Contributions to collaborative clustering and its potential applications on very high resolution satellite images

Jérémie Sublime

NNT : 2016SACLA005

Thèse de doctorat de l'Université Paris-Saclay préparée à AgroParisTech

Ecole doctorale n°581
Agriculture Alimentation BIologie Environnement Santé
Spécialité de doctorat: Informatique appliquée

par

M. Jérémie Sublime

Contributions au clustering collaboratif et à ses potentielles applications en imagerie à très haute résolution

Thèse présentée et soutenue à Paris, le 9 Novembre 2016.

Composition du Jury :

M.	Michaël Aupetit	DR - HDR QCRI - Hamad Bin Khalifa University	(Rapporteur)
M.	Younès Bennani	PR Université Paris 13	(Co-directeur)
M.	Antoine Cornuéjols	PR AgroParisTech – Université Paris-Saclay	(Directeur)
Mme	Pascale Kuntz	PR Polytech Nantes – Université de Nantes	(Rapportrice)
M.	François Yvon	PR Université Paris Sud – Université Paris-Saclay	(Président)

UMR MIA-Paris, AgroParisTech, INRA, Université Paris–Saclay
CNRS UMR 7030, LIPN, Université Paris 13

Contents

List of Figures

0.1 Acknowledgements

I would like to express my gratitude to all the people that helped me during these last years of doctoral studies, as well as all these without whom this work may not have been possible:

First of all, I would like to thank my two thesis directors Professor Antoine Cornuéjols and Professor Younès Bennani. This work would not have been possible without their recommendations, their advises and insightful guidance, as well as their help to secure financial support. They were also very helpful when the time came to think about what I would do after my PhD thesis, and for that too I am so grateful.

Credit must also be given to all my colleagues and friends who also directly participated in the work achieved during this thesis and helped me co-writing several articles: Associate Professor Nistor Grozavu, Associate Professor Guénaël Cabanes and Associate Professor Basarab Matei. In particular, Associate Professor Nistor Grozavu and I spent a lot of time working together and traveling to several conferences.

I also would like to thank my family that supported me during the course of my undergraduate and graduate studies. Without them, I do not think any of it would have been possible. Beyond material and effective support, I am part of a family that counts several scholars and teachers among its ranks. And I believe that they have played a huge part in setting my course toward a PhD Thesis and landing a job in higher education.

Then, I also wish to extend my gratitude to all the other friends and colleagues with whom I have been spending time these last three years:

- Associate Professors Mustapha Lebbah and Hanene Azzag, as well as PhD student -soon to be Doctor- Ehab Hassan with whom I shared my office at the University Paris 13. They are awesome office mates and were very helpful and supportive whenever I needed it.

- The colleagues with whom I shared my second office at AgroParisTech: Doctor Rim Touari, Doctor Ghazal Jaber, PhD students -soon to be doctors too- Mathieu Bouyrie, Pierre-Alexandre Murena, Asma Dachraoui and Oumaima Alaoui Ismaili, as well as Joe Raad and Dylan Cohen respectively PhD student and former intern from the office next door. I think we tried pretty much all the restaurants around AgroParisTech together.

- Several other PhD students at the University Paris 13, with whom I spent a lot of time drinking tea or swimming: Mohammed Ghesmoune, Issam Falih, Ievgen Redko, Marcos de Melo da Silva, Moufida Rehab, Leila Abidi, Inès Bannour, Imad Kissami and the now Doctor Aïcha Ben Salem.

- The members of the Computer Science department of the Paris 13 University with whom I worked or that helped me when I was a teaching assistant at the Galilée Institute: Professor Adeline Nazarenko, Professor Christine Choppy, Professor Yann Chevaleyre, Professor Ken Chen, and Associate Professor Pierre Boudes.

- Other professors and administrative members of AgroParisTech and the ABIES doctoral School: Professor Alexandre Pery, Dean at the ABIES Doctoral School; Doctor Irina Vassileva Deputy Dean at the ABIES doctoral school; Professor Liliane Bel head of the MMIP department at AgroParisTech, Professor Juliette Dibie, Associate Professor Liliana Ibanescu, and invited Associate Professor Cristina Manfredotti.

- Other fellow researchers and academics whom I met during the multiple COCLICO meeting in Paris and Strasbourg, and sometimes met again during conferences: Professor Pierre Gançarski, Associate Professor Cedric Wemmert, Associate Professor Nicoleta Rogovschi, former Associate Professor and now Researcher at Google trying to build Skynet: Laurent Orseau, and PhD student Andrés Troya-Galvis.

I would also like to thank several administrative staff members from both AgroPariTech and the Paris 13 University, because nothing is ever possible without them: Francine Puel former administrative staff member for the INRA at AgroParisTech -and we miss her-, Nathalie Tavarès the secretary for our team at the LIPN in the Paris 13 University, and Irène Desecures secretary at the applied mathematics and computer science department of AgroParisTech.

I also have a special thought for Associate Professor Maria Malek of the EISTI Engineering school and for Associate Professor Rushed Kanawati of the Paris 13 University. It is thanks to them that I met with Professor Younès Bennani in the first place, and that I was ultimately able to enlist for a PhD thesis.

Finally, this PhD thesis was founded by a grant from the ANR project COCLICO ANR-12-MONU-0001, and paid via the INRA.

Chapter 1

Introduction

1.1 Abstract

This thesis presents several algorithms developed in the context of the ANR COCLICO project. The aim of this project is to create collaborative algorithms and methods with the purpose of analyzing satellites images. Thus, this work is organized along two main axis:

The first axis is concerned with introducing Markov Random Fields (MRF) based models to provide a semantic rich and suited algorithm to apply to images that have been preprocessed and are already segmented. This method is based on the Iterated Conditional Modes Algorithm (ICM algorithm) and can be applied to segments (also called regions, or super-pixels) of very high resolution (VHR) satellite pictures. Our proposed method provides some basic semantic information on the clusters and their relationship within the image.

The second axis deals with collaborative clustering methods developed with the goal of being applicable to as many clustering algorithms as possible, including the algorithms used in the first axis of this work. A key feature of the methods proposed in this thesis is that they can deal with either of the following two cases : 1) several clustering algorithms working together on the same data represented in different features spaces, 2) several clustering algorithms looking for similar clusters in different data sets having the same features. Clustering algorithms to which these methods are applicable include the ICM algorithm, the K-Means algorithm, density based algorithms such as DB-scan, all Expectation-Maximization based algorithms (EM) such as the Self-Organizing Maps (SOM) and the Generative Topographic Mapping (GTM) algorithms. Unlike previously introduced collaborative methods, our models have no restrictions in term of the types of algorithms that can interact together, do not require that all methods be looking for the same number of clusters, and are provided with solid mathematical foundations.

1.2 Thesis Scope

Clustering is the process of organizing objects into groups the members of which are similar. In the context of machine learning, it consists in partitioning a set of data (sometimes also called objects or observations) into subsets made of data that have similar characteristics. The main idea behind this task in *unsupervised learning* is to find structures in a collection of unlabeled data. The subsets built from this process are called clusters. A cluster is therefore by definition a collection of objects that are

"similar" to the other data from their own cluster, and "dissimilar" to the data belonging to other clusters.

Clustering is by definition a difficult task because it implies to find an usually unknown number of clusters based on a notion of "similarity" that lacks a universal definition. Until now, most clustering algorithms have been designed to operate on single data sets. However, with data sets getting bigger, more complex and being quite often distributed across several sites, it has become impossible for a single algorithm to tackle such kinds of data.

Satellite images are a good example of such data. Satellite images are made of several types of features: colors, shapes, neighborhood characteristics, and human made features. While all these different features are used to describe the same objects on the image, there is no single algorithm that can process such a wide range of features. To do so, it is necessary to have a different clustering algorithm processing each type of feature. Furthermore if we resume with the same example, once the structures and clusters have been found on a first satellite image, there is currently no easy way to re-use the already mined knowledge to fasten the clustering process on another image that shows similar characteristics, or better to process several images in parallel while exchanging the information in real time.

In this work, we discuss clustering techniques and methods that can be applied to such complex data sets. We first introduce clustering algorithms that can handle complex data sets such as satellites images where the data have neighborhood dependencies. And then we propose a generic framework to enable clustering algorithms to work together.

This leads us to the fundamental concept of *collaborative clustering*. The aim of collaborative clustering is to reveal the common underlying structures found by different algorithms while analyzing their own data. The fundamental concept of collaboration is that clustering algorithms operate locally but collaborate by exchanging information about the local structures found by each algorithm. This kind of collaborative learning can be beneficial to a wide number of tasks including multi-view clustering, clustering of distributed data, clustering of high-dimensional data, multi-expert clustering, multi-scale analysis and transfer learning. Collaborative clustering methods fall into two categories: *horizontal collaboration* and *vertical collaboration*. In the case of horizontal collaboration, several algorithms are working in different representations (or features spaces) of the same data objects. As for vertical collaboration it is concerned with the case of several clustering algorithms looking for similar clusters on different data sets that have the same features.

1.3 Thesis Overview

This thesis is structured into five chapters (Introduction, Conclusion and Appendices excluded) and is organized as follows:

Chapter 2, Clustering This chapter introduces the basis of Machine Learning with a particular focus on clustering techniques. We introduce the key notions of clustering, the potential applications of such techniques, and several examples of the most common clustering algorithms described in the literature and from which most of the current state of the art methods have been inspired. In particular, we give a detailed description of the K-Means algorithm and Expectation-Maximization algorithm (EM)

for the Gaussian Mixture Model (GMM) because we will use both algorithms a lot in the following chapters. This chapter can be seen as a state of the art in clustering, and highlights strengths, weaknesses and challenges that this field still faces.

Chapter 3, Clustering applied to image data sets This chapter starts with a brief introduction on image segmentation followed by a state of the art about how clustering techniques introduced in the previous chapter can be applied to image data sets. We highlight the differences between images data sets and regular data sets and the difficulties that arise from these differences. We then present Markov Random Field (MRF) based methods, which are hybrid clustering and segmentation methods commonly used in image segmentation and image clustering. This is followed by the presentation of one of our proposed method in which we modify the energy function the Iterated Conditional Modes (ICM) algorithm (an algorithm based on the MRF) in order to adapt it to the specific case of our very high resolution satellite images. This modification aims to cope with the highly irregular neighborhood dependencies present in such data sets while keeping a low computational complexity. In addition, we take advantage of our newly proposed energy function to add low level semantic information on our clusters. The chapter concludes on the experimental results from our proposed method.

Chapter 4, Collaborative Clustering We introduce here the notion of collaborative clustering, its applications, and how it can be used for the clustering of high resolution images data sets. This is followed by a state of the art on existing methods and a summary of their strengths and weaknesses. Then, after suggesting a larger context and a generic scheme to describe collaborative clustering methods, we explain the details of one of our proposed framework which solves several weaknesses of previous methods: 1) It allows algorithms from different types and families to collaborate together (the method includes both model-based and non model-based clustering algorithms). 2) It is more than a voting system as it keeps the inner model of each collaborators. After introducing the main principle of our proposed horizontal collaboration method and several of its variations, we study its convergence properties using an entropy based measure. We validate our proposed collaborative framework with numerous experiments using a decent number of data sets for several variations of our method.

Chapter 5, Optimizing the collaboration process We tackle the problem of weighting the collaborators to avoid cases negative collaborations. To do so, we propose to different strategies: A mathematical weighting criterion based on a KKT optimization of the likelihood of our proposed collaborative term, and a regression-based criterion based on empirical results when the quality of each collaborator is known. Both approaches are compared throughout a set of experiments that highlight the pros and cons of each of them.

