25 shows animals names collected by each cell. The map shows that the same class of animals is assigned to cells close to each other.

Clam Crab Crayfish lobster octopus seawasp slug starfish worm (7)	Hamster (1) Ladybird (6) Scorpion (7) (1.5)	chicken dove flamingo parakeet sparrow vulture (2)	dolphin leopard pony (1)	bass catfish chub dogfish haddock herring pike piranha stingray tuna (4)
flea honeybee moth wasp (6)	Toad (5) Tortoise (3) (3.5)	antelope hare vole (1)	gnat housefly termite (6)	mink oryx pussycat seal (1)
aardvark gorilla lion reindeer (1)	giraffe puma wallaby (1)	frog newt pitviper (5)	Frog (5) Penguin (2) (2.5)	crow hawk ostrich pheasant (2)
seasnake squirrel vampire (1)	porpoise sealion (1)	slowworm tuatara deer (3)	carp seahorse sole (4)	duck gull kiwi skua (2)
girl opossum platypus raccoon wolf (1)	boar calf elephant goat lynx (1)	bear cavy cheetah (1)	buffalo fruitbat mole mongoose polecat (1)	lark rhea skimmer swan wren (2)

Figure 3.25: Relational Topological Map : zoo dataset

We can observe that the animals corresponding to the class 1 are clustered in the cells situated on the left bottom part of the map (figure 3.25); the birds which correspond to the

class 2 are in the right bottom part of the map. Also, we can analyze that fruitbat from the class 1 situated nearest to the cell containing the birds (class 2) is explained that the fruitbat has nearest characterization with the birds even it comes from the insects family.

On the middle of the map there is a cell containing 2 observations from two different classes : the frog (class 5) and penguin (class 2). The RTM algorithm put these two observations in the same cell because the frog and the penguin has very closest specifications even the penguin belongs to birds family and frog - from the amfibia family. Moreover, on the left of this cell there is a cell containing the animals from class 5, and on the right - a cell labeled as class 2. We have the same situation for the cell labeled as 3.5 where the toad and the tortoise has highly correlated features, and the both cells labeled as 5 and 1 are bordered on the right from this cell. The same type of analysis can be applied to the remaining clusters. To give a global view of homogeneous clustering, we compute the classification rate for all animal dataset and we obtain a classification rate of 97.84%.

We compare our map with the map obtained using the BeSOM (Bernoulli on Self-Organizing Map) [85] which use a probabilistic reformulation of the classical SOM. The map obtained using the BeSOM method is presented in the figure 3.26. Analyzing both maps obtained with BeSOM (figure 3.26) and with the proposed RTM approach (figure 3.25) we can detect some correlations between them: class 5 and 2 are situated in the middle of the map, the majority of the cells containing animals forming the first class are situated on the left bottom corner of the map. Comparing with the BeSOM zoo map, we can observe that RTM zoo map provides more finer cells: in the case of the BeSOM map there are three cells which contains only one observation which respectively will attribute to these ones a 100% of purity, and 8 cells containing by two observations which is also easy to have a high purity score. The RTM map has no cell which contains only one observation an has only 3 cells with two observations that means that our map has cells with a better distribution of observations.

In order to show the good performance of the Relational Binary Topological Map (RTM) approach we use several binary datasets of different sizes. For each dataset we learned a map of different sizes (from 4x4 to 10x10) and we indicate in the table 3.2 the purity of clustering after the first iteration using the classical relational analysis and the map purity at the end of the map learning with the RTM technique. The results illustrate that the proposed technique increase the purity index compared to the classical RA and allow to obtain a topological map by computing the neighborhood function between the cells.

The datasets used for the experimentations are described in the Appendix of this work.

antelope buffalo deer elephant giraffe hare mole opossum oryx vole (1)	dolphin porpoise (1)	bass catfish chub dogfish herring pike piranha stingray tuna (4)	carp haddock seahorse sole (4)	clam seawasp (7)
aardvark bear boar cheetah leopard lion lynx mink mongoose polecat puma pussycat raccoon wolf (1)	frog frog newt toad tuatara (5)	pitviper slowworm (3)	slug worm (7)	crab crayfish lobster octopus starfish (7)
calf cavy goat hamster pony reindeer (1)	scorpion (7)	kiwi ostrich penguin rhea vulture (2)	girl seal sealion (1)	platypus seasnake tortoise (3)
fruitbat squirrel vampire (1)	duck flamingo swan (2)	chicken dove lark parakeet pheasant sparrow wren (2)	flea termite (6)	gnat (6)
gorilla wallaby (1)	crow gull hawk skimmer skua (2)	honeybee wasp (6)	housefly moth (6)	Ladybird (6)

Figure 3.26: 5x5 BeSOM map taking from [85]

3.4 Conclusions

We introduced in this chapter the problem of the modular and hybrid clustering. For the modular based clustering, we presented an original and new approach of fusion/ensemble/-consensus/aggregation clustering. The main idea was to find a clustering (or partition) of observations that represents the best consensus between several other clustering related to the same data set. The goal of the proposed algorithm is the improvement of confidence in cluster assignments by evaluating a history of cluster assignments for each observation. If we compare our algorithm (or method) to some recent clustering algorithm, we can assert that, unlike these new algorithms, our method is scalable, linear, has a small computational time and can handle data represented as observations cross attributes or as similarity matrix. Our clustering method handles missing values without replacing them by values that could be very far away from the true ones. It also contains a preprocessing module that, among other

Table 3.2: Experimentation results on different datasets using RTM approach

Dataset	Dataset size	Map size	AR purity	RTM purity
Zoo	101x21	5x5	69.08	97.84
Car	1728x21	10x10	70.31	80.17
Nursey	12960x29	6x6	50.47	78.69
SPECTF	80x1127	4x4	57.14	81.82
Pos-operative	90x24	5x5	71.59	78.21

processing, can compute how discriminant are the attributes measured on the observations to be clustered. Finally we verified the intuitive appeal of the proposed approach and we studied the behavior of our algorithm on real and synthetic heterogeneous data sets. We observed that the proposed method increases performance as more as iterations of the process are performed. Another advantage of our method is that, neither do we need to re-process the data; nor do we need to fix the same cluster numbers for each application or clustering algorithm. For the application of the proposed fusion schema we presented a novel solution for manage and process visual datasets. This process has been patented by us and Thales S.A. [15]. We proposed two scenarios: a clustering and a browsing schema which could be done simultaneously with the interactive learning. Also, we propose an original solution for the searching on the images libraries using the annotated text only to find the corresponding level of the map, and then to use the correlation between images on the map and inside the cell to display the information, in order to avoid the eventual noise (worst annotated text). The perspectives of the proposed fusion schema is to perform a more detailed analysis involving huger data set, and to attempts this model to characterize clusters after the fusion, by using some weighting/memory techniques during the learning process from the beginning (local learning) until the fusion.

In order to attempts a hybrid based clustering, we proposed a new model for clustering and visualization of binary data based on the neighborhood relation from the classical SOM but using the formalism of the relational analysis approach. The proposed hybrid RTM approach combines the advantages of both methods, indeed it allows a natural clustering identification without fixing the number of clusters (or cells). However, this model addresses in the same manner variables describing the data by ignoring their internal structures, the number of modalities per variable and the size of each modality. One of the perspective of the RTM approach is to ameliorate the learning process by weighting the Condorcet criteria used in this model.

Chapter 4

Collaborative Clustering

4.1 Introduction

In an industrial context of increasing competition, companies are constantly called upon to work together (to collaborate) to face up to this strategic issue and to be able to provide the level of required service for their customers. To benefit from this collaboration, the tasks of Data Mining (Clustering, mining and knowledge management ...) should consider all the datasets associated with these collaborating companies although they are distributed on several different sites.

Obviously, for confidentiality reasons (ex. medical or bank data), sharing data between collaborating companies is not allowed. So, centralizing their data by combining them into one dataset and then performing the task of Data Mining is not appropriate.
In this work, we are interested in the problem of clustering and specifically in collaborative clustering preserving data confidentiality and using self-organizing maps of Kohonen.

The rest of this chapter is organized as follows: we'll present our vertical and horizontal collaboration approaches in section 4.3 and 4.4, after introducing the problem of collaborative clustering. In Section 4.5, we'll present different results. Finally, the chapter is completed by the conclusion and some perspectives of the proposed methods.

4.2 Collaborative Clustering

Collaboration between the companies has intensified in recent years and has become one of the usual corporate strategies. Often, large local companies outsource the manufacture of certain parts or the provision of certain services to small businesses. Moreover, there are small companies in the same activity sector or even in the same industry forming between them strategic organizations which favors flexible specialization and collective efficiency.

In both cases, to better take advantage of collaboration and clustering, the collaborating companies must have a global clustering result considering all their data. Obviously, for reasons of confidentiality, data sharing between these companies is forbidden, which prevents centralizing their data by combining them into one database and perform clustering on the latter. Thus, in literature. some studies have been proposed which allow collaboration of several distributed datasets over several different sites while preserving the confidentiality of these data. According to the structure of collaborated datasets, there are three main types of collaboration: horizontal, vertical and hybrid collaboration [106]. In this work, we are particularly interested in horizontal and vertical topological collaborative clustering.

In the chapter 3, we introduced our approach in the dimensionality reduction, but imagine the case when we have a very large data set and all the features a pertinants and no reduction can be made. In this case, to accelerate the data mining of these data sets we can separate the data in some subsets (using a separate collaborative function) and then we can use the collaborative clustering to cluster this data.

Collaborative clustering was first investigated by Pedrycz [106, 107, 108, 94, 109], using a fuzzy k-means algorithm. The fundamental concept of collaboration is : “the clustering algorithms operate locally (namely, on individual data sets) but collaborate by exchanging information about their findings” Pedrycz.

4.3 Topological unsupervised horizontal collaborative clustering

In the case of horizontal clustering, all datasets describe the same observations. So, all these collaborative datasets have the same number of observations but a different number of variables. What we would like is that after the collaboration, if an observation of the i-th dataset

is projected onto the j-th neuron of the ii-th SOM map, then the same observation of the jj-th dataset is projected on the same neuron j in the jj-th map or on one of the neighboring neurons. In other words, neurons that correspond to different maps should capture the same observations. That's why we have added a term to the objective function of the classical SOM learning algorithm in order to approximate the neurons to which an individual belongs on all the maps. We weighted this function by the collaboration parameter α which is set by the expert depending on the confidence of the collaborated map.

Formally, we have achieved the following new objective function:

$$R_{SOM}^{HCol}[ii] = \sum_{i=1}^{N}\sum_{j=1}^{|w|} K_{j,\chi(x_i)}[ii]\left\|x_i[ii] - w_j[ii]\right\|^2 + \tag{4.1}$$

$$+ \sum_{jj=1, jj\neq ii}^{P} \alpha[ii, jj] \sum_{i=1}^{N}\sum_{j=1}^{|w|} \left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2 \left\|x_i[ii] - w_j[ii]\right\|^2$$

Where P is the number of datasets, N – the number of observations on the dataset and ii is the same for all bases, $|w|$ is the number of prototype vectors of the map ii. This function is minimized for each dataset during the collaboration step.

Figure 4.1 shows the schema of horizontal collaboration between multiple maps from several datasets.

The minimization of the horizontal collaborative clustering

In the case of the horizontal collaborative clustering, the function to optimize is the following:

$$R_{SOM}^{HCol}[ii] = \sum_{i=1}^{N}\sum_{j=1}^{|w|} K_{j,\chi(x_i)}[ii]\left\|x_i[ii] - w_j[ii]\right\|^2 + \sum_{jj=1, jj\neq ii}^{P} \alpha[ii, jj] \sum_{i=1}^{N}\sum_{j=1}^{|w|} \left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2$$

$$\left\|x_i[ii] - w_j[ii]\right\|^2$$

So, we have $x_i = (x_{i1}, x_{i2}, .., x_{in})$ et $w_j = (w_{j1}, w_{j2}, .., w_{jn})$ and the $R_{SOM}^{HCol}[ii]$ function can be re-written as following:

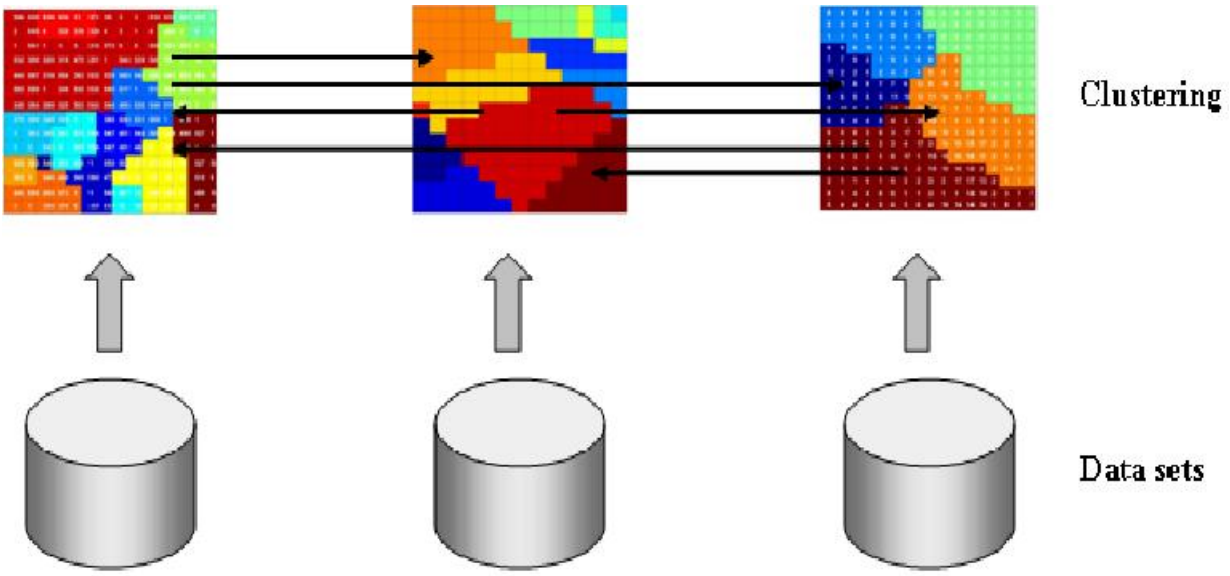


Figure 4.1: Horizontal collaborative SOM

$$R_{SOM}^{HCol}[ii] = \sum_{i=1}^{N}\sum_{j=1}^{|w|} K_{j,\chi(x_i)}[ii] \sum_{k=1}^{n}[ii] \left\|x_{ik}[ii] - w_{jk}[ii]\right\|^2 + \sum_{jj=1,jj\neq ii}^{P} \alpha[ii,jj] \sum_{i=1}^{N}\sum_{j=1}^{|w|} \left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2$$

$$\sum_{k=1}^{n[ii]} \left\|x_{ik}[ii] - w_{jk}[ii]\right\|^2$$

The necessary condition for finding the minimum of the $R_{SOM}^{HCol}[ii]$ is $\nabla w[ii] R_{SOM}^{HCol}[ii] = 0$

$$\frac{\partial R_{SOM}^{HCol}[ii]}{\partial w_{jk}[ii]} = 0 \Rightarrow -2\sum_{i=1}^{N} K_{j,\chi(x_i)}\left(x_{ik} - w_{jk}[ii]\right) - 2\sum_{jj=1,jj\neq ii}^{P}\sum_{i=1,N} \alpha[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2$$

$$\left(x_{ik}[ii] - w_{jk}[ii]\right) = 0 \ldots (1)$$

$$(1) \Rightarrow \sum_{i=1}^{N} K_{j,\chi(x_i)}\left(x_{ik} - w_{jk}[ii]\right) + \sum_{jj=1,jj\neq ii}^{P}\sum_{i=1}^{N} \alpha[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2$$

$$\left(x_{ik}[ii] - w_{jk}[ii]\right) = 0$$

$$(1) \Rightarrow \sum_{i=1}^{N} K_{s,\chi(x_i)}[ii] - \sum_{i=1,N} K_{s,\chi(x_i)} w_{sk}[ii] + \sum_{jj=1,jj\neq ii}^{P} \sum_{i=1}^{N} \alpha[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2 x_{ik}[ii]$$

$$- \sum_{i=1}^{N} \alpha[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2 w_{jk}[ii] = 0$$

$$(1) \Rightarrow w_{jk}[ii]\left(\sum_{i=1}^{N} K_{j,\chi(x_i)} + \sum_{jj=1,jj\neq ii}^{P} \sum_{i=1}^{N} \alpha[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2\right) =$$

$$\sum_{i+1}^{n} K_{j,\chi(x_i)} x_{ik}[ii] + \sum_{jj=1,jj\neq ii}^{P} \sum_{i=1}^{N} \alpha[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2 x_{ik}[ii]$$

$$(1) \Rightarrow w_{jk}[ii] = \frac{\sum_{i+1}^{n} K_{j,\chi(x_i)} x_{ik}[ii] + \sum_{jj=1,jj\neq ii}^{P} \sum_{i=1}^{N} \alpha[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2 x_{ik}[ii]}{\sum_{i=1}^{N} K_{j,\chi(x_i)} + \sum_{jj=1,jj\neq ii}^{P} \sum_{i=1}^{N} \alpha[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2}$$

The topological horizontal collaboration clustering algorithm is summarized in the procedure 20. In this procedure the two steps: local and collaborative are indicated. We note, that in case of collaborating of existed maps (clustering results) the algorithm escape the local phase because the map existed already and will start directly the collaboration step.

Algorithme 20 : Topological horizontal collaboration Algorithm

Initialization:

Initialize all the map prototypes W randomly. Set the matrix of collaboration $\alpha[ii, jj]$

1. Local step:

for each DB, $X[ii]$, $ii = 1$ à P **do**

Minimize the objective function of the classical SOM:

$$R_{SOM}(\chi, W) = \sum_{i=1}^{N} \sum_{j=1}^{|w|} K_{j,\chi(x_i)}[ii] \|x_i[ii] - w_j[ii]\|^2 \tag{4.2}$$

for $t = 1$ to T_{max} (T_{max} indicates the number of iterations) **do**

1.A. Learning step:

At each iteration t choose an input $x(t)$ in general, randomly, and present it to the map.

1.A.1 Competition step:

Choose the best matching unit (BMU) $i*$ by computing $\|x_i - w_j(t)\|^2$

1.A.2 Updating step:

Update the winner neuron $i*$ and its neighbors:

$w_j(t+1) = w_j + \varepsilon(t) K_{j,\chi(x_i)}(x_i - w_j(t))$

Decreasing the size of the bmus neighborhood area and the learning coefficient $\varepsilon(t) = 0$.

end for

end for

2.Collaborative step:

for each DB $X[ii]$, $ii = 1$ to P **do**

Minimize the horizontal collaborative learning objective function 4.1.

Compute prototypes of the new ii-th map:

for $s = 1$ to $|w|$ **do**

for $t = 1$ to $|x|$ **do**

$$w_{jk}[ii] = \frac{\sum_{i+1}^{n} K_{j,\chi(x_i)} x_{ik}[ii] + \sum_{jj=1, jj\neq ii}^{P} \sum_{i=1}^{N} \alpha[ii, jj] \left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2 x_{ik}[ii]}{\sum_{i=1}^{N} K_{j,\chi(x_i)} + \sum_{jj=1, jj\neq ii}^{P} \sum_{i=1}^{N} \alpha[ii, jj] \left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2} \tag{4.3}$$

end for

end for

end for

4.4 Topological unsupervised vertical collaborative clustering

Unlike horizontal clustering, in the case of vertical clustering, all datasets have the same variables.

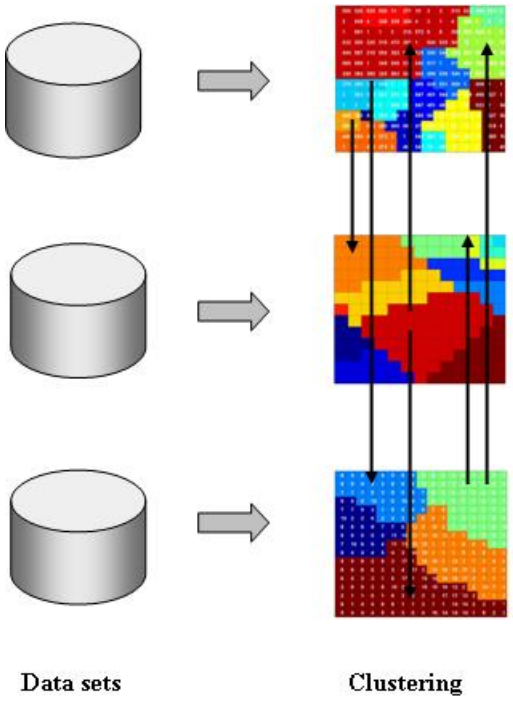


Figure 4.2: A general schema of vertical clustering

Therefore, all observations of these datasets have the same size and the size of prototype vectors of all SOM maps is the same. We would like a neuron j in the i-th SOM map be similar to the same neuron j of the jj-th map in terms of Euclidean distance. In other words, neurons that correspond to different maps represent similar classes. This is why we have changed the objective function of the classical SOM algorithm to approximate the similarity of vectors corresponding to different maps. We weighted this function by the collaborative β parameter. Formally, we have achieved the following new objective function:

$$R_{SOM}^{VCol} = \sum_{i=1}^{N[ii]} \sum_{j=1}^{|w|} K_{j,\chi(x_i)}[ii] \sum_{k=1}^{n} (x_{ik}[ii] - w_{jk}[ii])^2 + \tag{4.4}$$

$$+ \sum_{jj=1, jj \neq ii}^{P} \beta[ii, jj] \sum_{i=1}^{N[ii]} \sum_{j=1}^{|w|} \left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj] \right)^2 \sum_{k=1}^{n} \left\| w_{jk}[ii] - w_{jk}[jj] \right\|^2$$

Where P is the number of datasets, $N[ii]$ - the number of observations of the ii-th dataset, $|w|$ is the number of prototype vectors of the map ii which is the same for all the maps. This function is minimized for each dataset ii during the collaboration step. Figure 4.2 shows the general schema of vertical collaboration between several SOM maps.

Minimization of the vertical collaborative objective function

For the vertical collaborative objective function, the objective function to optimize is the following:

$$R_{SOM}^{VCol} = \sum_{i=1}^{N[ii]} \sum_{j=1}^{|w|} K_{j,\chi(x_i)}[ii] \left\| x_i[ii] - w_j[ii] \right\|^2 + \sum_{jj=1, jj \neq ii}^{P} \beta[ii, jj] \sum_{i=1}^{N[ii]} \sum_{j=1}^{|W|} \left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj] \right)^2$$

$$\left\| w_j[ii] - w_j[jj] \right\|^2$$

So, we have $x_i = (x_{i1}, x_{i2}, .., x_{in})$ et $w_j = (w_{j1}, w_{j2}, .., w_{jn})$, and, we can re-write the R_{SOM}^{VCol} function as following:

$$R_{SOM}^{VCol} = \sum_{i=1}^{N[ii]} \sum_{j=1}^{|w|} K_{j,\chi(x_i)}[ii] \sum_{k=1}^{n} (x_{ik}[ii] - w_{jk}[ii])^2 + \sum_{jj=1, jj \neq ii}^{P} \beta[ii, jj] \sum_{i=1}^{N[ii]} \sum_{j=1}^{|w|} \left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj] \right)^2$$

$$\sum_{k=1}^{n} \left\| w_{jk}[ii] - w_{jk}[jj] \right\|^2$$

The necessary condition for finding the minimum of R_{SOM}^{VCol} is $\nabla w[ii] R_{SOM}^{VCol} = 0$

$$\frac{\partial R_{SOM}^{VCol}}{\partial w_{jk}[ii]} = 0 \Rightarrow -2 \sum_{i=1}^{N[ii]} K_{j,\chi(x_i)}[ii](x_{ik}[ii] - w_{jk}[ii]) + 2 \sum_{jj=1, jj \neq ii}^{P} \sum_{i=1}^{N[ii]} \beta[ii, jj] \left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj] \right)^2$$

$$(w_{jt}[ii] - w_{jt}[jj]) = 0 \ldots (1)$$

$$(1) \Rightarrow -2\left(\sum_{i=1}^{N[ii]} K_{j,\chi(x_i)}[ii] - \sum_{i=1}^{N[ii]} K_{j,\chi(x_i)}[ii] w_{jk}[ii]\right)$$

$$+2\left(\sum_{jj=1,jj\neq ii}^{P}\sum_{i=1}^{N[ii]}\beta[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2 w_{jk}[jj]\right) = 0$$

$$(1) \Rightarrow 2w_{jk}[ii]\left(\sum_{i=1}^{N[ii]} K_{j,\chi(x_i)}[ii] + \sum_{jj=1,jj\neq ii}^{P}\sum_{i=1}^{N[ii]}\beta[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2\right) =$$

$$-2\left(\sum_{i=1,N[ii]} K_{j,\chi(x_i)}[ii] x_{ik}[ii] + \sum_{jj=1,jj\neq ii}^{P}\sum_{i=1}^{N[ii]}\beta[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2 w_{jk}[jj]\right)$$

$$(1) \Rightarrow w_{jk}[ii] = \frac{\sum_{i=1}^{N[ii]} K_{j,\chi(x_i)}[ii] x_{ik}[ii] + \sum_{jj=1,jj\neq ii}^{P}\sum_{i=1}^{N[ii]}\beta[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2 w_{jt}[jj]}{\sum_{i=1}^{N[ii]} K_{j,\chi(x_i)}[ii] + \sum_{jj=1,jj\neq ii}^{P}\sum_{i=1}^{N[ii]}\beta[ii,jj]\left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2}$$

The vertical collaborative clustering approach based on self-organizing maps is presented in the procedure 21 containing two steps: the local step and the collaboration step. As for the horizontal collaboration, the vertical collaborative based clustering algorithm will start directly with the collaboration phase if the maps are already known (protypes matrix is computed).

Algorithme 21 : Topological vertical collaboration Algorithm

Initialization:

Initialize all the map prototypes W randomly. Set the matrix of collaboration $\beta[ii, jj]$

1. Local step:

for each DB, $X[ii]$, $ii = 1$ à P **do**

Minimize the objective function of the classical SOM:

$$R_{SOM}(\chi, W) = \sum_{i=1}^{N} \sum_{j=1}^{|w|} K_{j,\chi(x_i)}[ii] \|x_i[ii] - w_j[ii]\|^2 \tag{4.5}$$

for $t = 1$ to T_{max} (T_{max} indicates the number of iterations) **do**

1.A. Learning step:

At each iteration t choose an input $x(t)$ in general, randomly, and present it to the map.

1.A.1 Competition step:

Choose the best matching unit (BMU) $i*$ by computing $\|x_i - w_j(t)\|^2$

1.A.2 Updating step:

Update the winner neuron $i*$ and its neighbors:

$w_j(t+1) = w_j + \varepsilon(t) K_{j,\chi(x_i)}(x_i - w_j(t))$

Decreasing the size of the bmus neighborhood area and the learning coefficient $\varepsilon(t) = 0$.

end for

end for

2.Collaborative step:

for each DB $X[ii]$, $ii = 1$ to P **do**

Minimize the horizontal collaborative learning objective function 4.4.