Chapter 6, From Horizontal to Vertical collaborative clustering using the GTM architecture In this chapter, we introduce two unsupervised neural network algorithms: the self-organized maps (SOM) and the Generative Topographic Mapping algorithm (GTM). Then, we explain how the architecture of such neural networks can be used to turn the horizontal collaborative framework introduced in the previous

chapter into an horizontal collaborative one. This chapter features several experiments where we compare our results with these of other vertical frameworks.

A summary of this thesis in French is available in Appendix A.

1.4 Main Contributions

International Journals :

- J. Sublime, N. Grozavu, G. Cabanès, Y. Bennani, and A. Cornuéjols: *From Horizontal to Vertical Collaborative Clustering using Generative Topographic Maps*, International Journal of Hybrid Intelligent Systems, Volume 12-4, 2016.

International Conferences :

- J. Sublime, Y. Bennani, and A. Cornuéjols: *A New Energy Model for the Hidden Markov Random Fields*, 21th International Conference on Neural Information Processing, At Kuching, Sarawak, Malaysia, Volume: in Lecture Notes in Computer Science, LNCS Springer, Proc of ICONIP'14, 2014.
- J. Sublime, A. Troya-Galvis, Y. Bennani, P. Gançarski, and A. Cornuéjols: *Semantic Rich ICM Algorithm for VHR Satellite Images Segmentation*, International Association for Pattern Recognition (IAPR), International Conference on Machine Vision Applications, At Tokyo, Japan, 2015.
- J. Sublime, N. Grozavu, Y. Bennani, and A. Cornuéjols: *Collaborative Clustering with Heterogeneous Algorithms*, IEEE International Joint Conference on Neural Network (IEEE IJCNN), At Killarney Convention Center in Killarney, Ireland, 2015.
- J. Sublime, N. Grozavu, Y. Bennani, and A. Cornuéjols: *Vertical Collaborative Clustering using Generative Topographic Maps*, 7th International Conference on Soft Computing and Pattern Recognition, At Fukuoka, Japan, 2015.
- J. Sublime, Y. Bennani, and A. Cornuéjols: *Collaborative-based multi-scale clustering in very high resolution satellite Images*, The 23rd International Conference on Neural Information Processing (ICONIP 2016), Kyoto, Japan: in Lecture Notes in Computer Science, LNCS Springer, Proc of ICONIP'16, 2016.

National Conferences :

- J. Sublime, Y. Bennani, and A. Cornuéjols: *Un nouveau modèle d'énergie pour les champs aléatoires de Markov cachés*, SFC'14, Société Francophone de Classification, at Rabat, Morocco, 2014.
- J. Sublime, Y. Bennani, and A. Cornuéjols: *A Compactness-based Iterated Conditional Modes Algorithm For Very High Resolution Satellite Images Segmentation*, Extraction et Gestion des Connaissances 2015, Luxembourg, 2015.
- J. Sublime, N. Grozavu, G. Cabanes, Y. Bennani, and A. Cornuéjols, *Collaborative learning using topographic maps*, AAFD&SFC'16, at Marrakech, Morocco, 2016.

Chapter 2

Clustering

Contents

Machine Learning: Machine Learning is a subfield of Computer Science defined in 1959 by Arthur Samuel as "*a field of study that gives computer the ability to learn without being explicitly programmed*". Machine Learning evolved from pattern recognition, computational learning theory and artificial intelligence -all of which it is still closely related with- and overlaps with several other fields such as computational statistics or mathematical optimization. Machine Learning is a science that consists in building algorithms and methods that helps computer to learn from data, process data and make predictions on data. Such algorithms operate by building a model from the data they have access to and can then make decisions or predictions based on this model rather than simply following static program instructions.

With the omnipresence of numerous data within a large number of science fields, Machine Learning has become a common tool for Data Mining, model building and prediction in a large number of areas such as biology and medicine [135, 85, 32, 162, 155], mathematics [158], finance and business [22, 153, 23], physics [8], chemistry [133], marketing and so many others.

Machine learning tasks are usually divided into three categories:

- **Supervised Learning:** The computer program is presented with a set of input examples provided with their desired label (training set) from which it will have to build a model or learn some rules that maps the inputs to the right outputs. Once the model has been learned, the computer can apply it to new data for which the output labels are unknown. Tasks related to supervised learning include classification, regression and time series predictions.

- **Unsupervised Learning:** With no labels given, the computer program has to find interesting structures, patterns and groups in a set of data. Potential applications include clustering (that we will formally introduce in the next section), feature learning, regression and pattern recognition.

- **Reinforcement Learning:** Given a dynamic environment, a computer program must perform a certain task and will improve its behavior based on positive or negative rewards inputs decided according to its actions. The algorithm is never directly told how to find the right answer, but has to explore different possibilities based on the rewards it gets for each of its action.

Machine Learning is sometimes conflated with data mining. However, the most common opinion is that data mining is rather a specific field that focuses more on exploratory data analysis, and thus is restricted to the unsupervised aspects of Machine Learning. Machine learning would therefore be a broader field, used among other things for data mining purposes, but which contains a larger range of applications.

2.1 Introduction to clustering

Clustering -sometimes called *data segmentation*- is a machine learning task of exploratory data mining the aim of which is to split a data set made of several data (also called *objects*, *data objects*, or *observations*) into several subsets. Each object is described by several *attributes*, also called *features* that describe its properties. The subsets created by the process of clustering a data set are called *clusters*. Objects from a given cluster are supposed to be homogeneous and to share common characteristics. A very simple example of a data set with two attributes and three visually distinct clusters is shown in Figure 2.1.

There is a huge number of clustering methods that automatically create clusters, each with its own strategy and criteria. The main criterion to build clusters relies on the notion of *similarity* between two objects. Based on this concept of similarity, a clustering algorithm will have to make the decision to group several objects in the same cluster, or to separate them. In this context, the choice of the *similarity measure* is critical since it will ultimately determine the final clusters.

While this notion of similarity is a first step to define a clustering method, it is however not enough. Once an agreement as been found on which similarity measure will be used, the next step is to define a strategy to build the clusters using this similarity measure. Given the large number of similarity measures available, and considering that several strategies are usually available for each of them, it is no surprise that a huge variety of clustering methods is available in the specialized literature.

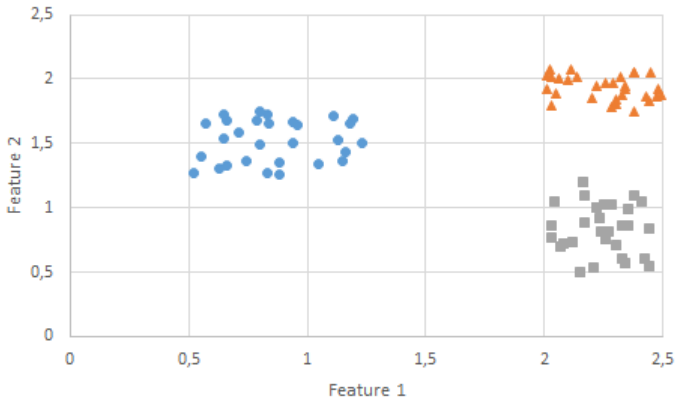


Figure 2.1: Example of a 2 dimensional data set with 3 visually distinct clusters.

Clustering methods have been used in a large array of applications such as pattern recognition, web mining, business intelligence, biology for taxonomy purposes, or security applications. Clustering can also be used for *outliers detection* [66, 112], where *outliers* (objects that are "far" from any cluster) may be more interesting than common cases.

In business intelligence, clustering can be used to sort a large number of customers into groups where customers share strong similar characteristics. In pattern recognition, clustering can be used to discover clusters in handwritten character recognition systems. Clustering also has many applications in Web mining. For example, a keyword search may often return a very large number of hits (i.e., pages relevant to the search) due to the extremely large number of web pages. Clustering can be used to organize the search results into groups and present the results in a more concise and easily accessible way. Moreover, clustering techniques have been developed to cluster documents into topics, which are commonly used in information retrieval practices. Applications of outliers detection include the detection of credit card fraud and the monitoring of criminal activities in electronic commerce. For example, exceptional cases in credit card transactions, such as very expensive and infrequent purchases, may be of interest as possible fraudulent activities [61].

In the forthcoming sections of this chapter, we will present several clustering methods based on different measures and strategy. In particular, we will focus on prototype-based methods that will be used in several chapters of this thesis, as well as on clustering algorithms that are specific to data sets acquired from satellite images.

2.2 Different types of clustering

Far from being an exhaustive state of the art, this section introduces some of the key concepts in clustering. We advise our readers that may want a more exhaustive state of the art on clustering methods, or to read more about the differences between existing methods to read one of the following documents [12, 152, 72].

Notations

- Let $X = \{x_1, ..., x_N\}$ be a data set of N objects. Since we will mostly consider quantitative data in this thesis, we have $x_n \in \mathbb{R}^d$;
- Let $C = \{c_1, ..., c_K\}$ be a clustering result with K clusters;
- Let $|c_i|$ be the number of objects belonging to the cluster c_i;
- Let $d(x_i, x_j)$ be the distance between two objects x_i and x_j;
- Let $D(c_a, c_b)$ be the distance between two clusters c_a and c_b

The result produced by a clustering algorithm may have different representation depending on whether or not it is possible for an object to belong to several clusters.

(a) Hard clustering result

	c_1	c_2	c_3
x_1	1	0	0
x_2	0	1	0
x_3	0	0	1
x_4	0	0	1

(b) Soft clustering result

	c_1	c_2	c_3
x_1	1	1	0
x_2	0	1	1
x_3	0	0	1
x_4	0	0	1

(c) Fuzzy clustering result

	c_1	c_2	c_3
x_1	0.9	0.1	0
x_2	0	0.8	0.2
x_3	0	0.3	0.7
x_4	0	0	1.0

Table 2.1: Example of several objects' degree of belonging to three clusters: hard clustering, soft clustering and fuzzy clustering

So far we have implicitly considered that each object may only belong to a single cluster. This is the easiest case and by far the most common in the literature. The type of clustering is called *hard clustering*: the data X are sorted into in K clusters $C = \{c_1, ..., c_K\}$ that are subsets of X so that $\bigcup_{k=1}^{K} = X$.

This type of result is the most common and the easiest to interpret for humans. However, it is sometimes necessary to give some more flexibility to the clusters. First there is the case of the outliers that we already mentioned, were some objects may be too different from all other objects and thus may not find any suitable cluster. This is called *partial clustering*. And then there is the case of objects that have characteristics that could match several clusters. For this later case when objects are at the border of several cluster, *soft clustering* allows one object to belong to more than one cluster.

Soft clustering however may sometimes be difficult to interpret from an expert point of view. Furthermore, it does not reflect if an object may belong more to a cluster than another and to which degree. *Fuzzy clustering* solves this problem by providing for each object its degree of belonging to each cluster [14]. While it is not possible to go back to hard clustering from a soft clustering result, it is possible to transform a fuzzy clustering result into a hard one just by linking each object to the cluster with which it has the highest degree of belonging (likewise, it is possible to go from fuzzy to soft clustering using a threshold). Examples of the three types of clustering (hard, soft and fuzzy) are shown in Table 2.2.

2.3 Distance-based clustering algorithms

The vast majority of clustering algorithms are based on the notion of distance between the data as a similarity (or dissimilarity criterion). Within this context, clustering algorithms often try to optimize an objective function which favors clusters that are both compact and well separated. For these algorithms, the choice of the distance function is key. Examples of common distances are shown in Table 2.2.

Euclidian distance	$\|\|a-b\|\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
Squared Euclidian distance	$\|\|a-b\|\|_2^2 = \sum_i (a_i - b_i)^2$
Manhattan distance	$\|\|a-b\|\|_1 = \sum_i \|a_i - b_i\|$
Maximum distance	$\|\|a-b\|\|_\infty = \max_i \|a_i - b_i\|$
Mahalanobis distance	$\sqrt{(a-b)^\top S^{-1}(a-b)}$ where S is the covariance matrix
Hamming distance	$Hamming(a,b) = \sum_i (1 - \delta_{a_i,b_i})$

Table 2.2: Examples of common distances

The following subsections show examples of distance-based clustering algorithms.

2.3.1 Density-based clustering methods

Density-based clustering methods [115, 95] consider the most basic and perhaps the most visual definition of a cluster: a cluster is an area of space with a high density of data and is separated from other clusters by low density regions. This notion of density relies on the concept of object neighborhood. By object neighborhood, we mean other objects that are located at a certain distance of the observed object. For density-based clustering methods, the higher the number of neighbors in an object's vicinity, the more chances that this object belongs to a high density region, and thus is part of a cluster formed with its neighbors. Unlike many other clustering methods, density-based algorithms do not assume that the clusters should have specific shapes and can easily detect concave clusters [62] if the parameters are well tuned.

The parameters for this type of clustering algorithms generally include a distance threshold to determine what should be considered a given object's neighborhood: let $V_\epsilon(x)$ be the neighborhood of an object x so that $V_\epsilon(x) = \{y \in X | d(x,y) \leq \epsilon\}$, with ϵ a threshold and $d(x,y)$ a distance between x and y. Examples of such density-based method include the DBSCAN algorithm (Density-Based Spatial Clustering of Applications with Noise) [41, 141, 116], or the OPTICS algorithm (Ordering points to identify the clustering structure) [2] which adds a second threshold determining the minimum number of objects that must be in a neighborhood for the said neighborhood to be considered dense.

In Algorithm 1, we give the pseudo-code for the DBSCAN algorithm.

Algorithm 1: DBSCAN Algorithm

```
Function DBSCAN(X,ϵ,MinPts)
    C=0
    forall x_n ∈ X do
        if x_n has not been visited then
            mark x_n as visited
        end
        V_n= regionQuery(x_n,ϵ)
        if sizeof(V_n)≥MinPts then
            C++
            expandCluster(x_n,V_n,C,ϵ,MinPts)
        else
            mark x_n as noise
        end
    end
Function expandCluster(x_n,V_n,C,ϵ,MinPts)
    Add x_n to cluster C
    forall x_i ∈ V_n do
        if x_i has not been visited then
            Mark x_i as visited
            V_i = regionQuery(x_i,ϵ)
            if sizeof(V_i)>MinPts then
                V_n+ = V_i
            end
        end
        if x_i does not belong to any cluster yet then
            Add x_i to cluster C
        end
    end
Function regionQuery(x_n,ϵ)
    list = ∅
    forall x_i ∈ X do
        if i ≠ n and d(x_n, x_i) ≤ ϵ then
            list+={x_i}
        end
    end
    return list
```

2.3.2 Hierarchical clustering methods

The vast majority of clustering algorithms is building results that come in the form of a flat separated clusters where the clusters are all independent and no structure exists between them. However, another approach consists in trying to have results in the form of clusters between which there is a hierarchical structure. The most common structure is to build clusters as trees, very similar to phylogenetic trees in biology: at the top of the tree is a single cluster containing all the objects. This cluster will then be split into several sub-clusters that will also be split into other clusters and so on. The clusters close from the root of the tree will be crude and will contain a lot of objects

that may still be relatively dissimilar, and the clusters far from the root will contain less but more similar objects [72, 144, 134, 159, 37].

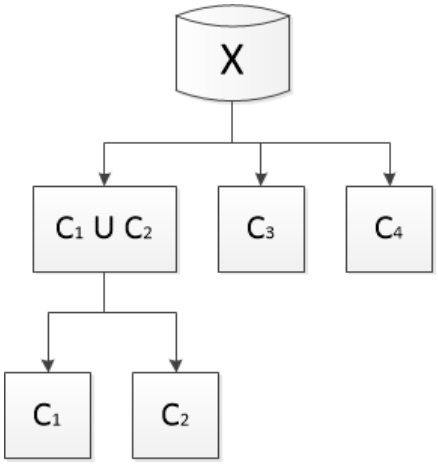


Figure 2.2: Example of a hierarchical result.

This type of clustering where the solution is given in the form of a tree (or dendrogram) is called *hierarchical clustering*. In this case, each data object belongs to a single leaf cluster, but as a consequence it also belongs to all the father nodes up to the root of the tree. Hierarchical clustering is further divided into two sub-categories: agglomerative approaches (or "bottom-up" approaches) and divisive approaches (or "top-down" approaches). In agglomerative methods, the clustering algorithm starts with all objects belonging to a different leaf and then regroup them until there is a single cluster containing all the objects. Divisive approaches on the contrary start with all the data in the same cluster, and this cluster is then divided into sub-clusters in a recursive manner. In Figure 2.2, we show an example of a hierarchical result with 4 clusters.

There are many different methods to create a tree of hierarchical clusters. In Algorithm 2, we show the main framework followed by agglomerative methods.

The main difference between the various algorithms proposed in the literature lies on the choice of a similarity measure to merge (or split) the clusters. The main measures for hierarchical clustering are listed bellow:

- *Single-linkage*: Assesses the minimal distance that exists between data belonging to two different clusters. This linkage is very popular because it can be used to detect clusters of any shapes and that may not be hyper-spherical. However, it is noise-sensitive and cannot detect clusters that are in direct contact.

$$D_s(c_1, c_2) = \min_{x \in c_1, y \in c_2} d(x, y) \tag{2.1}$$

- *Complete-linkage*: Assess the maximal distance that exists between data belonging to two different clusters. It is highly noise-sensitive and rarely used in practice.

$$D_c(c_1, c_2) = \max_{x \in c_1, y \in c_2} d(x, y) \tag{2.2}$$

- *Average-linkage*: It considers the average distance between the data belonging to two different clusters. It is less noise-sensitive than the two previous links. But it tends to favor hyper-spherical clusters.

$$D_a(c_1, c_2) = \frac{1}{|c_1||c_2|} \sum_{x \in c_1} \sum_{y \in c_2} d(x, y) \tag{2.3}$$

- *Centroid-linkage*: It assesses the distance between the mean values of two clusters. This linkage is not noise-sensitive but also tends to favor hyper-spherical clusters. One possible variation of this linkage is the Ward Criterion [144] where the mean-values are weighted depending on the number of elements in the cluster.

$$D_{\mu}(c_1, c_2) = ||\mu_1 - \mu_2|| \tag{2.4}$$

Algorithm 2: Hierarchical clustering algorithm: general framework (agglomerative)

```
Create a cluster for each element
Initialize the dendrogram's leaves
while There is more that one cluster left do
    Compute all the pairwise similarities between the clusters
    Merge the two clusters that are the most similar
    Update the Dendrogram
end
Cut the dendrogram depending on the desired number of clusters
```

The CURE algorithm [58] uses an alternative linkage that enables detecting clusters of any shape while remaining resistant to noise. To do so, a few elements are selected in each cluster. These elements are chosen by first picking the farthest element from the cluster centroid, and then the farthest element from the previously picked one, and so on until c elements per cluster have been picked. These representative are artificially modified and moved closer to the cluster centroid. Finally the single-linkage criterion is used as a merging criterion.

Hierarchical clustering has two main limitations: First, once the clusters dendrogram is built, one still needs to decide where to cut to get the final clusters. This choice remains a difficult one despite a plethora of available methods in the literature (see [72]). Second, these methods have a high computational complexity of at least N^2 for a data set of size N, which makes them difficult to use with large data sets.

2.3.3 Prototype-based algorithms

Prototype-based algorithms are another family of clustering algorithms. The principle of these algorithms is based on *vector quantization*, a data compression process which consists in representing the data with a few representatives called *prototypes*. Each data will then be linked to its closest prototype in the data space. The main task of these algorithms is therefore to build relevant prototypes and link the data to them.

A common example of prototype would be a centroid of a high density area. Depending on the number of prototypes, each of them may represent a cluster, or several of them may need to be regrouped to find the clusters.

2.3.3.1 The K-Means algorithm

The K-Means algorithm is one of the most famous prototype-based clustering algorithm. It is a simple and fast, yet relatively good clustering method. Its principle is the following [96, 122]: Suppose that we would like to divide our data into K clusters, the value of K being known in advance. We allocate K cluster prototypes (also called

mean-values) to our input space, and we would like to move these prototypes so that each of them will become the centroid of a cluster. Given that we have chosen a distance measure, the procedure to do so consists in alternating the following two steps until convergence: 1) Link each data to the closest prototype, 2) Move the prototype so that is becomes the barycenter of the current data to which it is linked. This procedure is described in Algorithm 3 and an example with 2 clusters on the "*Old Faithful*" data set is shown in Figure 2.3.

It is convenient at this point to give a notation that describes the assignment of data points to clusters. For each data point x_n, let $s_{n,i} \in \{0, 1\}$ be a set of binary indicator variables with $i \in [1..K]$. The $s_{n,i}$ are used to describe to which one of the K clusters a data has been assigned. For instance, if x_n is assigned to cluster c_k, then $s_{n,k} = 1$ and $\forall i \neq k, s_{n,i} = 0$. Ultimately, what the K-Means algorithm does is to optimize a cost function $\tilde{R}(\mu)$ as given in Equation (2.5).

$$\tilde{R}(\mu) = \sum_{n=1}^{N} \sum_{i=1}^{K} s_{n,i} ||x_n - \mu_i||^2 \tag{2.5}$$

Because each phase reduces the value of the objective function $\tilde{R}(\mu)$, convergence of the algorithm is assured. However, it may converge to a local rather than global minimum of $\tilde{R}(\mu)$. The convergence properties of the K-means algorithm have been studied in [96].

Algorithm 3: K-Means Algorithm

Choose a value for K
Randomly initialize the K centroids μ_i
while Learning **do**
  forall $x_n \in X$ **do**
    Assign x_n to the cluster c_i with the closest centroid:
    $s_{n,i} = \begin{cases} 1 & \text{if } i = argmin_i ||x_n - \mu_i||^2 \\ 0 & \text{otherwise} \end{cases}$
  end
  Minimize Equation (2.5):
  forall μ_i **do**
    $\mu_i = \frac{\sum_n x_n \cdot s_{n,i}}{|c_i|}$
  end
end

Several algorithms based on improved or modified versions of the K-Means algorithm have been proposed in the literature [41, 46, 136, 64, 77, 34, 29]. Algorithms based on the K-Means algorithm suffer from several weaknesses: The main one is the need to provide a K. It requires to know in advance how many clusters are to be found. In practice this is rarely the case because we expect the clustering algorithm to actually discover the clusters. Therefore, the only solution when the number of clusters is really unknown is to run the algorithm several times with different values of K and to pick the best clustering based of a given quality index (for instance the *Silhouette index* [113] or the *Davies-Bouldin* index [36]). This method is costly and may prove ineffective because of the non-deterministic nature of the K-Means algorithm. Adaptations of the K-Means algorithm have been proposed [106] to solve this issue, but they remain only partially satisfying. Second, algorithms based on the K-Means can only

find hyper-spherical clusters and will also fail to detect the clusters properly if their sizes are significantly different.