Compute prototypes of the new ii-th map:

for $s = 1$ to $|w|$ **do**

for $t = 1$ to $|x|$ **do**

$$w_{jk}[ii] = \frac{\sum_{i=1}^{N[ii]} K_{j,\chi(x_i)}[ii] x_{ik}[ii] + \sum_{jj=1, jj\neq ii}^{P} \sum_{i=1}^{N[ii]} \beta[ii, jj] \left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2 w_{jt}[jj]}{\sum_{i=1}^{N[ii]} K_{j,\chi(x_i)}[ii] + \sum_{jj=1, jj\neq ii}^{P} \sum_{i=1}^{N[ii]} \beta[ii, jj] \left(K_{j,\chi(x_i)}[ii] - K_{j,\chi(x_i)}[jj]\right)^2} \tag{4.6}$$

end for

end for

end for

4.5 Experimental results

To validate our approach, we computed the quantization error (distortion) on several maps of different size. To show the importance of our approach to the clustering problem, we calculated the purity index of each map.

The purity of a neuron is the percentage of data belonging to the majority class. Assuming knowledge of all classes of data $L = (l_1, l_2, ..., l|L|)$ and all the neurons $C = (c_1, c_2, ..., c_{|C|})$, the term which expresses the purity of a map is defined as follows:

$$purity = \sum_{k=1}^{|C|} \frac{c_k}{n} \frac{max_{i=1}^{|L|} |c_k^i|}{c_k} \tag{4.7}$$

Where $|c_k|$ represents the total number of data associated with the neuron c_k, $|c_{ik}|$ represents the number of data class l_i which are associated with neuron c_k and n the total number of data. The purity of the map is equal to the average purity of neurons. A good SOM map should have a greater purity.

4.5.1 Results on the waveform dataset

We use the waveform dataset (Appendix A.1) in order to show the improvement of the collaborative topological clustering approaches because this dataset has twenty noise features added to the data, and allows us to provide a visualization of each results. We note that the analysis of these results must be made in a color mode. The computed validation indices are shown in tables 4.1 and 4.2, and are discussed in the chapter only those which changes significantly.

4.5.1.1 Improvement of the horizontal approach

In order to have four datasets with the same observations but described by different variables, we divided the Waveform dataset size 5000×40 into four databases: the first and the second part of the dataset (5000×10; 5000×10) which corresponds to all the relevant variables - variables $[1, 2, ..., 20]$; the second and third part (5000×10; 5000×10) containing the noise of the base - the set of variables $[21, 22, ..., 40]$. We use these datasets to show the entire process that will enable collaboration of all datasets in a horizontal manner.

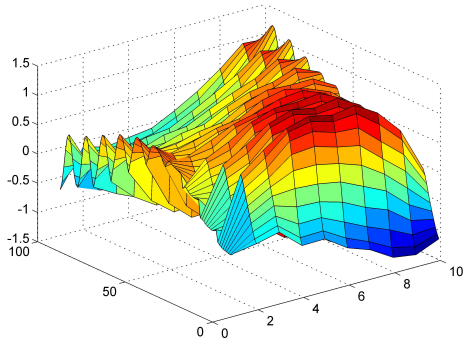
The collaboration matrix α was set initially at $\alpha = [11, 11]$. We selected maps of dimension 10x10, and then we execute the local step of the collaboration clustering algorithm on the first dataset which is to learn a SOM map for all the observations of this dataset. Another map has been used to learn the observations of the second dataset during the local learning step. Figure 4.3(a) and Figure 4.3(b) present respectively the prototype vectors obtained on the first and second map after the local learning step. The two axes X and Y represent respectively the indices of variables and prototypes. We, then construct another map for the 1st dataset by collaborating with the second map (SOM2) and obtained a map presented in the figure 4.3(c). The same collaboration was done for the 2nd dataset in collaboration with the first map (SOM1) and we obtained the SOM21 map which is shown in the figure 4.3(d).

If we compare the first local map (SOM1) which has a purity index equal to 75.71% and the one obtained after the collaboration (SOM12), we can notice that the second local map has influenced the latter. It is thus that, due to the influence of the high important variables from the second local map (SOM2) having 79.61% purity index, the variables $[1-4]$ of the collaborated SOM12 map has more importance (red color) compared to the SOM1 map obtained from the same dataset. The collaborated SOM12 map purity increase to 76.21% compared to the first local map thanks to the SOM2 map with which the map collaborated. And similarly, the first four variables from the collaborated SOM21 map has less importance compared to the local SOM2 map due to the collaboration with the first map (SOM1) where the variables $[1-4]$ has less importance compared to the SOM2 local map, and respectively the accuracy index decrease to 78.72% as a result of the collaboration with a map which has a smaller accuracy. The same analysis can be done for the learning quantization errors of these maps. The respectives purity indexes and quantization errors can be find in the table 4.1 from this chapter.

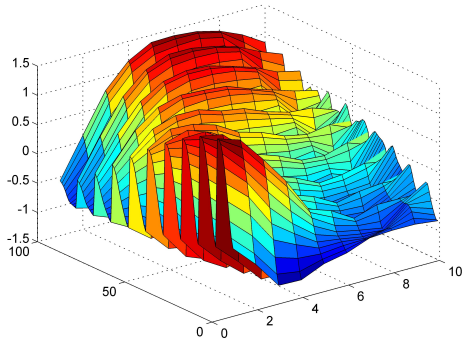
Table 1 summarizes the evaluation criteria associated with these maps.

The local step of the collaboration approach was done also for the third and fourth datasets which contains noise variables. The figure 4.4 shows the both local maps SOM3 and SOM4 issued from these dataset which shows the noise features for all the prototypes. Due to the features noise in these datasets, the purity indices are very small compared to the first two maps, and are equals to 47.19% for SOM3 map and to 51.26% for SOM4 map (table 4.1).

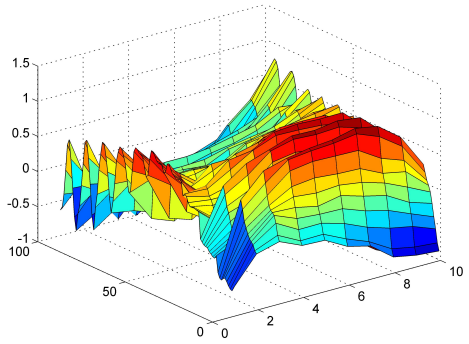
Now, we will collaborate the local SOM1 map (figure 4.3(a)) with local the SOM3 map (figure 4.4(a)) in order to see the influence of the noise features on the important features and viceversa. On the SOM13 map (figure 4.5(a)) obtained after the collaboration with the SOM3 map, we can see that the high important variables $[6-10]$ reduced the importance due to the collaboration with the noise variables compared to the local SOM1 map and the purity index decrease considerably to 62.47% (from 75.71%). Contrarily, the collaborated SOM31



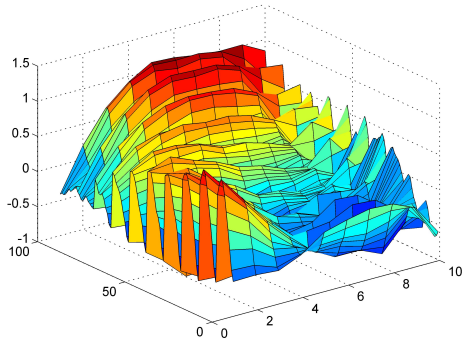
(a) The map of the 1st dataset before collaboration : SOM1



(b) The map of the 2nd dataset before collaboration : SOM2



(c) The map of the 1st dataset after collaboration with SOM2 : SOM12

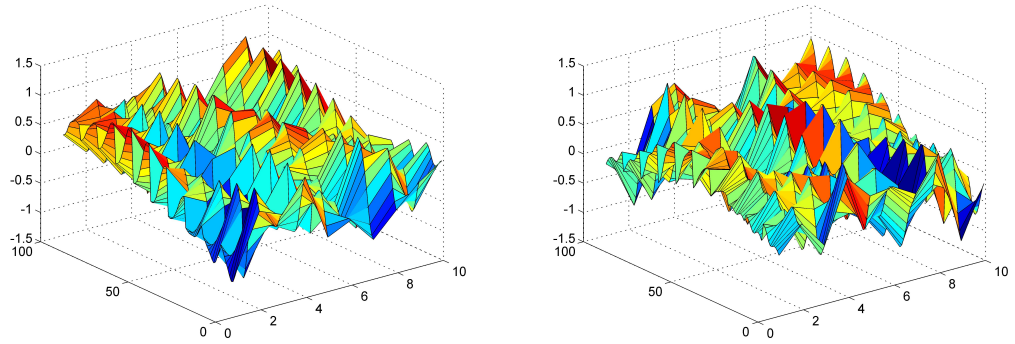


(d) The map of the 2nd dataset after collaboration with SOM1 : SOM21

Figure 4.3: Horizontal collaboration for datasets 1 and 2

map (figure 4.5(b)) has increase only a little the importance of the features $[1-4]$ compared to the local SOM3 map (the accuracy index increase to 54.63% from 47.19%). Therefore, these results confirm that the noise features has a big influence on other features and should be eliminated before the learning process by using a features weighting/selection algorithm as that presented in the Chapter 2.

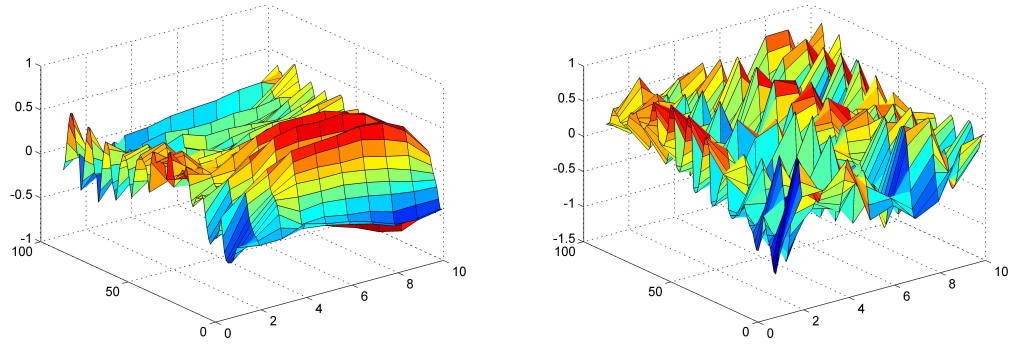
We will made the same collaboration process between the SOM1 local map containing relevant variables and the local SOM4 map containing noise variables. As for the SOM13 map, the noise SOM4 map influenced the SOM1 map but a little less than collaborating with the SOM3 map. The relevant features decreased their importance after the collaboration due to the noise features presented in the local SOM4 map (figure 4.6 (a)). Relevant variables can't increase the importance of features from the 4th map, to do this we need to increase the confidence parameter α for the 1st map. The purity indices and quantization errors which



(a) The map of the 3th dataset before collaboration : SOM3

(b) The map of the 4th dataset before collaboration : SOM4

Figure 4.4: SOM maps for the dataset 3 and 4 before collaboration



(a) The map of the 1st dataset after collaboration with SOM3 : SOM13

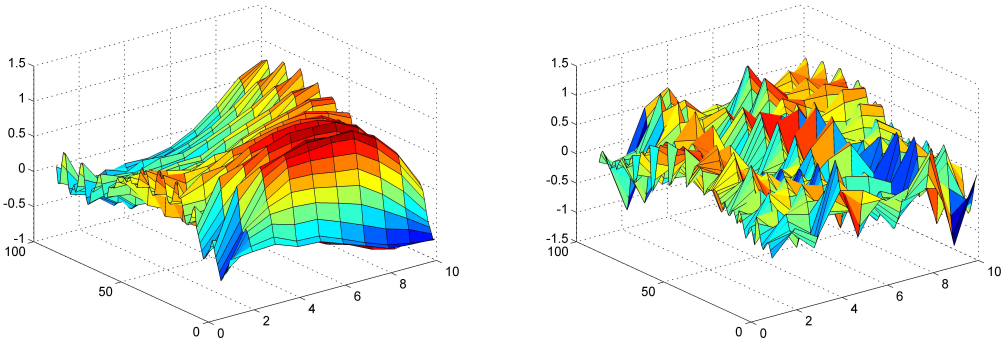
(b) The map of the 3th dataset after collaboration with SOM1 : SOM31

Figure 4.5: Horizontal collaboration between the datasets 1 and 3

explain this results are shown in the table 4.1.

In the figure 4.7 we shows the collaboration between the two maps SOM14 and SOM21 obtained after the collaboration between 1st and 4th maps; and between the 2nd and 1st map respectively. This result shows that in the case of collaborating of two maps containing relevant variables, the obtained collaborated map will contains high relevant variables which will increase the map accuracy (table 4.1). We remind that red color represent important variables for each map's prototype. We note obtained map SOM14-21.