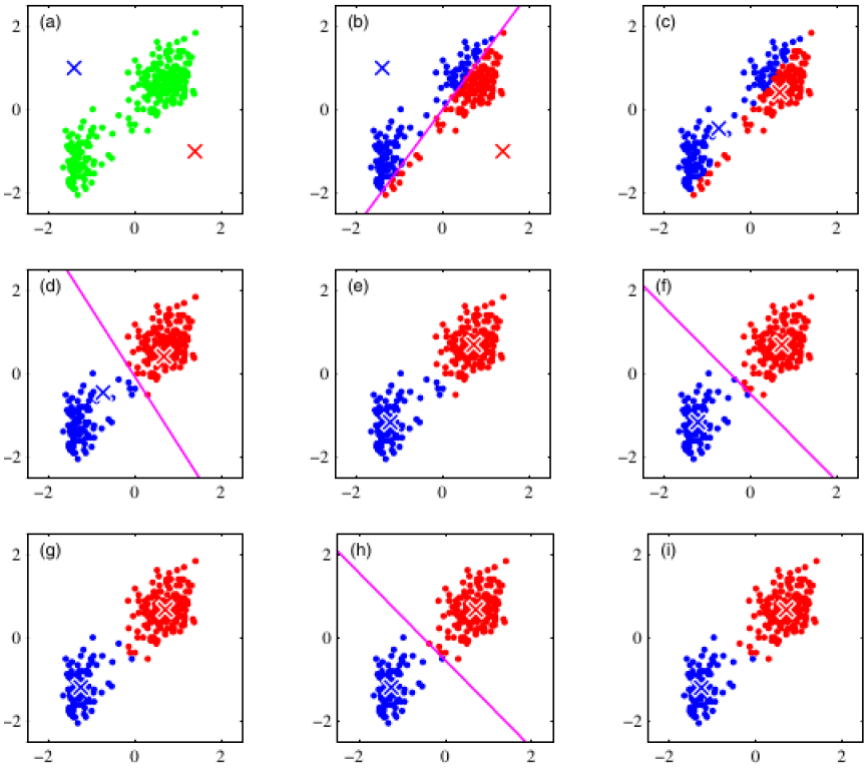


Figure 2.3: Illustration of the K-Means algorithm on the Old Faithful data set

2.3.3.2 Fuzzy C-Means

Since fuzzy clustering allows the objects to belong to several clusters simultaneously with different degrees of membership, in many situations it is more natural than hard clustering. Within this context, the Fuzzy C-Means algorithm [14, 91] is a major extension of the K-Means algorithm that enables fuzzy clusters.

A generalization of the hard partition from the K-Means to the fuzzy case is easily possible by allowing the $s_{n,i}$ to attain real values: $s_{n,i} \in [0,1]$ with the properties shown in Equation (2.6). A partition can then be represented by the partition matrix $S = (s_{n,i})_{C \times N}$ where N is the size of the data set and C the number of clusters (equivalent to the K in K-Means).

$$s_{n,i} \in [0,1], \quad \forall n \sum_{i=1}^{C} s_{n,i} = 1 \tag{2.6}$$

The *objective function* optimized by the fuzzy C-Means algorithm is given in Equation (2.7) where m is a fuzziness parameter (sometimes called a *fuzzifier*), which determines the fuzziness of the resulting clusters. For $m = 1$ the resulting partition will be hard, and when $m \to \infty$ the partition will be completely fuzzy ($s_{n,i} \to \frac{1}{C}$). Usually when the fuzzy C-Means relies on an Euclidian distance, $m = 2$ is the most common choice.

$$\tilde{R}(\mu) = \sum_{n=1}^{N} \sum_{i=1}^{C} (s_{n,i})^m ||x_n - \mu_i||^2 \tag{2.7}$$

In Algorithm 4 bellow, we show the Fuzzy C-Mean algorithm. As one can see, it is fairly similar with the K-Means algorithm.

Algorithm 4: Fuzzy C-Means Algorithm

Choose a value for C and for m
Randomly initialize the $s^0_{n,i}$ so that the conditions in (2.6) are respected
$r = 0$
do
  Minimize Equation (2.7):
  forall μ_i **do**
    $\mu_i = \frac{\sum_n x_n \cdot (s^{r-1}_{n,i})^m}{\sum_{n=1}^{N} (s^{r-1}_{n,i})^m}$
  end
  Update de partition matrix S:
  for $n \in [1..N]$ **do**
    for $i \in [1..C]$ **do**
      $s^r_{n,i} = \frac{1}{\sum_{j=1}^{C} \left(\frac{||x_n - \mu_i||^2}{||x_n - \mu_j||^2} \right)^{\frac{2}{m-1}}}$
    end
  end
  $r++$
while $||S^r - S^{r-1}|| > \epsilon$;

2.3.3.3 Affinity propagation clustering

The affinity propagation algorithm (AP) [48] is another recent and popular clustering algorithm that relies on representing the clusters in the form of prototypes. The main idea of this method is to use a data similarity graph (containing pairwise similarities between all the data) with the goal of determining which data elements can best be used to represent the other data locally.

Let $X = \{x_1, ..., x_N\}$ be a data set, and $s(x_i, x_j)$ a similarity measure. Let $R = (r_{i,k})_{N \times N}$ be a responsibility matrix where each $r_{i,k}$ quantifies how well-suited x_k is to be a good representative for x_i relatively to the other candidates. And let $A = (a_{i,k})_{N \times N}$ be the availability matrix that quantifies how appropriate it would be for x_i to pick x_k as a representative when taking into account the other points preferences to also pick x_k as their representative. The algorithm then works as shown in Algorithm 5: First it builds the matrices A and R. Then it extract the most accurate representative for each data, and pick a few of them to become prototypes based on which data are their own best representative. Finally, each data is linked to one of the final prototype.

The main advantage of the Affinity propagation algorithm is that unlike the K-Means algorithm or the Fuzzy C-Means algorithms, the number of clusters need not be provided. The algorithm will find by itself a certain number of prototypes based on the initial similarity graph.

One drawback of this algorithm is its high N^2 complexity and the need to use matrices of the same size. These requirements make the Affinity propagation algorithm unsuitable for large size data sets.

Algorithm 5: Affinity Propagation Algorithm

Build the similarity graph S
Initialize A and R to all zeroes
while the algorithm did not converge **do**
  Update R:
  $r_{i,k} = s_{i,k} - \max_{k' \neq k}(a_{i,k'} + s_{i,k'})$
  Update A:
  $\forall i \neq k, a_{i,k} = min\left(0, r_{k,k} + \sum_{i' \notin i,k} max(0, r_{i',k})\right)$
  $r_{k,k} = \sum_{i' \neq k} max(0, r_{i',k})$
end
Initialize the prototypes: $C = \emptyset$
Find the prototypes:
forall $x_i \in X$ **do**
  Find $k^* = \text{argmax}_k(a_{i,k} + r_{i,k})$
  if $k^* == i$ **then**
    $C+ = \{k^*\}$
  end
end
Link each data to a prototype:
forall $x_i \in X$ **do**
  Find $c = \text{argmax}_{k \in C}(a_{i,k} + r_{i,k})$
  Link x_i to x_c as a prototype
end

2.4 Spectral methods

Spectral methods are recent clustering techniques that have had a lot of success recently. The main idea of spectral clustering is to see clustering as a graph partitioning problem. Spectral clustering considers the adjacency matrix of the graph representing the data, and the eigenvalues of this matrix. This type of clustering is called "spectral" because it uses the spectrum (eigenvalues) of the data set similarity matrix. Since these methods use a similarity matrix between the different objects, without using proper kernel matrices, they are actually quickly limited when the number of objects becomes relatively big.

Example of spectral techniques include the *Normalized Cuts* algorithm (also called Shi-Malik algorithm) [120]. This algorithms split the data into two subsets (X_1, X_2) based on the eigenvectors matching the first and second smallest eigenvalue of the similarity matrix Laplacian. The algorithm then uses a recursive hierarchical approach to create clusters based on the eigenvectors values and a chosen threshold.

The Meila-Shi algorithm [97] is another example of spectral method. For k given, it considers the eigenvectors matching the k highest matrix eigenvalues. Then it uses a regular clustering algorithm (such as the K-Means algorithm) to regroup the data based on their respective components in the eigenvectors.

2.5 Probabilistic clustering methods

Probabilistic clustering methods (sometimes called probabilistic model-based methods, or generative models) are algorithms the main hypothesis of which is that the data

are following a given probability density function. The goal of such algorithms is to estimate the parameters of these density functions and to define a mixture model to represent the different clusters. Many clustering techniques can be depicted in this model, e.g. fuzzy C-Means, Gaussian mixtures models (GMM), mixtures of Bernouilli distributions, etc. These methods make the hypothesis that each cluster c_i is linked to probability density function $p(X, \theta_i)$, where θ_i contains the parameters of the function. These laws can then be used to assess the probability of a data x_n to belong to a cluster c_i, thus generating a fuzzy partition. If we note π_i the proportion of the i^{th} component in the mixture model, then the parameters of the model are: $\Theta = \{(\pi_1, \theta_1), ..., (\pi_K, \theta_K)\}$, and the global density function is the following:

$$p(X, \Theta) = \sum_{i=1}^{K} \pi_i p(X, \theta_i) \tag{2.8}$$

This type of model is called a *generative model*, because once the parameters are known, it is possible to re-create the data just from the probability density functions and the mixing coefficients.

2.5.1 The EM Algorithm

The Expectation-Maximization (EM) algorithm [38] is an iterative method used to find the *maximum likelihood* or the *maximum a posteriori* (MAP) estimate of parameters in probabilistic and statistical models. As such it can be seen as an alternative to gradient-descent/ascent methods [35, 74] to find the optimal parameters of a given function.

When applied to clustering, the EM algorithm is used to find the clusters parameters. Classically, in probabilistic clustering, one is interested in estimating the parameters of the probabilistic model entertained by the algorithm when observing the data set X. This is generally done through a *maximum likelihood* estimation:

$$\Theta_{ML} \;=\; \underset{\Theta}{\text{Argmax}}\Big\{\log p(X|\Theta)\Big\} \tag{2.9}$$

the best estimates being the one that maximizes the probability of generating the observed data set.

If, in addition, there is a priori information $p(\Theta)$ about the parameters, this leads to the *maximum a posteriori* (MAP) estimation:

$$\Theta_{MAP} \;=\; \underset{\theta}{\text{Argmax}}\Big\{\log p(X|\Theta) \;+\; \log p(\Theta)\Big\} \tag{2.10}$$

Assuming that the observations in X are independent from each other, the estimation problem becomes:

$$\Theta_{MAP} \;=\; \underset{\Theta}{\text{Argmax}}\left\{\log\left[\sum_{n=1}^{N} P(x_n|\Theta)\right] \;+\; \log p(\Theta)\right\} \tag{2.11}$$

The EM algorithm starts with a guess about the parameters θ_k of the components as well as with the mixing probabilities π_k. Then, it alternates between two updating steps:

- In the *expectation* step (E-step), the current parameter values $(\theta_k, \pi_k)_{k=1,...,K}$ are used to compute the posterior probabilities of the clusters (latent variables) for each data element x_n.

- In the *Maximization* step (M-step), the log-likelihood is maximized using the updated responsibilities $s_n(k)$ (where $s_n(k)$ is the degree to which the data x_n belongs to the cluster c_k), leading to updated *a posteriori* estimations about the parameters and the mixing probabilities.

It is proved that the algorithm converges towards a local maximum of the log-likelihood. In practice, convergence is said to happen when the change in the log-likelihood falls below some threshold [151]. Graphically, this can be depicted as a *run* made of a multi-step process:

$$\Theta_0, S_0 \xrightarrow{M} (\Theta_1 \xrightarrow{E} S_1) \xrightarrow{M} \ldots \xrightarrow{M} (\Theta_t \xrightarrow{E} S_t) \xrightarrow{M} \ldots (\Theta_{t^*} \xrightarrow{E} S_{t^*})$$

where the successive partitions S_t can be soft, hard or fuzzy partitions.

2.5.1.1 The EM algorithm for the Gaussian Mixture Model

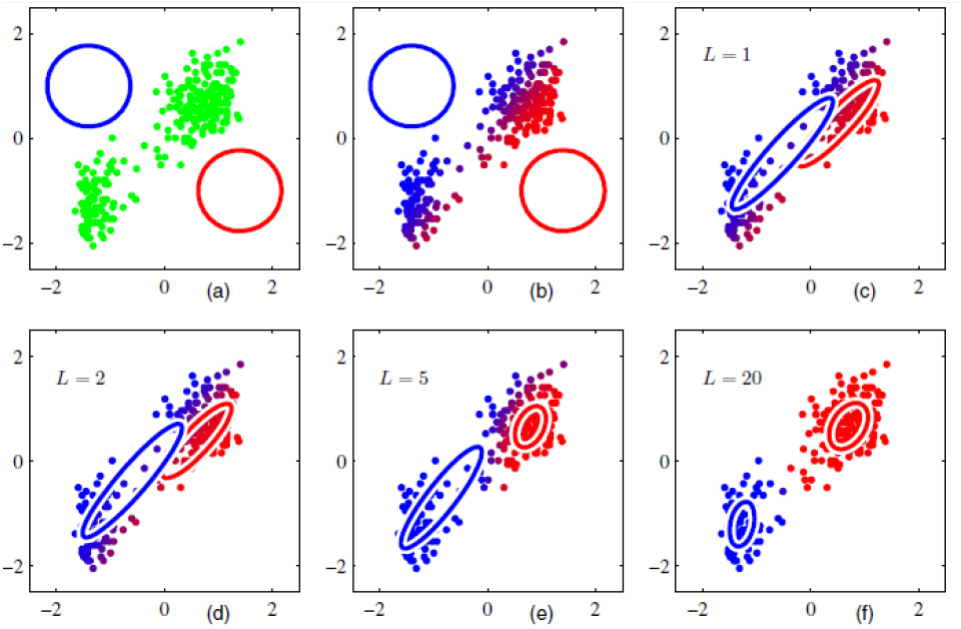


Figure 2.4: Illustration of the EM algorithm (GMM) on the Old Faithful data set

When it comes to applications of the EM algorithm for clustering, most of the time the model used will be the Gaussian Mixture Model (GMM), where each cluster c_k follows a distribution given by a mean value μ_k and covariance matrix Σ_k in addition to a mixing probability π_k. This version of the EM algorithm with the Gaussian Mixture model can been seen as a generalized version of the K-Means algorithm where clusters can have different sizes and can take an elliptical shape instead of being limited to a

spherical one. The full EM algorithm for the GMM pseudo-code is shown in Algorithm 6. In Figure 2.4 shows an example of such application on the Old Faithful data set.

Algorithm 6: EM Algorithm for the GMM

Initialize Θ randomly
while Learning **do**
  E-Step: evaluate $S = argmax_S p(S|X, \Theta)$
  forall $x_n \in X$ **do**
    $s_n(k) = \frac{1}{Z}\pi_k \mathcal{N}(x_n, \mu_k, \Sigma_k)$
  end
  M-Step: Re-evaluate Θ
  forall $k \in [1..K]$ **do**
    $N_k = \sum_{n=1}^{N} s_n(k)$
    $\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} s_n(k) \cdot x_n$
    $\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} s_n(k) \cdot (x_n - \mu_k)(x_n - \mu_k)^T$
    $\pi_k = \frac{N_k}{N}$
  end
end

2.6 Evaluating clustering results

Evaluating the quality of clustering results is a difficult task that has been an active research area for years, with new methods being proposed on a regular basis. The main difficulty with the evaluation of clustering results lies in the inherent unsupervised nature of clustering itself and the lack of consensus about what a *"good clustering"* should be. In this context, evaluating a clustering result is always more or less subjective, with each evaluation criterion favoring one concept of a *good clustering* (shape, compactness separation, etc.) over the others. Therefore the notions of *good clustering* and *best clustering* will depend on both the evaluation criterion and the clustering algorithm, with some evaluation criterion favoring some algorithms over others.

Still, despite this assumed relative subjectivity, there are a wide range of evaluation criterion that are commonly used in machine learning to assess and compare clustering results. There are several taxonomies available in the literature for these evaluation criteria [59, 71, 132], most of them defining 3 distinct groups:

- *Unsupervised indexes*, also called internal indexes: they only use internal information from the data as well as the clusters' characteristics.
- *Supervised indexes*, also called external indexes: they assess the degree of similarity between a clustering solution and a known partition of the data set (sometimes called a ground truth).
- *Relative indexes*: they are a separate class of criteria that make it possible to compare several clustering results of the same algorithm. Relative indexes simply use both external and internal criteria to choose the best solution among several proposed partitions.

2.6.1 Unsupervised indexes

Unsupervised evaluation criteria [98] are based on internal information from both the data and the clusters. For instance, several of them are based on the distance between

the data and the cluster centroids. These indexes have been introduced based on the simplest principles of what defines a cluster: (1) Objects from a given cluster are supposed to be as close as possible from each other. (2) Objects belonging to different clusters should be well separated and as far as possible. To assess these intuitive criteria, most indexes adopt a strategy that consists in measuring the distance between each data elements and some object representing the clusters (centroids, representative data elements, etc.). By doing so, it is quite straightforward to evaluate the compactness and separability of the clusters.

However, with no clear definition of what a "good cluster" is, each unsupervised index has its own way of computing the compactness and separability of the clusters and to use these two values to compute a final quality criterion.

Some of these criteria can be used as objective functions. The goal of a clustering algorithm would then be to find a solution that maximizes the said objective function. Some criteria however are too costly to be used in an objective function and are usually only computed when the clustering process is done.

Mean Squared Error (MSE) :

The mean squared error is one of the easiest way to evaluate the quality of a result for clustering algorithms that use centroids. Given a clustering solution S with K clusters, it can be computed as shown in Equation (2.12) where $d(\cdot)$ is a distance function, $|c_i|$ is the number of elements linked to the cluster c_i and μ_k the centroid of a cluster c_k.

$$MSE = \frac{1}{\sum_{i=1}^{K} |c_i|} \sum_{k=1}^{K} \sum_{x \in c_k} d(x - \mu_k)^2 \tag{2.12}$$

For a clustering result to be considered good, the Mean Squared Error must be as low as possible.

Dunn Index (DI) :

The Dunn index [40] is another internal criterion defined as in Equation (2.13) where $D(c_i, c_j)$ is a distance metric between two clusters c_i and c_j, and Δ_i is a measure of scatter for a cluster c_i. Any quality index using such a kind of ratio is called a "Dunn-like index".

A higher Dunn index indicates a better clustering.

$$DU = \frac{\min_{i \neq j} D(c_i, c_j)}{\max_{i \in [1..K]} \Delta_i} \tag{2.13}$$

One particularity of the Dunn Index is that the distances $D(c_i, c_j)$ and Δ_i can be defined in many different ways:

- $D(c_i, c_j)$ can be the smallest distance between two objects belonging to c_i and c_j respectively, see Equation (2.1). In this case Δ_i must be the largest distance between two objects belonging to a cluster c_i:

$$\Delta_i = \max_{x,y \in c_i} d(x, y) \tag{2.14}$$

- $D(c_i, c_j)$ can be the largest distance between two objects belonging to c_i and c_j respectively, see Equation (2.2). This is quite uncommon and give rather

poor results. In this case Δ_i must be the smallest distance between two objects belonging to a cluster c_i:

$$\Delta_i = \min_{x,y \in c_i} d(x, y) \tag{2.15}$$

- $D(c_i, c_j)$ can also be the distance between the centroids of c_i and c_j, see Equation (2.4). Then Δ_i usually is the largest distance between an object belonging to c_i and its centroid μ_i:

$$\Delta_i = \frac{1}{|c_i|} \sum_{x \in c_i} (x - \mu_i) \tag{2.16}$$

- Finally, $D(c_i, c_j)$ can be the average distance between the data belonging to c_i and c_j respectively, see Equation (2.3). Then Δ_i usually is the mean distance between all pairs of a cluster c_i:

$$\Delta_i = \frac{1}{|c_i| \cdot (|c_i| - 1)} \sum_{\substack{x,y \in c_i \\ x \neq y}} d(x, y) \tag{2.17}$$

Davies-Bouldin Index (DB) :

The Davies-Bouldin Index [36] is a possible alternative to the Dunn Index. It assesses whether the clusters are compact and well separated. It is based on the same possible measure of separation $D(c_i, c_j)$ and measure of scatter Δ_i than the Dunn Index. Given a clustering solution that contains K clusters, the Davies-Bouldin Index is defined as shown in Equation (2.18).

The Davies-Bouldin index is not normalized and a lower value indicates a better quality.

$$DB = \frac{1}{K} \sum_{i=1}^{K} \max_{j \neq i} \frac{\Delta_i + \Delta_j}{D(c_i, c_j)} \tag{2.18}$$

The most commonly used measure of separation and measure of scatter for the Davies-Bouldin Index -as introduced in the original article [36]- are based on the cluster mean values using Equations (2.4) and (2.16).

Silhouette Index (SC) :

The Silhouette index [112] is yet another internal criterion that assesses the compactness of the clusters and whether or not they are well separated. The main difference between the Silhouette index and the Dunn index or the Davies-Bouldin index is the following: the Silhouette Index can be computed for a given object x, a given cluster c_i, or for the whole clustering C.

For a data element, the Silhouette index is defined as shown in Equation (2.19) where a_x is the mean distance between the observed object x and all other objects that belong to the same cluster that x, and b_x is the mean distance between x and all other objects that are not in the same cluster that x.

$$SC(x) = \frac{b_x - a_x}{max(a_x, b_x)} \tag{2.19}$$

The Silhouette index takes values between -1 and 1. A positive value ($a_x < b_x$) means that x is closer from the objects belonging to its clusters than from the objects belonging to other clusters. Therefore a positive value close to 1 means that x is probably in the right cluster, while a negative one means that x should be in another cluster.

The Silhouette index for a given cluster c_i is the mean value of the Silhouette index computed on all the objects of this cluster:

$$SC(c_i) = \frac{1}{|c_i|} \sum_{x \in c_i} SC(x) \tag{2.20}$$

A Silhouette index that is positive and close to 1 means that the observed cluster is both compact and well separated from the other clusters.

Finally, the Silhouette index can be computed on the whole partition as shown in Equation (2.21). Once again, 1 is the best value meaning that the clusters are very compact and well separated and -1 is the worst value. Usually, the Silhouette Index must be positive for a clustering result to be considered acceptable.

$$SC(C) = \frac{1}{K} \sum_{i=1}^{K} SC(c_i) \tag{2.21}$$

Before considering to use the Silhouette index, one must consider the high computational cost of this index with large data sets: since it is not based on the mean vectors, the Silhouette index requires to compute several time the pairwise distances between all the data.

Wemmert-Gancarski Index (WG) :

The Wemmert-Gancarski index [147] is another elegant way to assess the quality of a clustering result based on the compactness and separability of the clusters. For a cluster c_i, it is computed as follows with $j = \text{argmin}_{k \neq i}\, d(x, \mu_k)$:

$$WG(c_i) = \begin{cases} 0 \text{ if } \frac{1}{|c_i|} \sum_{x \in c_i} \frac{d(x,\mu_i)}{d(x,\mu_j)} > 1 \\ 1 - \frac{1}{|c_i|} \sum_{x \in c_i} \frac{d(x,\mu_i)}{d(x,\mu_j)} \text{ otherwise} \end{cases} \tag{2.22}$$

The Wemmert-Gancarski index takes its values between 0 and 1, 1 meaning that the clusters are very compact and well separated. When applied to a complete clustering result, it is defined as follows:

$$WG(C) = \frac{1}{\sum_{k=1}^{K} |c_k|} \sum_{i=1}^{K} |c_i| WG(c_i) \tag{2.23}$$

2.6.2 Supervised indexes

Whenever the real objects classes are known, it is possible to compare the result of a clustering with the real partition. While these external criteria are not proper clustering indexes, it is the most convenient way to evaluate a new clustering algorithm by applying it to a data set for which the classes or clusters are known. Indexes that makes it possible to rate a clustering based on the comparison between a clustering partition and the real classes are called supervised indexes or external criteria, because they rely on information that comes neither from the data nor from the clustering but from an external ground truth used for comparison purposes.