As, the SOM14-21 map contains high relevant features, we will collaborate this one with the SOM41 map containing noise variables, but increasing the confidence parameter α to 8



(a) The map of the 1st dataset after collaboration with SOM4 : SOM14

(b) The map of the 4th dataset after collaboration with SOM1 : SOM41

Figure 4.6: Horizontal collaboration for datasets 1 and 4

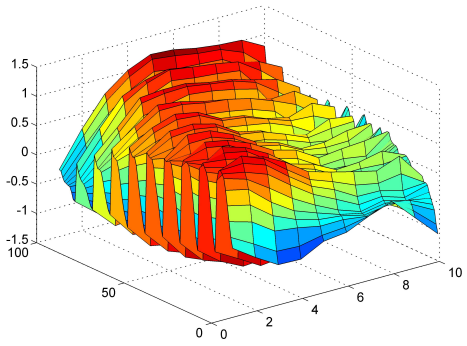


Figure 4.7: The SOM14 map after collaboration with SOM21 : SOM14-21

for the SOM14-21 map and decreasing α to 0.2 for the SOM41 map in order to weight the importance of the SOM14 map. As we can see on the figure 4.8, the collaborated obtained map changed its structure by increasing the importance of features $[4-9]$, that means that the important features from the first map high influenced on the noise features, and increase the purity index to 66.84%, and decreasing the quantization error to 2.01.

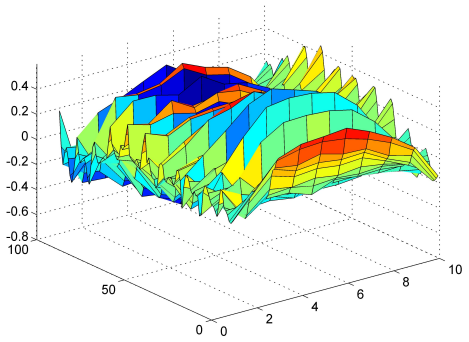


Figure 4.8: The SOM14-21->41 map after collaboration of SOM14-21 with SOM41 : SOM14-21->41

Table 4.1: Experimental validation of the horizontal collaboration

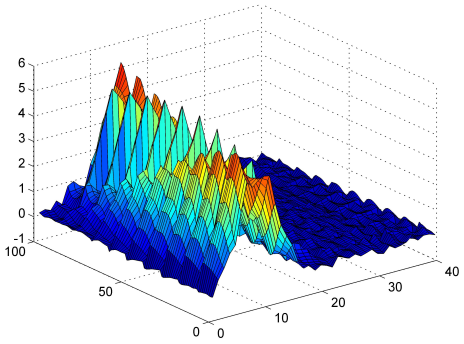
Measures	Purity	Quantization error
SOM1	75.71	1.98
SOM2	79.61	1.87
SOM3	47.19	2.64
SOM4	51.26	2.41
SOM12	76.21	1.93
SOM21	78.72	1.81
SOM13	62.47	2.14
SOM31	54.63	2.27
SOM14	64.17	2.05
SOM41	56.18	2.35
SOM14-21	69.67	1.94
SOM14-21->41	66.84	2.01

From table 4.1, we can notice that then collaborating a map with a higher purity index with another which has a smaller purity, the accuracy of the first one will decrease and viceversa. The same situation can be analyzed for the quantization error, where it will increase while collaborating a map with another which has a bigger quantization error. This is the main enigma of the collaborating technique: to know when the collaborating process will increase clustering validation indices; but in the unsupervised learning, computing these indices are not usually possible, so, the main goal remains to attempts the collaborating between maps.

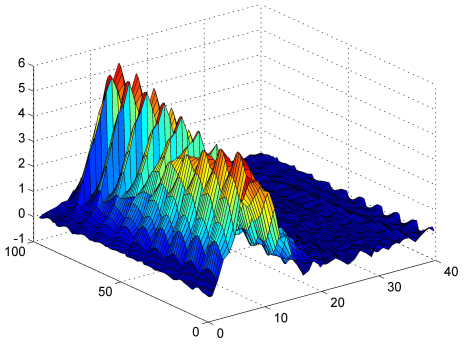
4.5.1.2 Improvement of the vertical approach

In order to have datasets with the same descriptive variables, the base waveform size 5000×40 was divided into four databases size 1250×40. So these datasets has the same variables but different observations labeled in three classes. We will use these four datasets to show the entire process of vertical collaborative SOM. The collaboration matrix β has been set at [0.8 0.8, 0.8 0.8]. We selected maps of dimension 10×10. Then we execute the local learning step on the first dataset which is to learn a SOM map (SOM1) for all the observations of the corresponding dataset. Another map (SOM2) has been used to learn the observations of the second dataset during the learning local step.

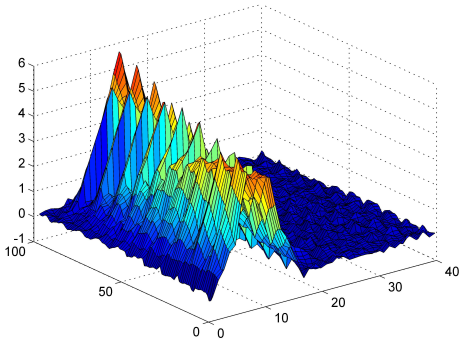
In the figure 4.9, the two axes X and Y represent respectively the variables and prototype indices. Figure 4.9(a) represents the 100 prototype vectors associated with neurons of the first SOM1 map with a purity index equal to 88.33% and a 5.64 quantization error. Figure 4.9(b) represents the 100 prototype vectors associated with neurons of the second SOM2 map which has an accuracy of 87.75%, and a quantization error to 5.83. It is clear from the two above-mentioned figures that the prototype vectors associated with these two maps are different because on the SOM map in 4.9(a), each vector w_j is regarded as the average of observations in the first dataset which has been projected onto the j-th neuron. Also, in figure 4.9(b), each vector w_j is considered as the average of all the observations of the second dataset which has been projected onto the j-th neuron.
After obtaining these two maps, the collaboration/cooperation learning step is initiated. As mentioned earlier, we can't assemble both databases for reasons of confidentiality that prevents the clustering of the entire dataset (the set of both datasets) on a single SOM map. Thus, we will use the collaborative process of collaborating the first dataset with the second map and the second dataset with the first map. This would allow repartition of observations of the first dataset to a new SOM map in order to obtain a similar map to the one which we would have got if datasets had collaborated since we use the information (prototype vectors) of the first map associated with the second dataset and similarly for the observations of the second map. Therefore the two maps after the vertical collaboration step must have similar



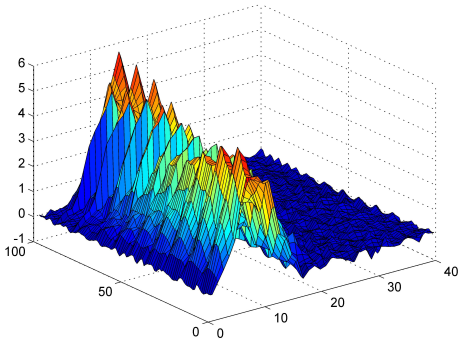
(a) The map of the 1st dataset before collaboration : SOM1



(b) The map of the 2nd dataset before collaboration : SOM2



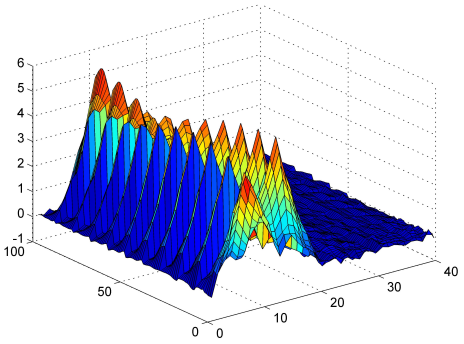
(c) The map of the 1st dataset after collaboration with SOM2 : SOM12



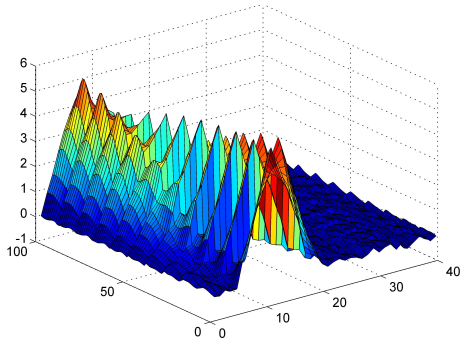
(d) The map of the 2nd dataset after collaboration with SOM1 : SOM21

Figure 4.9: Vertical collaboration for datasets 1 and 2

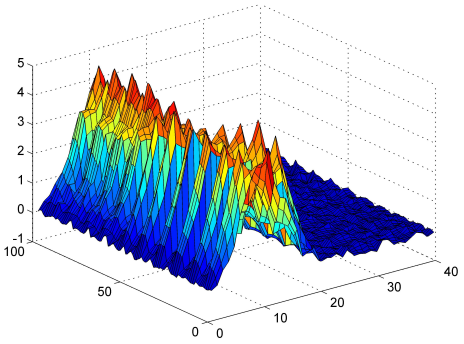
prototype vectors. Figure 4.9(c) and figure 4.9(d) represent respectively the prototype vectors obtained on the first and second map after the vertical collaboration step. Figure 4.9(c) represents the 100 prototype vectors (SOM12) associated with the first SOM1 map after the collaboration of the first dataset and the second SOM2 map of the local learning step. Figure 4.9(d) represents the 100 prototypes vectors (SOM21) associated with the second map after collaboration of the second dataset and the first SOM1 map of the local learning step. It is clear from the two above-mentioned figures that the prototype vectors associated with these two maps are similar. This proves that the collaboration actually happens and that two neurons j corresponding to the two collaborated maps represent similar classes/cells. The purity indices of these collaborated maps doesn't change radically: SOM12 has a 88.06% purity and SOM21 - 87.93%. The same situation is for the quantization errors: 5.62 for SOM12 and 5.79 for SOM21.



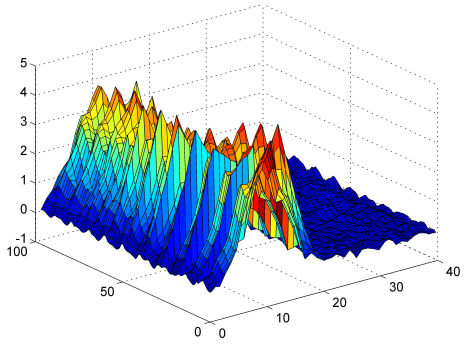
(a) The map of the 3st dataset before collaboration : SOM3



(b) The map of the 4th dataset before collaboration : SOM4



(c) The map of the 3st dataset after collaboration with SOM4 : SOM34



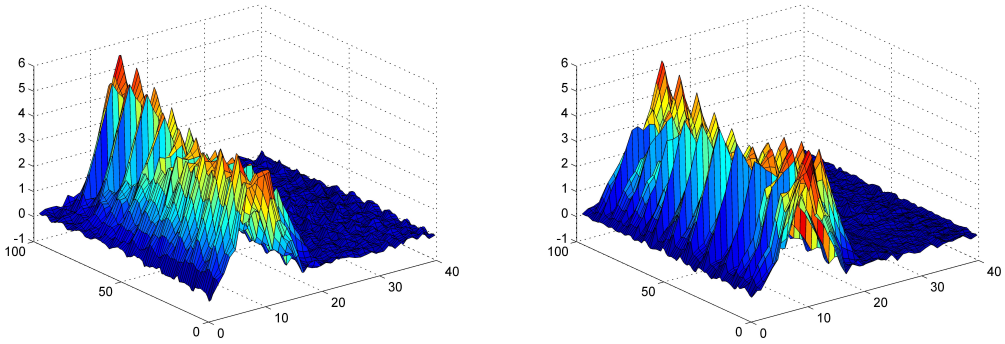
(d) The map of the 4th dataset after collaboration with SOM3 : SOM43

Figure 4.10: Vertical collaboration for datasets 3 and 4

We attempt a learning of another two maps using the third and fourth datasets during the local step of the collaborative learning approach, and the figure 4.10(a) and 4.10(b) show the corresponding two maps. The SOM3 map obtained from the third dataset has the important features $[1-21]$, and the most important variables (red color) were captured by the cells $[70-100]$. The quantization error of this map is equal to 5.24 and the accuracy of this one is 90.04%. Contrarily to this map, the SOM4 map (88.76% of purity and quantization error equal to 5.57) obtained from the 4th dataset, the most important variables ($[15-25]$) were captured by the cells $[1-40]$ because of using different observations from the original dataset. The SOM34 map was learned using the information from the SOM4 map and as we can see in the figure 4.10(c) the most relevant features ($[1-22]$) were captured by almost all the cells except some neurons $[40-60]$ which demonstrate that the collaboration between these two maps allowed to change the information and to give more importance to the features $[1-40]$ thanks to the SOM4 local map. The similar situation there is for the collaborated

SOM43 map which uses the fourth dataset and the SOM3 local map to collaborate. The variables [10 – 25] for the cells [50 – 100] are less important compared to the SOM34 map due to the SOM4 local map there the corresponding variables are less important compared to the variables for the cells [1 – 40]. As is shown in the table 4.2, the accuracy indices of these maps increase to 90.12% for the SOM34 map and to 89.57% for the SOM43 map, that means that during the collaboration step, the map accuracy can be increased if collaborating well-fined maps (high accuracy index).