A non-exhaustive list of external criteria is available in Appendix B.

2.7 Problems and limits of clustering

Despite the fact the clustering algorithms have been around for a while now and have proven their efficiency in a broad range of domains, there are still many unanswered questions and open issues about this field. Most of these problems are the direct consequences of the lack of a formal definition of what a cluster is and what its properties should be, which in turn comes from the many possible definitions for the notion of similarity between objects and between clusters. Given these issues, it is also very difficult to define a proper objective function to solve a clustering problem using already existing optimization methods. The second type of problem that one may encounter when dealing with clustering comes from the unsupervised nature of this task and lack of knowledge on the data themselves.

The recurrent questions on this topic could be summarized as follows:

- What is a cluster? What should it look like?
- How do we define the notion of similarity between two objects?
- Which clustering algorithm should be used? With what parameters?
- Are there any clusters to be found in my data sets? If yes, how many?
- Should the data be normalized?
- Are all the attributes relevant to my clustering problem?
- Are there any outliers? How do we deal with them?
- How do we know whether a clustering partition is good or not?

Some of these issues are addressed thereafter.

2.7.1 Choosing the right number of clusters

Choosing the right number of clusters is one of the most difficult and common problem in the field of clustering. Several algorithms (such as the K-Means algorithm or the EM algorithm) actually require the number of cluster to be given as a parameter. However, in the case of true clustering problems with new data that have never been mined yet, the number of clusters to be found is generally unknown.

The most common solution to this problem consists in running the algorithm several times looking of a different number of clusters. A comparison of the results is then established based on a selection criterion to pick the best solution. However, not only this solution is costly because it requires to run the algorithm several times, but it is also very subjective because the choice of the best solution will be very dependent on the choice of the criterion.

The two most common criteria are the Bayes Information Criterion (BIC) [118] and the Akaike Information Criterion (AIC) [1]. These two criteria are very close and have a strong statistical basis [26, 142]. They work well with probabilistic clustering methods (see Section 2.5), but they can cumbersome with other types of clustering algorithms. Furthermore, they are time consuming as they both require to run the algorithm multiple times.

The Minimum Description Length criterion (MDL) [63] and the Minimum Message Length criterion (MML) [43] are two other possible choices that are considered to be more objective. They consists in beginning with a relatively high number of clusters and then merging them to optimize either of the two criterion (MDL or MML). They are well adapted for hierarchical clustering (see Section 2.3.2).

Ultimately, it is often better to ask the opinion of an expert to determine the ideal number of clusters. In some cases, criteria such as the BIC, AIC, MDL or MML may help the expert to decide which number of clusters is the best [152].

2.7.2 Cluster validation

The goal of clustering algorithms is to find the intrinsic structures of a data set in order to split the data into several clusters. However, this supposes that there are clusters to be found, which may or may not be the case. In practice, some data sets do not have any remarkable structures and there are no clusters to be found.

Because we generally do not know if there are clusters to be found before launching the clustering process, it is necessary to check the validity of the clusters found by the said clustering algorithm. Even when there are clusters to be found, it is still common sense to check whether the resulting partition is meaningful. To do so, the clusters need to be checked in a quantitative and objective way [71]. Thus unsupervised criteria introduced in Section 2.6.1 can be very useful.

The notion of stability [11, 94] is another elegant way to assess the validity of a clustering result. A clustering is said to be stable if the partitioning remains similar when the data set or the clustering process are perturbed. For instance, when similar clusters are found over several random sub-samples of a data set, then these clusters can be considered stable. Another example would be to use several times the same algorithms with slightly different parameters. If this algorithm always finds similar clusters, then this partitioning is stable too. This can be useful with algorithms such as the K-Means algorithm that are initialized randomly, to check that the clusters are not random too. There are also interesting applications with model based algorithms such as the EM Algorithm using the Gaussian Mixture Model. In this later case, we will want to check that the parameters of the Gaussian distributions remain similar on sub-samples of the data.

2.8 Discussion

In this chapter, we have studied the broad concept of clustering, the main existing methods, as well as different ways of validating clustering results. We have also discussed several issues and limitations inherent to this type of unsupervised learning. Among other things, we have highlighted that there is a huge number of available clustering methods with a lot of parameters, and that it is very difficult to pick the right one and to find the best parameters. Nevertheless this choice has a heavy influence on the quality of the results and cannot be overlooked. In an attempt to solve this problem, in Chapter 4 we will introduce the concept of *collaborative clustering* which among other things makes it possible to have several clustering methods working together. Using this paradigm, it is therefore possible to combine the information provided by several algorithms and thus to partially address the issue of choosing the right clustering method.

Another point is that the choice of the right clustering method also depends on the type of data set studied. In the next chapter, we will study clustering methods that are adapted to image data sets and more particularly to the case of very high resolution satellite images.

Chapter 3

Clustering applied to image data sets

Contents

In this chapter, we talk about the specific case of clustering applied to images. To do so, we will introduce the different steps, and some of the possible algorithms to process and analyze an image data set. Then, we will present some algorithms that we have been developing and testing during the course of this PhD thesis.

3.1 Introduction to data mining applied to images

Data mining applied to images is a trendy field since the early 1980 and is more commonly known under the name *Computer Vision*. It has numerous applications: analysis of medical images, security scans, remote-sensing, etc. The final goal of computer vision is to mimic the skills of human vision when analyzing an image: finding the different element of the image and eventually identifying them.

Automatizing computer vision is a complex task the goal of which is to reduce the need for human intervention when extracting knowledge from an image. This process can be split into several steps:

1. Acquiring the data
2. Preparing the data (denoising, encoding, and so forth.)
3. Mining the data (choosing a model and a method, and then applying it to the data)
4. Validating and interpreting the results
5. Integrating the mined knowledge into a data base for further use (optional)

In this chapter, we will focus on the data mining step. In particular, we will focus on what makes images specific types of data sets, and on how to apply clustering algorithms like these introduced in the previous section to such data. To do so, we first need to consider the nature of an image: from a computer point of view, an image is a data set made of several elementary objects called pixels. These pixels have characteristics of their own such as colors (levels of red, green and blue for regular images, but there can be more or less color channels for multi-spectral images), or position (x,y coordinates) on the image. Therefore, one of the main difference between an image set and a regular data set is that in the cases of images, the data to classify have coordinates -and thus geographical dependencies- in addition to their regular attributes. Another important difference is that the pixels are not necessarily the most interesting elements to process using a machine learning algorithm. As we will discuss bellow, while it is possible to use a machine learning (such as a clustering algorithm) directly on the pixels, it is quite often more interesting to process "*regions*" that are made of several pixels.

3.1.1 Pixel-based approaches

3.1.1.1 Pixels used as data

Pixel-based approaches are among the most common approaches for data mining in computer vision. This type of approaches have been widely studied in the last 30 last years and still remains widely used [75, 73, 109]. These approaches consider the pixels composing an image as data to be labeled (supervised learning) or clustered

(unsupervised learning). Pixels are described as vectors built from physical attributes (radiometric and geographic attributes), to which can be added *neighborhood dependencies* either in the form of a set of neighboring data, or in the form of coordinates in the computer image itself. By neighborhood dependencies, we mean the possible links between clusters that are next to each other and the way they may influence each other classification or clustering.

For instance, the description of a regular red, green, blue (RGB) computer image of size $H \times L$ is the following:

$$X = \{x_1, ..., x_N\}, \quad x_n = \{r, g, b\}, \quad (r, g, b) \in [0, 255], \quad N = H \times L$$

After the process of clustering (or classification), each pixel x_n will be linked to a cluster (*resp.* a class). In the case of clustering, the clusters do not have any direct semantic meaning, i.e. they are just a number or label. When possible, it is up to the user to link each cluster to an understandable label.

3.1.1.2 Limitations of pixel-based approaches

While pixel-based approaches are still widely used, they suffer from many defects.

First, most pixel-based approaches only use the radiometric information from the pixels leaving all other characteristics unused [10]: shape, length, width, texture, etc.

However, the most important and recent issue with pixel-based approaches is that in modern imaging single pixels have no signification because the objects of interest are generally covering a large number of pixels. This is particularly true with very high resolution (VHR) satellite images such as those that we used in this thesis. When dealing with a high resolution satellite image, even an average quality one, the objects of interest such as roads, trees, or houses are already too big for any of them to fit in a single pixel. They are actually composed of several heterogeneous pixels. Thus a pixel-based analysis not only would make little sense, but it would also result in loosing key information -such as the real shape of these objects- that is key to their identification.

One solution to reduce this kind of issue is to label the pixels, not only based on their own attributes, but also depending on the characteristics and labels of the pixels in their neighborhood [80, 13, 119]. These approaches consider a neighborhood window around the pixel to analyze and add texture information to the pixel color attributes.

While neighborhood enhanced pixel-based methods are an interesting first step, recent studies have shown that in the case of very high resolution pictures it is still not enough to achieve good performances [149]. To cope with these issues, other approaches based on regions of agglomerated pixels have been developed. The principles of these methods as well as their pros and cons are introduced in the next subsection.

3.1.2 Region-based approaches

Region-based approaches are the basis of Object Based Image Analysis (OBIA) [21]. The main idea behind these methods is that since the pixels themselves have no semantic meaning, a first step is required to regroup the pixels into regions that will represent the real objects of interest to be identified or put into clusters. Therefore, for region-based approaches the data mining process consists in two steps instead of only one:

1. Segmentation of the original image to determine the border of the objects of interest

2. Clustering or classification of the newly identified regions

These regions have new and unique characteristics that are based on both the characteristics of the pixels they are made of, but also shape, size and texture features.

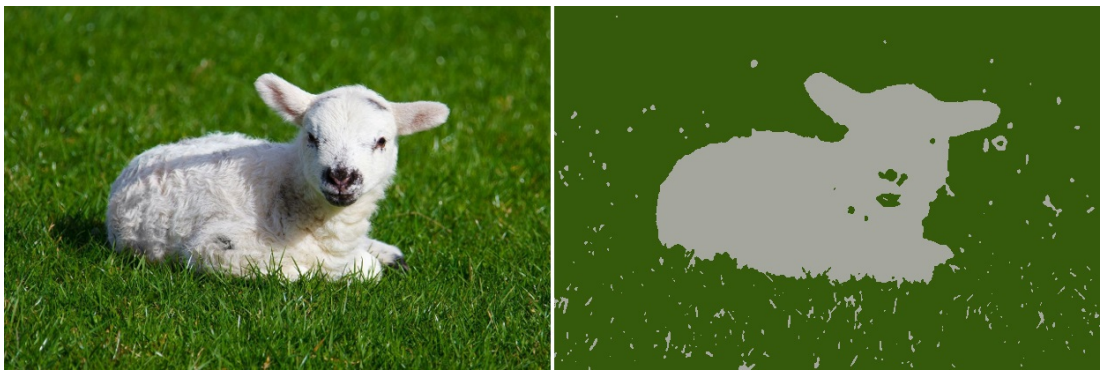


Figure 3.1: Example of a crude segmentation to detect a white lamb on a grass background

3.1.2.1 Image Segmentation

The segmentation of an image is a process that consists in grouping together neighbor pixels with the goal of finding homogeneous segments the borders of which will be a good approximation of the objects present within the image [52]. The segments created using this process are supposed to be relevant and match the real objects that can be found in the picture.

The definition of a proper image segmentation has been formalized by Pavlidis and Zucker [100, 162] in the form of the 4 following axioms:

- Each pixel of the image must belong to one and only one segment.
- Each segment must be continuous, i.e. made of connected neighbor pixels.
- Each segment must be an homogeneous entity.
- Two adjacent segments must be two distinct homogeneous entities.

Among these four conditions, axioms 3) and 4) rely on a notion of homogeneity that - not unlike the notion of similarity in clustering- is rather difficult to assess. Thus, image segmentation is a difficult process that can lead to results of varying quality depending on the homogeneity criterion and the algorithm that are used. Over-segmentation and under-segmentation are the two most common problems:

- *Over-Segmentation*: The image contains too many segments after the segmentation process. In this case, many of the objects to be found remain spread over several small segments that do not contain enough pixels. This problem can generally be solved by merging together segments that are too similar or do not represent anything.
- *Under-Segmentation*: The image does not contain enough segments. The resulting segments are so big that they contain several objects inside of them. Unlike with over-segmentation, this problem is more difficult to solve.

In Figure 3.2, we show an example of over-segmentation: the river and some of the buildings are clearly over-segmented. The colors in the image are representing the real object of interest that "should" be found.

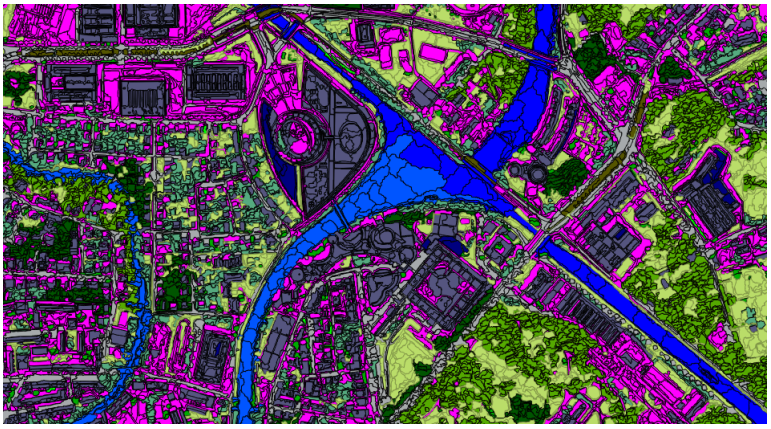


Figure 3.2: Example of an over-segmentation

While both cases should be avoided when possible, there is no generic method that solves these problems. In any case, it is always better to have an over-segmentation rather than an under-segmentation. In the case, over-segmentation the real objects may still be found during the clustering or classification process, even if they are split between several segments. However when several objects are merged in the single segment because of an under-segmentation, there is no way to fix it during the clustering/classification process, and some classes or clusters may be lost for good. Over-segmentation it therefore a much more preferable preprocessing result.

More details on the different segmentation algorithms can be found in the literature [99].

3.1.2.2 Limitations of region-based approaches

While region-based approaches are more adapted than pixel-based approaches when dealing with VHR images, they also have their limits and disadvantages.

The first obvious limitation is the segmentation process needed to create the regions. As we have shown in the previous subsection, this process can be cumbersome and requires that the user choose carefully a potentially large number of parameters to achieve acceptable results. Because the segmentation process is a mandatory step for region-based approaches, the quality of the segmentation will have a huge impact on the subsequent clustering or classification process.

Another important aspect is that when the segments and regions are created they add a large number of new attributes that may have to be taken into consideration: surface of the segments, perimeter and elongation, extrema, variance and average values of the attributes in a given segment, contrast with the neighboring segments, etc. In the study realized by Anne Puissant [107] in her PhD thesis on the redundancy of geometric attributes (surface, perimeter, elongation, etc.), the author concludes that these attributes may or may not be relevant, or redundant depending on the type of objects that one wants to identify.

Finally, another obvious limitation of region-based approaches lies in the fact that -particularly with satellite pictures- there may be several levels of objects of interests to be found depending of the desired level of detail during the clustering process. However,

it is not yet possible for that kind of hierarchy between objects made of other objects to be displayed in a segmentation. Therefore the risk of having an under-segmentation at an acute level of detail remains high, while on the contrary the image may end up being over-segmented for a lesser and broader level of detail.

Example: An urban area is made of several different urban sectors that in turn are made of different buildings and streets.

3.2 The Markov Random Field Model

Now that we have explained the specificity of image data sets and the necessary pre-processing before using an actual data mining algorithm, in this section we introduce a model and a family of clustering algorithms that have been specifically designed for this type of image data sets (either pixel-based or region-based).

As we have shown in the previous section, images can be transformed so that either the pixels or regions made of several pixels can be used like regular data in order to apply a clustering or a classification algorithm. However, this transformation neglects two problems:

- As we have discussed neither pixels nor with regions are always representative of the real objects to be found.
- This transformation into a data set breaks the spatial links between the pixels or the regions.

As a result, regular clustering algorithms such as these introduced in the first chapter of this thesis often fail to achieve good results with such data sets, simply because they cannot take into consideration the spatial dependencies between the data. Assuming that we are using either a pixel-based model, or a region-based model that is not under-segmented, the clustering of such images can be achieved by using a Markov Random Fields based representation of the images' data [110]. Markov Random Fields (MRF) rely on the notion of neighborhood to represent the dependencies between two neighbor pixels or regions.

A *Markov Random Fields network* is a graphical probabilistic model aiming to take into consideration the neighborhood interactions between the data in addition to the observed a priori knowledge [79]. This model allows to consider the explicit dependencies between the data and to weight their influence. In the case of image segmentation, these dependencies are the links between two neighbor pixels or segments/regions (patches of pixels). Markov Random Fields networks rely on this notion of neighborhood, and are represented as non-oriented graphs the vertices of which are the data, and the edges the links between these data. This additional information on the data has been shown to significantly improve the global results of the image segmentation and clustering processes [65, 3, 114].

The *Hidden Markov Model* is also a probabilistic model in which a set of random variables $S = \{s_1, ..., s_N\}$, $s_i \in 1..K$ are linked by neighborhood dependencies and are emitting observable data $X = \{x_1, ..., x_N\}$ where the x_i are the vectors containing the attributes of each observation. The goal is then to determine the optimal configuration for S, i.e. finding the values of the s_i in order to get the clustering, which will eventually repair the segmentation.

The *hidden Markov random field model* is the application of the hidden Markov model to the specific dependency structure of the Markov random fields. This model is quite common in image segmentation [156]. An example of a typical structure for the HMRF model is shown in Figure (3.3).

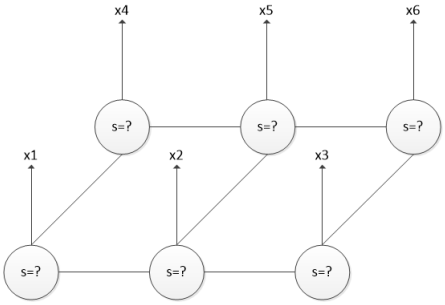


Figure 3.3: An example of graph modeling the HMRF model.

The main goal of the hidden Markov random field model applied to image segmentation is to find the right label for each pixel or segment of a picture in order to get a clustering of homogeneous and meaningful areas. There are several algorithms that can do a clustering or a segmentation of an image represented as a MRF network: the Graph-Cut Algorithm [24], the Integer Projected Fixed Point method [90], the Graduated Non-Convexity and Concavity Procedure [92], and the Iterated Conditional Modes (ICM) [13].

3.2.1 The ICM Algorithm

3.2.1.1 General Framework

In this section, we introduce the standard version of the ICM algorithm [13].

In its original version, the ICM algorithm principle is the following: First a regular clustering algorithm is applied on the data without considering the neighborhood dependencies (K-Means, EM, etc.). Then, considering a neighborhood window of fixed size, the original clustering result is improved using the influence of the spatial dependencies between the data.

Formally, given a data set $X = \{x_1, ..., x_N\}$ and a hard partitioning $S = \{s_1, ..., s_N\}$, it consists in maximizing an energy function over all the data as shown in Equation (3.1) where $u(s, x)$ is an energy function computed for each data.

$$\tilde{S} = \underset{S}{\operatorname{argmax}}\, U(S, X) = \underset{S}{\operatorname{argmax}} \sum_{n=1}^{N} u(s_n, x_n) \tag{3.1}$$

The data energy function $u(s, x)$ can be split into a local term and a neighborhood term as shown in Equation (3.2), where x is the observed data, s a potential hidden label evaluated for x, V_x the neighborhood of x (i.e. a set containing all its spatial neighbors in X), s_v the label linked to a neighbor data $v \in X$. The function $u_{loc}(s, x, \theta)$ is a local energy function evaluating the likelihood of s being the hidden label given the observation x and some parameters θ. The function $u_{neighb}(s_v, s)$ is a neighborhood energy function that assesses the likelihood of two label s and s_v to be neighbors in the

image. And β a user input parameter that determines the influence of the local term compared with the neighborhood term.

$$u(s,x) = u_{loc}(s,x,\theta) + \beta \sum_{v \in V_x} u_{neighb}(s_v, s) \tag{3.2}$$

In Figure 3.4, we show an example of two different types of neighborhoods, with 4 and 8 neighbors. In this figure, we assume that we use the pixel model. The red circle would then be the currently observed data, and the black circles would the neighboring data the labels of which will be used to compute the value of the neighborhood energy $u_{neighb}(s_v, s)$.

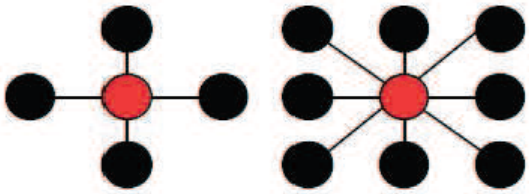


Figure 3.4: Example of first order neighborhood with 4 and 8 neighbors respectively for the pixel-model.

The simplest form for Equation (3.2) that was used in the original ICM algorithm is the Potts model [13, 146] shown in Equation (3.3). In this Equation, $\delta_{a,b}$ is the Kronecker delta where $\delta_{a,b} = 1$ if $a = b$ and 0 otherwise.

$$u_{POTTS}(s,x) = \delta_{s,s^{t-1}} + \beta \sum_{v \in V_x} \delta_{s,s_v} \tag{3.3}$$

As one can see, the principle of Equation (3.3) is the following: the local energy function penalizes a change in label from one iteration to another, while the neighborhood energy gives a penalty equal to β whenever spatially neighbor data have different labels. What this energy function does is that it tries in a crude way to smooth the final clustering by favoring large neighborhoods of pixels or regions that belong to the same cluster.

3.2.1.2 Two-step model with the EM algorithm

While it is true that the ICM algorithm is not the most effective MRF-based segmentation/clustering method, it still has several advantages:

- It is the fastest algorithm among these presented in the previous section. Considering that the main application for this thesis are VHR satellite images with a very high number of segments and neighborhood dependencies -see 3.3.3-, a faster algorithm is a good option. Algorithms that are considered the most effective such as the Graph-Cut algorithm simply cannot scale for this type of images due to the large size of the spatial dependencies graph.

- Since in our experimental part we will use a region-based satellite image data set. The data have already been preprocessed and denoised, thus reducing the risk that the ICM will perform poorly.

- The algorithm has been improved several times since its creation. One notable improvement is that the ICM algorithm can be coupled with the EM algorithm for the Gaussian Mixture Model [78, 160].