We execute the vertical collaboration between the first (SOM1) and fourth (SOM4) and the results are shown in the figure 4.11. The SOM14 map (figure 4.11(a)) take the same structure as SOM1 4.9(a) by giving more importance to the variables [10 – 22] captured by cells [20−60] due to the collaboration with the SOM4 local map. Contrarily, the SOM41 map has a structure more similar to the SOM4 local map and the variables [15 – 22] for the cells [80 – 100] increase compared to the SOM4 map (figure 4.10(b)) due to the collaboration with the 1st map were these features are the most important. The purity indices and quantization errors doesn't change significantly, and can be analyzed in the table 4.2



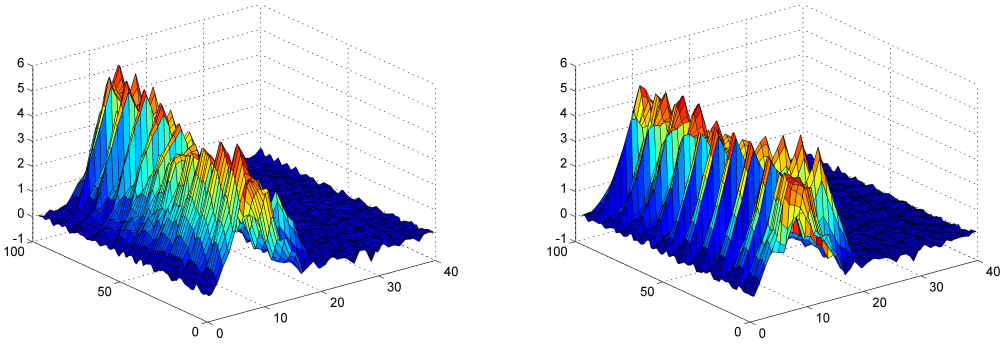
(a) The map of the 1st dataset after collaboration with SOM4 : SOM14

(b) The map of the 4th dataset after collaboration with SOM1 : SOM41

Figure 4.11: Vertical collaboration for datasets 1 and 4

As for the first and fourth maps, the same analysis can be done for the collaboration between the SOM2 local map and SOM3 local map. As the features [10 – 22] are important for the both local maps SOM2 (figure 4.9(b)) and SOM3 (figure 4.10(a)) but for different cells, almost all the cells of the both collaborated maps: SOM23 and SOM32 represented in the figure 4.12 give a high importance of these features. In this case, for the both maps, as in the case of the SOM34 and SOM43 maps, the purity indices increased compared to SOM3 and SOM4 maps, as well as quantization error decreased for the both maps (table 4.2).

Finally, we will made a collaboration between a local map and an already collaborated one.

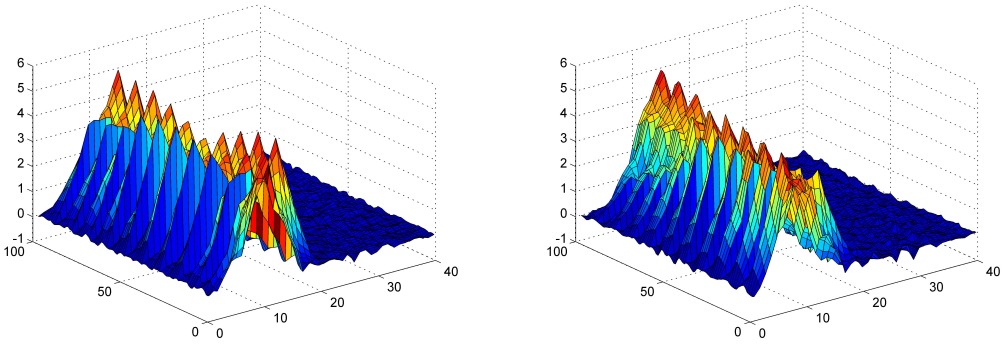


(a) The map of the 2nd dataset after collaboration with SOM3 : SOM23

(b) The map of the 3th dataset after collaboration with SOM2 : SOM32

Figure 4.12: Vertical collaboration for datasets 2 and 3

We will use the collaborated SOM23 map (figure 4.12(a)) obtained from the SOM2 (figure 4.9(b)) and SOM3 (figure 4.10(a)) maps, to change the information with the local SOM4 map (figure 4.10(b)). Obtained two maps presented in the figure 4.13 have similar characteristics (important variables) as the maps SOM23 and SOM32, although, the variables are more correlated between them for these new maps. The purity indices and quantization errors can be find in the table 4.2, which in this case doesn't change significantly.



(a) The SOM map after collaboration between SOM4 and SOM23 : SOM4-23

(b) The SOM map after collaboration of the SOM 23 and SOM4 : SOM23-4

Figure 4.13: Vertical collaboration between the maps SOM4 and SOM23

The results of this vertical collaboration experiment on the waveform dataset are summarized in table 4.2.

Table 4.2: Experimental validation of the vertical collaboration

Measures	Purity	Quantization error
SOM1	88.33	5.64
SOM2	87.75	5.83
SOM3	90.04	5.24
SOM4	88.76	5.57
SOM12	88.06	5.62
SOM21	87.93	5.79
SOM34	90.12	5.07
SOM43	89.57	5.16
SOM14	88.46	5.59
SOM41	88.57	5.51
SOM23	89.02	5.43
SOM32	90.97	4.96
SOM4-23	89.14	5.03
SOM23-4	88.83	5.48

4.5.2 Results on others data sets

We applied the experimental protocol for other datasets and all indices are presented in table 4.3 and 4.4 for the horizontal and vertical collaboration respectively.

Table 4.3: Experimentation results on different datasets for the horizontal collaboration

Dataset	DB..	Purity	QE
wdbc	SOM1	94.9550	1.9993
	SOM2	97.2777	2.0749
	SOM12	95.7798	1.8477
	SOM21	97.3262	1.9438
isolet 5x5	SOM1	81.2081	12.6149
	SOM2	95.1220	14.4591
	SOM12	81.3953	12.2188
	SOM21	96.0674	14.1876
madelon	SOM1	60.8879	15.5896
	SOM2	62.6402	15.5065
	SOM12	61.0132	15.4871
	SOM21	63.5744	15.4092
spam	SOM1	83.8603	3.4582
	SOM2	85.7205	2.5580
	SOM12	83.1774	3.2398
	SOM21	83.5983	2.4108

Concerning the other datasets, we will point out in this part the results obtained after applying our algorithm. All used maps have size 10x10 except the map for the isolet dataset which has the size 5x5. From Table 3, we note that the purity of SOM maps after horizontal collaboration increases for each dataset and the quantization error decreases. This is due to the use of map's information associated with the collaborative dataset.

For the wdbc dataset, we may note that the purity of the first map after vertical collaboration has improved. Contrarily, the purity of the second map after the collaboration has decreased. We also note that the quantization error of the first and second map was improved after the collaboration. For the isolet dataset, we do not notice any improvement on maps obtained from the collaboration compared to those before. For the database Madelon, we may note that the map's purity after the collaboration and the quantization error are improved. For the spambase dataset, quantization error has improved. These results show that the purity of the maps and the quantization error is not always improved during the collaboration.

Table 4.4: Experimentation results on different datasets for the vertical collaboration

Dataset	DB..	Purity	QE
wdbc	SOM1	96.7153	90.5413
	SOM2	97.8723	67.6035
	SOM12	96.9925	71.4997
	SOM21	97.4910	61.4791
isolet 5x5	SOM1	98.8506	8.1904
	SOM2	98.4615	8.7671
	SOM12	79.5455	8.3419
	SOM21	98.3051	8.7805
madelon	SOM1	69.7198	612.3251
	SOM2	69.8718	611.5365
	SOM12	74.5704	595.9780
	SOM21	70.7182	595.5441
spam	SOM1	76.2624	61.8324
	SOM2	70.4306	48.2763
	SOM12	72.2861	45.9831
	SOM21	69.7861	36.7421

In this section, we have found out that our vertical collaborative SOM approach improves the similarity between prototype vectors. If the vectors of the two map prototypes after the collaboration are similar and they have the same attributes in both datasets, then every two corresponding neurons on the map represent two similar classes.

We showed as well that our two approaches,: horizontal and vertical collaborative topological clustering, often improves the quality of maps (purity index and quantization error) after the collaboration. Even when it does not improve the purity or the quantization error of a map after the collaboration, they remain close to the values obtained on their corresponding local maps. In order to increase the collaboration results, the directional based clustering should be taken in consideration.

4.6 Conclusion

In this study, we have proposed two approaches for clustering of multiple databases derived from several datasets using Kohonen's self-organizing maps: horizontal collaborative SOM

and vertical collaborative SOM. The horizontal collaborative SOM approach is adapted for horizontal collaboration datasets that describe same observations but with different variables. The vertical collaborative SOM approach is appropriate to the problem of collaboration of several datasets containing the same variables as in the case of collaboration between several banks.

Thanks to the principle of these two approaches, data confidentiality is preserved. During the collaboration phase, each dataset is collaborated with all the maps obtained during the local phase. Thus, each site uses its dataset and the information from other SOM maps, thereby learning a new map that is similar to the map that would be obtained if we had centralized datasets and then clustering it. Thus, the SOM maps obtained after the collaboration step are similar. Both approaches have been validated on multiple datasets and experimental results have shown very promising performance. Several goals can be achieved, such as: automatically learn the matrices α and β together instead of fixing them, combine the horizontal and vertical collaborative SOM approaches to work out a new hybrid collaborative SOM approach; fusion all SOM maps obtained after the collaboration and construct a common map for all sites.

Conclusion

Contributions

In this dissertation, cluster analysis was extended in several directions driven by the complex data. The contributions are organized along four areas of clustering analysis techniques: memory based weighted topological learning, modular, hybrid and collaborative topological clustering, which are grouped in two main research axes: single model and multiple models based clustering approaches (figure 4.14).

For the memory based weighted topological clustering, we proposed several approaches of variable analysis during the clustering process. Ours methods are based on Self-organizing Map and the estimation of features weights is done in conjunction with the automatic clustering. These weights are global or local and associated with each referent of the self-organizing map. They reflect the local(global) importance of each variable for clustering. The weights are used for a local segmentation of the topological map, giving also the richest clustering taking into account the relevance of the variables. We introduced new learning approaches that takes into account all neighborhood information during the competition phase and thus allows the memory learning. We used the SOM algorithm as a basic learning algorithm by introducing a voting matrix for taking into account the competition history. To take into consideration the memory during the learning process, we proposed the *wm*-SOM algorithm which compute a voting matrix for each iteration.
Also, we have described a process for selecting relevant features in unsupervised learning paradigm using these new weighted approaches. These new methods are based on the SOM model and features weighting. Local weighted learning algorithms (*lwo*-SOM and *lwd*-SOM) and double weighting SOM (*dlw*-SOM) provide cluster characterization by determining the feature weights within each cluster. We described extensive testing using a new statistical method for unsupervised feature selection. Our approaches demonstrated the efficiency and effectiveness of this method in dealing with high dimensional data for simultaneous clustering and weighting.

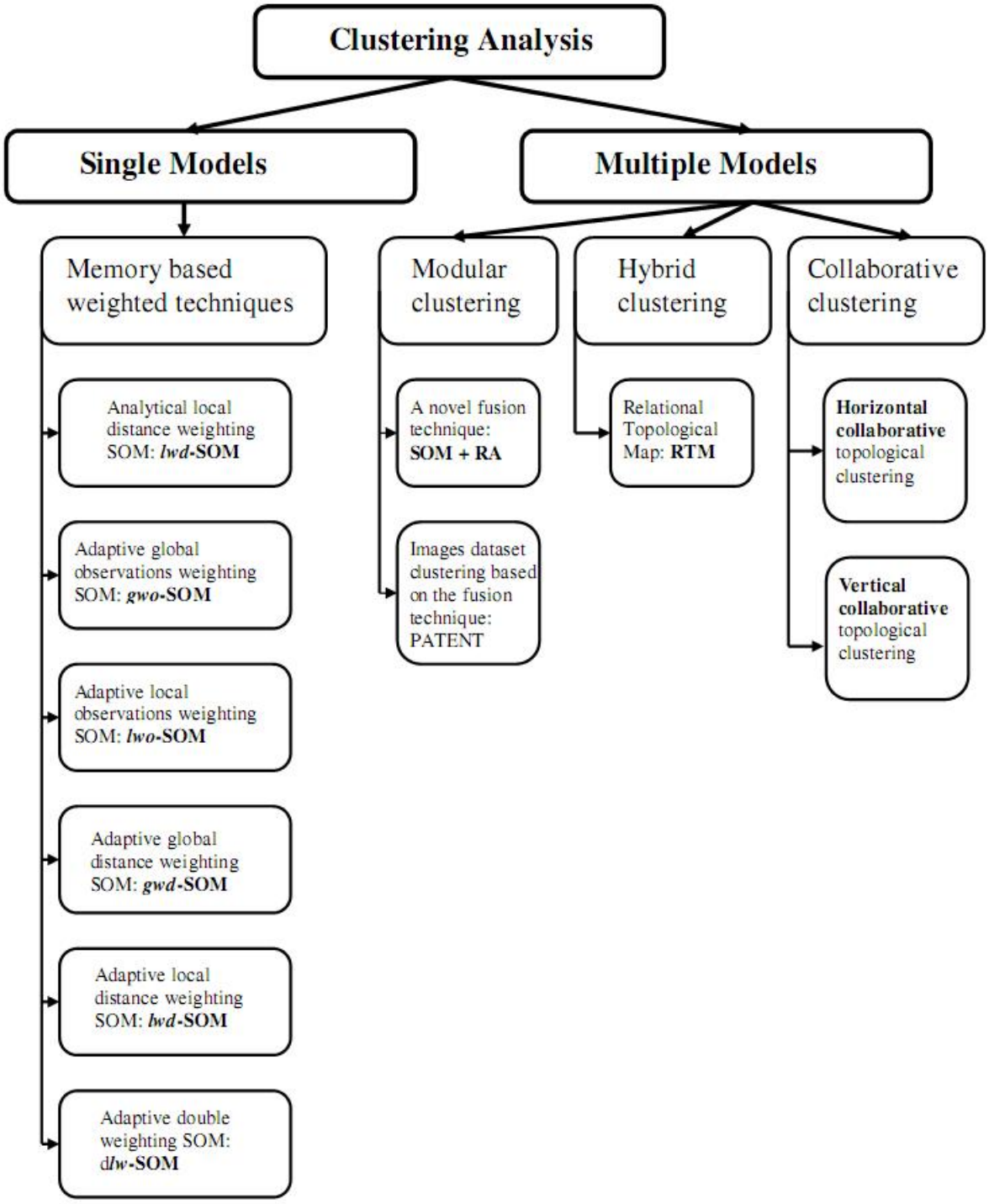


Figure 4.14: Summary of contributions in this dissertation

Concerning the modular based clustering, we presented an original and new approach of fusion/ensemble/consensus/aggregation clustering. The main idea was to find a clustering (or partition) of observations that represents the best consensus between several other clustering related to the same dataset. The goal of the proposed algorithm is the improvement of confidence in cluster assignments by evaluating a history of cluster assignments for each observation. If we compare our algorithm to some recent clustering algorithm, we can assert that, unlike these new algorithms, our method is scalable, linear, has a small computational time and can handle data represented as observations cross attributes or as similarity matrix. Another advantage of our method is that, neither do we need to re-process the data; nor do we need to fix the same cluster numbers for each application or clustering algorithm. For the application of the proposed fusion schema we presented a new solution to manage and process visual datasets. This process has been patented by us and Thales S.A. [15]. We proposed two scenarios: a clustering and a browsing schema which could be done simultaneously with the interactive learning. Also, we propose an original solution for searching in images libraries using the annotated text only to find the corresponding level of the map, and then to use the correlation between images on the map and inside the cell to display the information, in order to avoid the eventual noise (worst annotated text).
In order to attempts a hybrid based clustering, we proposed a new model for clustering and visualization of binary data based on the neighborhood relation from the classical SOM but using the formalism of the relational analysis approach. The proposed hybrid approach combines the advantages of both methods, indeed it allows a natural clustering identification without fixing the number of clusters (or cells).

Two approaches were proposed for clustering of multiple databases derived from several datasets using self-organizing maps: horizontal collaborative SOM and vertical collaborative SOM. The horizontal collaborative SOM approach is adapted for horizontal collaboration datasets that describe the same observations but with different variables. The vertical collaborative SOM approach is appropriate to the problem of collaboration of several datasets containing the same variables as in the case of collaboration between several banks.
Thanks to the principle of these two approaches, data confidentiality is preserved. During the collaboration phase, each dataset is collaborated with all the maps obtained during the local phase. Thus, each site uses its dataset and the information from other SOM maps, thereby learning a new map that is similar to the map that would be obtained if we had centralized datasets and then clustered them. Thus, the SOM maps obtained after the collaboration step are similar.
Collaborative clustering is useful to achieve interaction between different sources of information for the purpose of revealing underlying structures and regularities within data sets.

The introduced models of horizontal and vertical clustering achieve an active form of collaboration.

Future Work

This thesis offers several perspectives for future work. We can extend these models to take into account possible correlations between features and the robustness to noise. We can also, extend the algorithms to treat other types of features (categorical, mixed features) using an appropriate measure or distance.
For the *vm*-SOM approach, one of the perspectives is to use the estimated vote matrix during the learning process, to build a neighborhood matrix that can serve as input to the clustering algorithms.
The perspectives of the fusion schema is to perform a more detailed analysis involving huger dataset, and to attempts this model to characterize clusters after the fusion, by using some weighting/memory techniques during the learning process from the beginning (local learning) until the fusion.
Concerning the hybrid clustering, the proposed RTM model addresses in the same manner variables describing the data by ignoring their internal structures, the number of modalities per variable and the size of each modality. One of the perspective of the RTM approach is to ameliorate the learning process by weighting the Condorcet criteria used in this model.
For the collaborative clustering, several goals can be achieved, such as: automatically learn the matrices α and β together instead of fixing them, combine the horizontal and vertical collaborative SOM approaches to work out a new hybrid collaborative SOM approach; fusion of all SOM maps obtained after the collaboration and construct a common map for all sites. As the collaboration step depends on the maps between which collaboration is performed, the directional based clustering could be applied to detect the correct collaborating direction.

Finally, as all these proposed approaches are based on self-organization, they require some parameters during the learning/clustering process (learning step, size of map, β parameter for the g/lwd-SOM, α parameter for the RTM) and we want to extend these approaches to the autonomous machine learning techniques.

Appendix A

A.1 Data Sets

- **Waveform data set**: This data set consists of 5000 instances divided into 3 classes. The original base included 40 variables, 19 are all noise attributes with mean 0 and variance 1. Each class is generated from a combination of 2 of 3 "base" waves.

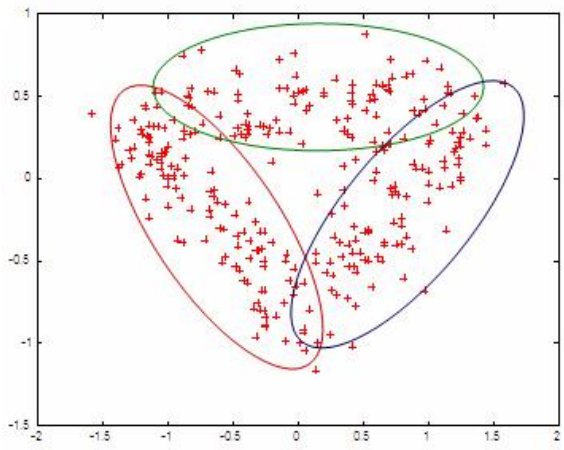


Figure A.1: Waveform dataset. 3 classes of waves

- **Wisconsin Diagnostic Breast Cancer (WDBC)**: This data has 569 instances with 32 variables (ID, diagnosis, 30 real-valued input variables). Each data observation is labeled as benign (357) or malignant (212). Variables are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

- **Isolet data set**: This data set was generated as follows. 150 subjects spoke the name of each letter of the alphabet twice. Hence, we have 52 training examples from each speaker. The speakers are grouped into sets of 30 speakers each, and are referred to as isolet1, isolet2, isolet3, isolet4, and isolet5. The data consists of 1559 instances and 617 variables. All variables are continuous, real-valued variables scaled into the range -1.0 to 1.0.

- **Madelon data set**: MADELON is an artificial dataset, which was part of the NIPS 2003 feature selection challenge. This is a two-class classification problem with continuous input variables. MADELON is an artificial dataset containing data points grouped in 32 clusters placed on the vertices of a five dimensional hypercube and randomly labeled +1 or -1. The five dimensions constitute 5 informative features. 15 linear combinations of those features were added to form a set of 20 (redundant) informative features. Based on those 20 features one must separate the examples into the 2 classes (corresponding to the +-1 labels). The order of the features and patterns were randomized. The original data set was splitting in three parts (learning, validation and test), but we used only 2600 observations from learning set and from validation for which the classes was known.

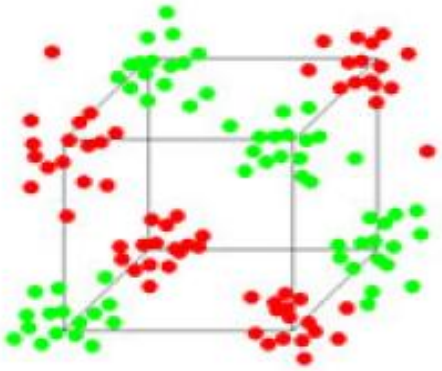


Figure A.2: Madelon data distribution

- **SpamBase data set**: The SpamBase data set is composed from 4601 observations described by 57 variables. Every variable described an e-mail and its category: spam or not-spam. Most of the attributes indicate whether a particular word or character was frequently occurring in the e-mail. The run-length attributes (55-57) measure the length of sequences of consecutive capital letters.

- **Heart disease data set:** this data set, which is D. Detrano's heart disease data set, was generated by the Clevlande Clinic. It consists in 303 observations, described by 6 numerical and 8 categorical variables. The observations are also classified into two classes: healthy class (buff) and with heart-disease class (sick).

- **Credit data set :** The data set has 690 instances, each being described by 6 continuous and 9 categorical variables. The observations were classified into two classes, approved class and rejected class.

- **Handwritten data:** this data set consists of the handwritten numerals ("0"−"9") extracted from a collection of Dutch utility maps. There are 200 samples of each digit such that there is a total of 2000 samples. Each sample is a 15×16 binary pixel image. The data set is represented as a 2000×240 binary data matrix. Each categorical variable is a pixel with two possible values "On=1" and "Off=0".

- **Wikipedia Dataset**. The dataset contains 17812 images extracted from wikipedia pages, each of which is described by it colors and texture. The images are accompanied with keyword-type annotations which specify a subset of available keywords for each image, and a web link on the wikipedia pages. The Fisher Kernels approach was used to obtain a numerical transformation of images by the Xerox Research Center [111]

- **2D2K** (http://strehl.com/). The 2D2K data set was artificially generated and contains 500 observations each of two 2-dimensional (2D) Gaussian clusters.

- **8D5K**. The 8D5K dataset contains 1000 observations from multivariate Gaussian distributions (200 observations each) in 8D space.

- **Nursery Database**

  Nursery Database was derived from a hierarchical decision model originally developed to rank applications for nursery schools. It was used during several years in 1980's when there was excessive enrollment to these schools in Ljubljana, Slovenia, and the rejected applications frequently needed an objective explanation. The final decision depended on three subproblems: occupation of parents and child's nursery, family structure and financial standing, and social and health picture of the family. The model was developed within expert system shell for decision making DEX.

  The Nursery Database contains examples with the structural information removed, i.e., directly relates NURSERY to the eight input attributes: parents, hasnurs, form, children, housing, finance, social, health.

This dataset has 12960 observations and 8 variables labeled in 6 classes.

- **Car Evaluation Database**. Car Evaluation Database was derived from a simple hierarchical decision model originally developed for the demonstration of DEX (M. Bohanec, V. Rajkovic: Expert system for decision making. Sistemica 1(1), pp. 145-157, 1990.). The model evaluates cars according their concept structure. The dataset represent a 4 classes problem containing 1728 observations and 6 variables.

- **Postoperative Patient Data.** The classification task of this database is to determine where patients in a postoperative recovery area should be sent to next. Because hypothermia is a significant concern after surgery (Woolery, L. et. al. 1991), the attributes correspond roughly to body temperature measurements. The dataset has 90 observations and 8 variables classified in 3 classes.

- **SPECTF heart dataset.** The dataset describes diagnosing of cardiac Single Proton Emission Computed Tomography (SPECT) images. Each of the patients is classified into two categories: normal and abnormal. The database of 267 SPECT image sets (patients) was processed to extract features that summarize the original SPECT images. As a result, 44 continuous feature pattern was created for each patient. This dataset contains 267 observations and 45 continuous variables

- **Zoo dataset.** This dataset contains 101 animals described with 16 qualitative variables: 15 of the variables are binary and one is numeric with 6 possible values. Each animal is labelled 1 to 7 according to its class. Using disjunctive coding for all variables with 6 possible values for the numerical variable, the data set consists of a 101x36 binary data matrix.

A.2 Implimentation details

For imlpimentation of the procedures described in this work we used Java and Matlab languages. The Java UIMA platform was used to implement the anlytical *lwd*-SOM technique, and for the Cluster characterization procedure we use Matlab with the Statistical and SOM tolboox. The image browsing technique were implemented in Matlab in the 2D; for the 3D browsing the procedure were implemented in Java language with PHP and SQL by Thales S.A.

Appendix B

B.1 Publications

1) GROZAVU N., BENNANI Y. (2009), «A new competitive strategy for Self Organizing Map Learning», Proceedings of the (ICMLA'09), International Conference on Machine Learning and Applications, 13-15 Dec. 2009, Miami Beach, Florida, USA.

2) GROZAVU N., BENNANI Y. (2009), «Voting Memory based Self-Organizing Map», TopoLearn'09: International Workshop on Topological Learning, ISMIS'09, International Symposium on Methodologies for Intelligent Systems, Prague, September 14 - 17, 2009.

3) GROZAVU N., BENNANI Y., LEBBAH M. (2009), «Cluster-dependent feature selection through a weighted learning paradigm», Invited Book Chapter, Accepted, to appear in Advances in Knowledge Discovery and Management, Springer Publisher.

4) GROZAVU N., ROGOVSCHI N. (2009), «Mining Visual Data», Proceedings of the ICMCS'09, 1-3 Oct. 2009, Chisinau, Moldova.

5) BENHADDA H., BENNANI Y., LEBBAH M., GROZAVU N. (2008) «SYSTEME DE RECHERCHE D'INFORMATION VISUELLE», BREVET 08 06947.

6) GROZAVU N., BENNANI Y. (2009), «Apprentissage topographique non supervisé avec mémoire», CAp'09 : Conférence francophone sur l'apprentissage automatique, Plate-forme AFIA, 25-29 Mai, Hammamat-Tunisie. - Best Poster Award CAp'09

7) LEBBAH M., BENNANI Y., BENHADDA H., GROZAVU N., (2009), «Relational Analysis for Clustering Consensus», Invited Book Chapter, Accepted, to appear in Machine Learning, ISBN 978-953-7619-X-X, IN-TECH Publisher.

8) GROZAVU N., BENNANI Y., LEBBAH M. (2009), «From variable weighting to cluster characterization in topographic unsupervised learning», Proc. IJCNN '09, International Joint Conference on Neural Network, 14-19 June 2009, Atlanta, Georgia.

9) GROZAVU N., BENNANI Y., LEBBAH M. (2009), «Caractérisation automatique des classes découvertes en classification non supervisée», Proc. of the EGC'09, Strasbourg, 27- 30 Janvier 2009 – RNTI, Revue des Nouvelles Technologies de l'Information, Editions Cépaduès.

10) BENNANI Y., LEBBAH M., GROZAVU N., MARCOTORCHINO J.F., BENHADDA H., LORIN S. (2008), « Analyse relationnelle comme algorithme de fusion de partitionnement», Workshop Infomagic, Analyse multimodale de l'information, Pôle de compétitivité Cap digital, 10 juin 2008 Telecom-ParisTech.

11) GROZAVU N., BENNANI Y., LEBBAH M. (2008), «Pondération locale des variables en apprentissage numérique non-supervisé», Proc. of the EGC'08, Sophia Antipolis-Méditerannée, 29 janvier - 1er février – RNTI, Revue des Nouvelles Technologies de l'Information,, Editions Cépaduès. Nominated Paper

12) GROZAVU N., BENNANI Y., GUERIF S. (2007), Towards a dimensionality reduction through unsupervised learning and local variable weighting», Proceedings of the ICMCS'07, September 19-21, 2007, Chisinau, Moldavie.

Bibliography

[1] Evolving data into mining solutions for insights. *Commun. ACM*, 45(8):28–31, 2002.

[2] Blansche A., Gancarski P., and Korczak J.J. Maclaw: A modular approach for clustering with local attribute weighting. *Pattern Recognition Letters*, 27(11):1299–1306, 2006.

[3] W. Punch A. Topchy, A. Jain. Combining multiple weak clusterings. *Third IEEE International Cnference on Data Mining*, pages 331–338, 2003.

[4] Charu C. Aggarwal, Joel L. Wolf, Philip S. Yu, Cecilia Procopiuc, and Jong Soo Park. Fast algorithms for projected clustering. In *SIGMOD '99: Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, pages 61–72, New York, NY, USA, 1999. ACM.

[5] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data, 2005.

[6] Mayer Aladjem. Linear discriminant analysis for two classes via removal of classification structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):187–192, 1997.

[7] M. R. Anderberg. *Cluster Analysis for Applications*. New York: Academic Press, 1983.

[8] U. Arsuaga and M. Diaz. Topology preservation in som. *Proceedings of World Academy of Science, Enginneering and Technology*, 15, 2006.

[9] A. Asuncion and D.J. Newman. Uci machine learning repository, 2007.

[10] Javad Azimi, Monireh Abdoos, and Morteza Analoui. A new efficient approach in clustering ensembles. In *IDEAL, International Conference on Intelligent Data Engineering and Automated Learning*, 2007.

[11] Arindam Banerjee and Joydeep Ghosh. On scaling up balanced clustering algorithms. In *SDM*, 2002.

[12] Jayanta Basak, Rajat K. De, and Sankar K. Pal. Unsupervised feature selection using a neuro-fuzzy approach. *Pattern Recogn. Lett.*, 19(11):997–1006, 1998.

[13] Robert Beiko and Robert Charlebois. Gann: Genetic algorithm neural networks for the detection of conserved combinations of features in dna. *BMC Bioinformatics*, 6(1):36+, February 2005.

[14] H. Benhadda and F. Marcotorchino. L'analyse relationnelle pour la fouille de grandes bases de données. In *Revue des Nouvelles Technologies de l'Information*, RNTI-A-2, pages 149–167. Cépaduès, 2007.

[15] LEBBAH M. GROZAVU N. BENHADDA H., BENNANI Y. In *BREVET (THALES, UNIVERSITE PARIS 13) N: 08 06947*.

[16] Y. Bennani. Adaptive weighting of pattern features during learning. In *IJCNN'99. International Joint Conference on Neural Networks. Proceedings*, volume 5, pages 3008–13, Piscataway, NJ, 1999. IEEE Service Center.

[17] James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 1981.

[18] C. M. Bishop, M. Svensén, and C. K. I.Williams. Gtm: The generative topographic mapping. *Neural Comput*, 10(1):215–234, 1998.

[19] Alexandre Blansché. *Classification non supervisé avec pondération d'attributs par des méthodes évolutionnaires*. PhD thesis, University of Louis Pasteur - Strasbourg I, Septembre 2006.

[20] A. L. Boulesteix and K. Strimmer. Partial least squares: a versatile tool for the analysis of high-dimensional genomic data. *Brief Bioinform*, 8(1):32–44, January 2007.

[21] R. Cattell. The scree test for the number of factors. *Multivariate Behavioral Research*, 1:245–276, 1966.

[22] Jae W. Chang and Du S. Jin. A new cell-based clustering method for large, high-dimensional data in data mining applications. In *Proceedings of the 2002 ACM symposium on Applied computing*, pages 503–507. ACM Press, 2002.

[23] Geoffrey J. Chappell and John G. Taylor. The temporal kohonen map. *Neural Netw.*, 6(3):441–445, 1993.

[24] Chun-Hung Cheng, Ada Waichee Fu, and Yi Zhang. Entropy-based subspace clustering for mining numerical data. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 84–93, New York, NY, USA, 1999. ACM.

[25] Shu-Chuan Chu, John F. Roddick, and Jeng-Shyang Pan. Improved search strategies and extensions to k-medoids-based clustering algorithms. *Int. J. Bus. Intell. Data Min.*, 3(2):212–231, 2008.

[26] Maquis Nicolas De Condorcet. Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix. *De l'imprimerie royale, Paris*, 1785.

[27] Marie Cottrell and Michel Verleysen. Advances in self organising maps. *CoRR*, abs/cs/0611058, 2006.

[28] Glen Cowan. *Statistical data analysis*. Oxford University Press, Oxford, UK, 1998.

[29] Juan Antonio Cuesta-Albertos and Ricardo Fraiman. Impartial trimmed k-means for functional data. *Comput. Stat. Data Anal.*, 51(10):4864–4877, 2007.

[30] Manoranjan Dash, Huan Liu, and Xiaowei Xu. '1 + 1 > 2': Merging distance and density based clustering. In *In Database Systems for Advanced Applications, 2001. Proceedings. Seventh International Conference on*, pages 32–39, 2001.

[31] William H. Day and Herbert Edelsbrunner. Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of Classification*, 1(1):7–24, December 1984.

[32] D.W. Bouldin D.L. Davies. A cluster separation measure. *IEEE Trans. Pattern Anal. Machine Intell.*, 1 (4):224–227, 1974.

[33] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley, 2001.

[34] Sandrine Dudoit and Jane Fridlyand. Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*, 19, 2003.

[35] Haytham Elghazel, Tetsuya Yoshida, and Mohand-Said Hacid. An integrated graph and probability based clustering framework for sequential data. In *DS '08: Proceedings of the 11th International Conference on Discovery Science*, pages 246–258, Berlin, Heidelberg, 2008. Springer-Verlag.

[36] Martin Ester, Hans-peter Kriegel, S. Jörg, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise, 1996.

[37] Brian Everitt, Sabine Landau, and Morven Leese. *Cluster analysis.* Arnold ; Oxford University Press, May 2001.

[38] Douglas Fisher. Iterative optimization and simplification of hierarchical clusterings. *Journal of Artificial Intelligence Research. (JAIR)*, 4:147–178, 1996.

[39] Chris Fraley and Adrian E. Raftery. Model-based clustering, discriminant analysis and density estimation. *Journal of the American Statistical Association*, 97:611–631, 2002.

[40] Jerome H. Friedman and Jacqueline J. Meulman. Clustering objects on subsets of attributes (with discussion). *Journal Of The Royal Statistical Society Series B*, 66(4):815–849, 2004.

[41] Jerome H. Friedman and Charles B. Roosen. An introduction to multivariate adaptive regression splines. *Stat Methods Med Res*, 4(3):197–217, September 1995.

[42] Hichem Frigui and Olfa Nasraoui. Unsupervised learning of prototypes and attribute weights. *Pattern Recognition 37(3)*, pages 567–581, 2004.

[43] Dimitrios S. Frossyniotis, Christos Pateritsas, and Andreas Stafylopatis. A multi-clustering fusion scheme for data partitioning, 2005.

[44] Dimitrios S. Frossyniotis, Minas Pertselakis, and Andreas Stafylopatis. A multi-clustering fusion algorithm. In *SETN '02: Proceedings of the Second Hellenic Conference on AI*, pages 225–236, London, UK, 2002. Springer-Verlag.

[45] Saporta G. *ProbabilitÈs, analyse des donnÈes et statistiques*. Editions Technip, 2006.

[46] Jerome Galtier. In *Topological Learning*.

[47] Byron J. Gao and Martin Ester. Cluster description formats, problems and algorithms. In *SDM*, 2006.

[48] Gautam Garai and B. B. Chaudhuri. A novel genetic algorithm for automatic clustering. *Pattern Recogn. Lett.*, 25(2):173–187, 2004.

[49] Aristides Gionis, Heikki Mannila, and Panayiotis Tsaparas. Clustering aggregation. *ACM Trans. Knowl. Discov. Data*, 1(1):4, 2007.

[50] Sanjay Goil, Harsha Nagesh, and Alok Choudhary. MAFIA: Efficient and scalable subspace clustering for very large data sets. Technical Report CPDC-TR-9906-010, Northwestern University, 2145 Sheridan Road, Evanston IL 60208, June 1999.

[51] David E. Goldberg and John H. Holland. Genetic algorithms and machine learning. *Machine Learning*, 3:95–99, 1988.

[52] N. Grozavu, Y. Bennani, and M. Lebbah. Pondération locale des variables en apprentissage numérique non-supervisé. *Extraction et Gestion des Connaissances (EGC 08)*, pages 45–54, 2008.

[53] Sébastien Guérif, Younès Bennani, and Eric Janvier. μ-SOM : Weighting features during clustering. In *Proceedings of the 5th Workshop On Self-Organizing Maps (WSOM'05)*, pages 397–404, Paris 1 Panthéon-Sorbonne University, France, September 2005.

[54] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Rock: A robust clustering algorithm for categorical attributes. *Data Engineering, International Conference on*, 0:512, 1999.

[55] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Rock: A robust clustering algorithm for categorical attributes. *Inf. Syst.*, 25(5):345–366, 2000.

[56] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003.

[57] Timo Hämmäinen, Harri Klapuri, Jukka Saarinen, and Kimmo Kaski. Mapping of som and lvq algorithms on a tree shape parallel computer system. *Parallel Comput.*, 23(3):271–289, 1997.

[58] Pierre Hansen and Brigitte Jaumard. Cluster analysis and mathematical programming. *Math. Program.*, 79:191–215, 1997.

[59] Xiaofei He, Deng Cai, and Partha Niyogi. Laplacian score for feature selection. In *NIPS*, 2005.

[60] Alexander Hinneburg, Er Hinneburg, and Daniel A. Keim. An efficient approach to clustering in large multimedia databases with noise. pages 58–65. AAAI Press, 1998.

[61] Alexander Hinneburg Hinneburg and Daniel A. Keim. Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. pages 506–517. Morgan Kaufmann, 1999.

[62] John H. Holland. Genetic algorithms and the optimal allocation of trials. *SIAM J. Comput.*, 2(2):88–105, 1973.

[63] John H. Holland. Erratum: Genetic algorithms and the optimal allocation of trials. *SIAM J. Comput.*, 3(4):326, 1974.

[64] John H. Holland. Genetic algorithms and classifier systems: Foundations and future directions. In *ICGA*, pages 82–89, 1987.

[65] Joshua Zhexue Huang, Michael K. Ng, Hongqiang Rong, and Zichen Li. Automated variable weighting in k-means type clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):657–668, 2005.

[66] Zhexue Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Min. Knowl. Discov.*, 2(3):283–304, 1998.

[67] A. Gordaliza J. A. Cuesta-Albertos and C. Matran. Trimmed k-means: an attempt to robustify quantizers. *Ann. Statist.*, 25(2):553–576, 1997.

[68] A. K. Jain and R. C. Dubes. Algorithms for clustering data. *PrenticeHall, Englewood Clis*, 1988.

[69] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[70] Anil K. Jain and Richard C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[71] Charbel Julien and Lorenza Saitta. Image databases browsing by unsupervised learning. In *ISMIS*, pages 389–398, 2008.

[72] U. Kaymak and M. Setnes. Extended fuzzy clustering algorithms. Research paper, November 2000.

[73] Seo Young Kim and Won Lee. Ensemble clustering method based on the resampling similarity measure for gene expression. *Stat Methods Med Res*, 16:539–564, 2007.

[74] K. Kiviluoto. Topology preservation in self-organizing maps. *In the proceedings of International Conference on Neural Networks (ICNN)*, 1996.

[75] T. Kohonen. *Self-organizing Maps*. Springer-Verlag Berlin, Berlin, 1995.

[76] T. Kohonen. *Self-organizing Maps*. Springer Berlin, 2001.

[77] Markus Koskela. Interactive image retrieval using self-organizing maps. *Dissertation Report*, 2003.

[78] K. Krishna and M. Narasimha Murty. Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 29(3):433–439, 1999.

[79] Sunil Kukreja, J. Löfberg, and Martin J. Brenner. A Least Absolute Shrinkage and Selection Operator (LASSO) for Nonlinear System Identification. Technical report, September 2005.

[80] Jorma Laaksonen, Markus Koskela, Sami Laakso, and Erkki Oja. Picsom—content-based image retrieval with self-organizing maps. *Pattern Recogn. Lett.*, 21(13-14):1199–1207, 2000.

[81] Lazhar Labiod. *Contribution au formalisme relationnel des classifications simultanées de deux ensembles*. PhD thesis, The Paris 6 University, December 2008.

[82] M. Lebbah, N. Rogovschi, and Y. Bennani. A probabilistic self-organizing map for binary data topographic clustering. In *IJCIA, World Scientific Publishing Company*, 2008.

[83] M. Lebbah, S. Thiria, and F. Badran. Topological map for binary data. In *Proceedings European Symposium on Artificial Neural Networks-ESANN 2000, Bruges, April 26-27-28*, pages 267–272, 2000.

[84] Mustapha Lebbah, Aymeric Chazottes, Fouad Badran, and Sylvie Thiria. Mixed topological map. In *ESANN*, pages 357–362, 2005.

[85] Mustapha Lebbah, Nicoleta Rogovschi, and Younès Bennani. Besom : Bernoulli on self organizing map. In *IJCNN 07*, Orlando, Florida.

[86] Daoying Ma and Aidong Zhang. An adaptive density-based clustering algorithm for spatial database with noise. *Data Mining, IEEE International Conference on*, 0:467–470, 2004.

[87] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[88] François Marcotorchino and Pierre Michaud. Optimisation en analyse ordinale des données. *Biometrica*, 31:324–345, 1978.

[89] Pierre Michaud. Agrégation à la majorité ii: analyse du résultat d'un vote. *Etude du centre scientifique IBM France*, F.094, 1985.

[90] Pierre Michaud and François Marcotorchino. Modèles d'optimisation en analyse des données relationnelles. *Mathématiques et Sciences Humaines*, 17(67):7–38, 1979.

[91] Pierre Michaud and Francois Marcotorchino. Modeles d'optimisation en analyse des données rélationnelles. *Math. Sci. Hum.*, 67:7–38, 1979.

[92] Pierre Michaud and Francois Marcotorchino. Optimisation en analyse des données rélationnelles. Data analysis and informatics, Proc. int. Symp., Versailles 1979, 655-670 (1980)., 1980.

[93] Behrouz Minaei-Bidgoli, Alexander Topchy, and William F. Punch. Ensembles of partitions via data resampling. In *ITCC '04: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04) Volume 2*, page 188, Washington, DC, USA, 2004. IEEE Computer Society.

[94] Sushmita Mitra, Haider Banka, and Witold Pedrycz. Collaborative rough clustering. In *PReMI*, pages 768–773, 2005.

[95] Stefano Monti, Pablo Tamayo, Jill Mesirov, and Todd Golub. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. *Mach. Learn.*, 52(1-2):91–118, 2003.

[96] E. S. Motakis, G. P. Nason, P. Fryzlewicz, and G. A. Rutter. Variance stabilization and normalization for one-color microarray data using a data-driven multiscale approach. *Bioinformatics*, 22(20):2547–2553, 2006.

[97] M. Narasimha Murty, Rashmin Babaria, and Chiranjib Bhattacharyya. Clustering based on genetic algorithms. In *Multi-Objective Evolutionary Algorithms for Knowledge Discovery from Databases*, pages 137–159. 2008.

[98] Harsha Nagesh, Sanjay Goil, and Alok Choudhary. Adaptive grids for clustering massive data sets. In *In 1st SIAM International Conference Proceedings on Data Mining*, 2001.

[99] Amir Navot, Lavi Shpigelman, Naftali Tishby, and Eilon Vaadia. Nearest neighbor based feature selection for regression and its application to neural activity. In *NIPS*, 2005.

[100] Bryan D. Nelsonm. Variable reduction for modeling using proc varclus. *In Conference Proceedings of the Twenty-Sixth Annual SAS® Users Group International Conference. Cary, NC: SAS Institute Inc.*, 2001.

[101] Danh V. Nguyen and David M. Rocke. On partial least squares dimension reduction for microarray-based classification: a simulation study. *Computational Statistics and Data Analysis*, 46(3):407–425, 2004.

[102] Mustapha Lebbah Nistor Grozavu, Younès Bennani. From variable weighting to cluster characterization in topographic unsupervised learning. In *in Proc. Proc. of IJCNN09, International Joint Conference on Neural Network*, 2009.

[103] Koikkalainen P. Progress with the tree-structured self-organizing map. In *in Proc. 11th Europ. Conf. Artificial Intell.*, 1994.

[104] Lance Parsons, Ehtesham Haque, and Huan Liu. Evaluating subspace clustering algorithms. In *Workshop on Clustering High Dimensional Data and its Applications, SIAM International Conference on Data Mining (SDM) 2004*, pages 48–56, 2004.

[105] Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace clustering for high dimensional data: a review. *SIGKDD Explor. Newsl.*, 6(1):90–105, June 2004.

[106] Witold Pedrycz. Collaborative fuzzy clustering. *Pattern Recognition Letters*, 23(14):1675–1686, 2002.

[107] Witold Pedrycz. Fuzzy clustering with a knowledge-based guidance. *Pattern Recogn. Lett.*, 25(4):469–480, 2004.

[108] Witold Pedrycz. Interpretation of clusters in the framework of shadowed sets. *Pattern Recogn. Lett.*, 26(15):2439–2449, 2005.

[109] Witold Pedrycz and Kaoru Hirota. A consensus-driven fuzzy clustering. *Pattern Recogn. Lett.*, 29(9):1333–1343, 2008.

[110] Dan Pelleg and Andrew Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *In Proceedings of the 17th International Conf. on Machine Learning*, pages 727–734, 2000.

[111] F. Perronnin and C.R. Dance. Fisher kernels on visual vocabularies for image categorization. pages 1–8, 2007.

[112] G. Pölzlbauer. Survey and comparison of quality measures for self-organizing maps. *WDA, Elfa Academic Press*, 2004.

[113] Christian Posse. Hierarchical model-based clustering for large datasets. *Journal of Computational and Graphical Statistics*, 10(3):464–??, 2001.

[114] W.M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association.*, pages 846–850, 1971.

[115] David Rogers. G/splines: A hybrid of friedman's multivariate adaptive regression splines (mars) algorithm with holland's genetic algorithm. In *ICGA*, pages 384–391, 1991.

[116] R.J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics.*, 20:53–65, 1987.

[117] Jörg Sander, Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data Min. Knowl. Discov.*, 2(2):169–194, 1998.

[118] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 2(6):461–464, 1978.

[119] Guérif Sébastien and Bennani Younès. Dimensionality reduction trough unsupervised features selection. *International Conference on Engineering Applications of Neural Networks*, 2007.

[120] Yves Grandvalet Stephane and Stephane Canu. Outcomes of the equivalence of adaptive ridge with least absolute shrinkage. In *In Advances in Neural Information Processing Systems*, pages 445–451. MIT Press, 1998.

[121] Alexander Strehl. *Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining*. PhD thesis, The University of Texas at Austin, May 2002.

[122] Alexander Strehl and Joydeep Ghosh. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *Journal on Machine Learning Research (JMLR)*, 3:583–617, December 2002.

[123] Alexander Strehl and Joydeep Ghosh. Cluster ensembles – a knowledge reuse framework for combining partitionings. In *Proc. Conference on Artificial Intelligence (AAAI 2002), Edmonton*, pages 93–98. AAAI/MIT Press, July 2002.

[124] Alexander P. Topchy, Anil K. Jain, and William F. Punch. A mixture model for clustering ensembles. In *SDM,proceedings of the Fourth SIAM International Conference on Data Mining, Lake Buena Vista, Florida, USA, April 22-24*, 2004.

[125] Member-Alexander Topchy, Fellow-Anil K. Jain, and William Punch. Clustering ensembles: Models of consensus and weak partitions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(12):1866–1881, 2005.

[126] Cheng-Fa Tsai and Chien-Tsung Wu. Gf-dbscan: a new efficient and effective data clustering technique for large databases. In *MUSP'09: Proceedings of the 9th WSEAS international conference on Multimedia systems & signal processing*, pages 231–236, Stevens Point, Wisconsin, USA, 2009. World Scientific and Engineering Academy and Society (WSEAS).

[127] J.J. Verbeek, N. Vlassis, and B.J.A. Krose. Self-organizing mixture models. *Neurocomputing*, 63:99–123, 2005.

[128] M. Verleysen. *Machine learning of high-dimensional data: local artificial neural networks and the curse of dimensionality*. Agregation in higher education thesis, Catholic University of Louvain, Louvain-la-Neuve, BELGIUM, 2001.

[129] M. Verleysen. *Limitations and future trends in neural computation*, chapter Learning high-dimensional data, pages 141–162. IOS Press, 2003.

[130] M. Verleysen, D. François, G. Simon, and V. Wertz. On the effects of dimensionality on data analysis with neural networks. In J.R. Alvarez eds. J. Mira, editor, *Artificial Neural Nets Problem solving methods*, Lecture Notes in Computer Science 2687, pages II105–II112. Springer-Verlag, 2003.

[131] J. Vesanto and E. Alhoniemi. Clustering of the self-organizing map. *Neural Networks, IEEE Transactions on*, 11(3):586–600, May 2000.

[132] P. Viswanath and Rajwala Pinkesh. l-dbscan: A fast hybrid density based clustering method. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*, pages 912–915, Washington, DC, USA, 2006. IEEE Computer Society.

[133] Nirmalie Wiratunga, Robert Lothian, and Stewart Massie. Unsupervised feature selection for text data. In Thomas Roth-Berghofer, Mehmet H. Goker, and H. Altay Guvenir, editors, *ECCBR*, volume 4106 of *Lecture Notes in Computer Science*, pages 340–354. Springer, 2006.

[134] Zhongzhe Xiao, Emmanuel Dellandréa, Weibei Dou, and Liming Chen. ESFS: A new embedded feature selection method based on SFS. Technical Report RR-LIRIS-2008-018, LIRIS UMR 5205 CNRS/INSA de Lyon/Université Claude Bernard Lyon 1/Université Lumière Lyon 2/Ecole Centrale de Lyon, September 2008.

[135] Xiaowei Xu, Martin Ester, Hans peter Kriegel, and Jörg S. A distribution-based clustering algorithm for mining in large spatial databases. pages 324–331, 1998.

[136] Méziane Yacoub and Younès Bennani. Features selection and architecture optimization in connectionist systems. *IJNS*, 10(5), 2000.

[137] Méziane Yacoub and Younès Bennani. Une mesure de pertinence pour la sélection de variables dans les perceptrons multicouches. *RIA, Apprentissage connexionniste*, pages 393–410, 2001.

[138] Jieping Ye, Ravi Janardan, and Qi Li. Two-dimensional linear discriminant analysis. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1569–1576. MIT Press, Cambridge, MA, 2005.

[139] F. Zehraoui and Y. Bennani. New self-organizing maps for multivariate sequences procing. volume 5, pages 439–456, 2008.

[140] Daoqiang Zhang, Songcan Chen, and Zhi-Hua Zhou. Constraint score: A new filter method for feature selection with pairwise constraints. *Pattern Recogn.*, 41(5):1440–1451, 2008.

Résumé

Cette thèse est consacrée d'une part, à l'étude d'approches de caractérisation des classes découvertes pendant l'apprentissage non-supervisé, et d'autre part, à la classification non-supervisée modulaire, hybride et collaborative. L'étude se focalise essentiellement sur deux axes :
1) la caractérisation des classes en utilisant la pondération et la sélection des variables pertinentes, ainsi que l'utilisation de la notion de mémoire pendant le processus d'apprentissage topologique non-supervisé; 2) l'utilisation de plusieurs techniques de clustering en parallèle et en série : approches modulaires, hybrides et collaboratives.
Nous nous intéressons plus particulièrement dans cette thèse aux cartes auto-organisatrices de Kohonen qui constituent une technique bien adaptée à la classification non-supervisée permettant une visualisation des résultats sous forme d'une carte topologique. Nous proposons plusieurs techniques de pondérations de l'apprentissage de ces cartes ainsi qu'une nouvelle stratégie de compétition permettant de garder en mémoire l'historique de l'apprentissage. En utilisant un test statistique pour la sélection des variables pertinentes pondérées, nous répondons au problème de la réduction des dimensions, ainsi qu'au problème de la caractérisation des classes découvertes.
Concernant le deuxième axe, nous utilisons le formalisme mathématique de l'analyse relationnelle (AR) pour combiner plusieurs résultats de classification. Les avantages de l'AR sont combinés avec ceux des cartes auto-organisatrices pour proposer un nouvel algorithme RTM (Relational Topological Map) alliant les points forts des deux approches.
Enfin, nous proposons une nouvelle approche conçue pour faire collaborer plusieurs classifications topologiques entre elles, en préservant la confidentialité des données. Cette approche est basée sur un formalisme d'apprentissage collaboratif qui dérive ses règles d'adaptations à partir d'une nouvelle fonction de coût composée de deux termes : un terme de quantification et un autre de collaboration.

Abstract

This thesis is focused, on the one hand, to study clustering anlaysis approaches in an unsupervised topological learning, and in other hand, to the topological modular, hybrid and collaborative clustering. This study is adressed mainly on two problems:
1) cluster characterization using weighting and selection of relevant features, and the use of the memory concept during the unsupervised topological learning process; 2) and the problem of the ensemble clustering techniques: modular, the hybrid and collaborative approaches.
We are particularly interested in this thesis in Kohonen's self-organizing maps which have been widely used for unsupervised classification and visualization of multidimensional datasets. We offer several weighting approaches and a new strategy which consists in the introduction of a memory process into the competition phase by calculating a voting matrix at each learning iteration. Using a statistical test for selecting relevant features, we will respond to the problem of dimensionality reduction, and to the problem of clusters characterization.
For the second problem, we use the relational analysis approach (RA) to combine multiple topological clustering results. The benefits of RA will be combined with those of self-organizing maps in order to obtain a new algorithm called RTM (Relational Topological Map). Finally, we adapt the classical algorithm of self-organizing maps to enable the collaboration between them.