The main idea is that since the ICM can easily be coupled with the EM algorithm, this property can be used to make the clustering process easier. In the EM Algorithm, we try to optimize a *Maximum a posteriori criterion* such as:

$$\tilde{S} = \underset{S}{\operatorname{argmax}}\, P(S|X,\Theta) = \underset{S}{\operatorname{argmax}}\, P(X|S,\Theta)P(S) \tag{3.4}$$

Using the two-step EM-ICM, this clustering problem can be solved as follows:

- Research of the prototype parameters Θ and maximization of $P(S|X,\Theta)$ using the Expectation Maximization algorithm (EM) [38], without considering the dependencies.

- For Θ fixed, optimization of a modified version of Equation (3.4) that considers the neighborhood dependencies, using the Iterated Conditional Modes Algorithm.

Equation (3.4) develops as follows:

$$P(X|S,\Theta)P(S) \propto \prod_{n=1}^{N} P(x_n|s_n,\theta_n)p(s_n) \tag{3.5}$$

If we use a Gaussian mixture model and add the neighborhood dependencies, locally Equation (3.5) becomes:

$$P(x|s,\theta)p(s|V) \propto \frac{1}{Z}\mathcal{N}(x,\mu_s,\Sigma_s) \times \prod_{v\in V_x} \exp^{\beta\times\delta_{s,s_v}} \tag{3.6}$$

In turn, using the logarithm of Equation (3.6) we can obtain an energy function with a form matching that of Equation (3.2).

$$u_{Gauss}(s,x) = -(x-\mu_s)^T\Sigma_s^{-1}(x-\mu_s) - \ln(\sqrt{|\Sigma_s|(2\pi)^d}) + \beta\sum_{v\in V_x}\delta_{s,s_v} \tag{3.7}$$

The complete 2-step algorithm using the Expectation-Maximization algorithm and the Iterated Conditional modes is described in Algorithm (7).

Algorithm 7: EM+ICM Algorithm

```
Initialize S and Θ with the EM algorithm
while the algorithm has not converged do
    for each x ∈ X do
        Find s that maximizes u_Gauss(s, x) as defined in Equation (3.7)
    end
end
Return S, A
```

3.3 Enriched Markov Random Fields and C-ICM algorithm

In this section as well as in the following experimental subsections, we introduce our contribution to the field of clustering for VHR satellites pictures in the form of a modified energy model leading to a semantic-rich version of the ICM algorithm [126, 125, 131].

3.3.1 Model

The method that we introduce thereafter has been proposed to cope with two major weaknesses found in the regular energy function of the Markov Random Field Model from Equation (3.2):

- The penalty in the neighborhood term of the original energy model can take only two values 0 or β. This is very limitating and makes β a critical parameter with a heavy influence on the final results.
- This model was thought and designed for the pixel-model and is unable to cope with irregular neighborhoods: different data having different number of neighbors sharing a different portion of their border (see Figures 3.5 and 3.9). This second limitation makes this model complex to use with region-based data sets such as very high resolution images data sets that feature regions of various sizes and shapes with highly irregular neighborhoods.

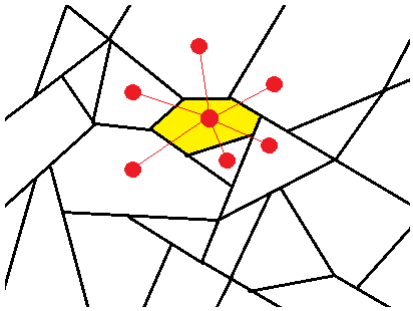


Figure 3.5: Example of a highly irregular neighborhood

To remove these weaknesses, we took some inspiration from the regular Hidden Markov Chain Model which features a matrix A of size $K \times K$ that stores the transition probability from one state (cluster) to another in time series. Using the same notation, let $A = \{a_{i,j}\}_{K \times K}$ be the probabilistic matrix describing the probability to have two spatially neighbor data (pixels or regions) linked to clusters c_i and c_j respectively on a MRF model. Using this matrix, Equation (3.7) can be modified so that we have a better neighborhood energy function as shown in Equation (3.8) where $\tau_{x,v}$ is the percentage of border occupied by neighbor v regarding to the observed segment x.

$$u_{CICM}(s,x) = -(x-\mu_s)^T \Sigma_s^{-1}(x-\mu_s) - \ln(\sqrt{|\Sigma_s|(2\pi)^d}) + \sum_{v \in V_x} \tau_{x,v} \ln(a_{s,s_v}) \quad (3.8)$$

Since the $a_{i,j}$ values are between 0 and 1, our neighborhood energy is a positive penalty function. This penalty function can take an infinite number of values (up from

K^2 for τ fixed) instead of only two values (0 or β) for the regular neighborhood energy functions. We note that this fitness function is as difficult to optimize as those from other models and that there is no warranty that the global optimum will be found.

The idea behind this energy model is that the ICM algorithm is already making an approximation by considering that the labels found in the previous iterations can be used to evaluate the potential neighborhood energy in the current iteration. We are using this same approximation to approach the probability of a given spatial transition between two clusters.

3.3.2 Proposed Algorithm and properties

The main difficulty in our approach is that we have to evaluate the transition matrix A in order to run our HMRF-EM algorithm. Without some external knowledge on the compatibility between the different clusters, there is no way to get the exact values for the transition probabilities. It is however possible to approach them by evaluating the *a posteriori* transition probabilities at the end of the EM algorithm and after each step of the ICM algorithm. This can be done using Equation (3.9). The matrix A also has the property of summing to 1 over the lines, whenever a cluster is not empty: $\forall i, C_i \neq \emptyset, \sum_{j=1}^{K} a_{i,j} = 1$.

$$\forall (i,j), a_{i,j} = \frac{\sum_{x \in X} \sum_{v \in V_x} (\tau_{x,v} \cdot \delta_{s_x,i} \cdot \delta_{s_v,j})}{\sum_{x \in X} \delta_{s_x,i}} \tag{3.9}$$

Algorithm 8: EM + C-ICM Algorithm

```
Initialize S and Θ with the EM algorithm
Initialize A
while Tr(A) increases do
    for each x ∈ X do
        Find s that maximizes u_CICM(s, x) as defined in Equation (3.8)
    end
    Update A
end
Return S
```

The full algorithm is shown in Algorithm 8 and Figure 3.6. Since we want to keep the spirit of the original ICM algorithm, our stopping criterion is based on the Trace of the Matrix A.

$$Tr(A) = \sum_{i=1}^{K} a_{i,i} \tag{3.10}$$

Since the values on the diagonal of A are linked to the probability of data in the same cluster to be neighbors, the higher the Trace, the more compact the clusters will be. Therefore $Tr(A)$ seems to be a good stopping criterion. Since this algorithm relies on an explicit compactness criterion, we renamed it "C–ICM", where the "C" stands for "compactness".

An unexpected benefit from the modification that we made to the original ICM algorithm is that unlike any other version of the ICM Algorithm, our proposed version has some low level semantic information stored in the Matrix A. As we have already explained, the diagonal of A contains information on the compactness of the different

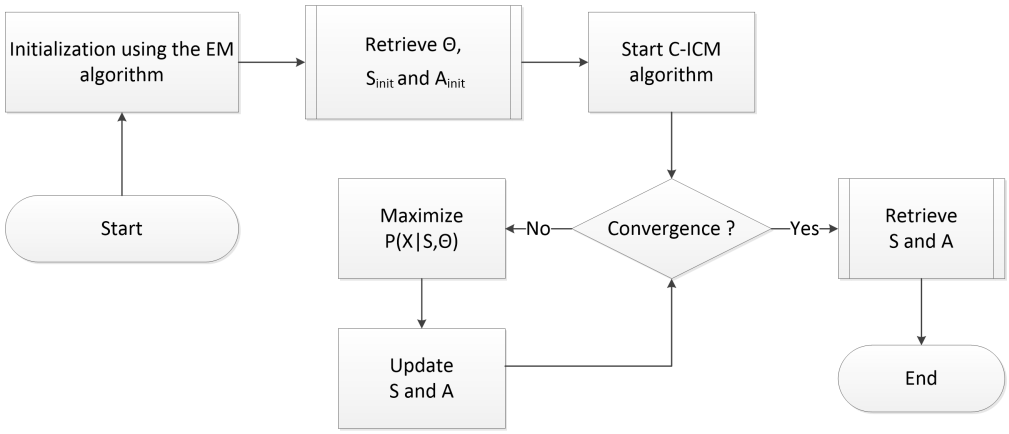


Figure 3.6: General framework of our proposed C-ICM algorithm

clusters. But the non-diagonal elements of A contain information that are no less interesting about the compatibility and organization of the different clusters in the image. This interesting property that we did not exploit when we first published on the new energy model later prompted us to change the name of our algorithm into "SR–ICM" for Semantic Rich Iterated Conditional Modes algorithm.

An example of some basic semantic interpretation from a small transition matrix is shown in Figure 3.7.

$$A = \begin{matrix} 0{,}12 & 0{,}01 & 0{,}17 & 0{,}70 \\ 0{,}02 & 0{,}81 & 0{,}01 & 0{,}16 \\ 0{,}13 & 0{,}03 & 0{,}50 & 0{,}24 \\ 0{,}36 & 0{,}25 & 0{,}14 & 0{,}25 \end{matrix}$$

- Areas of clusters 1 are surrounded by areas of cluster 4
- Areas of clusters 1 and 4 have a low compactness
- Clusters 1 and 2, as well as 2 and 3 are incompatible neighbors

Figure 3.7: Example of a probabilistic transition matrix and its semantic interpretation

Once projected on a ground truth or with the help of an expert, the information from the transition matrix A can have literal interpretations. For example, in the case of a satellite picture, we could have something like: "House are surrounded by gardens", "buildings are never in the middle of water areas", "there are huge forest areas on this image", and so forth.

3.3.3 Presentation of the VHR Strasbourg data set

The VHR Strasbourg data set is a complex set acquired from a very high resolution image of the French city of Strasbourg, $1px = (50cm)^2$, an extract of which is shown in Figure 3.8 and the fully dezoomed image in Figure 3.11. Other extracts can be seen on Figures 3.10(a), 3.10(b), 3.10(c) and 3.10(d).

This image has been segmented into a data set made of 187058 regions, each of them described by 27 attributes either geometric or radiometric [111]. These attributes

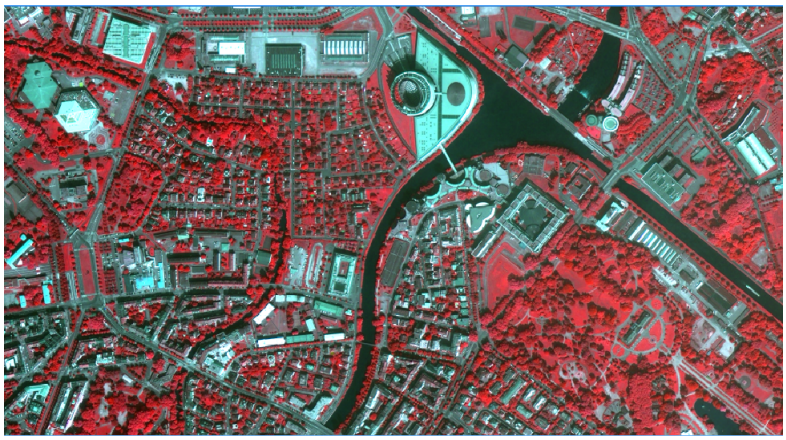


Figure 3.8: Extract of the original source image (approximately 1/25 of the full image), Pleiades ©CNES2012, Distribution Astrium Services / Spot Image S.A., France, All rights reserved.

include the geographic position of the segments, the surface of the area they cover, their mean RGB values, the contrast compared to neighbor pixels and segments, the brightness, and the standard deviations, as well as attributes such as vegetation indexes, among other things.

In addition to this information, this data set provides the neighborhood dependencies between the segments: number of neighbors, id number of neighbor segments, and relative percentage of shared border. For this data set, each segment has between 1 and 15 neighbors.

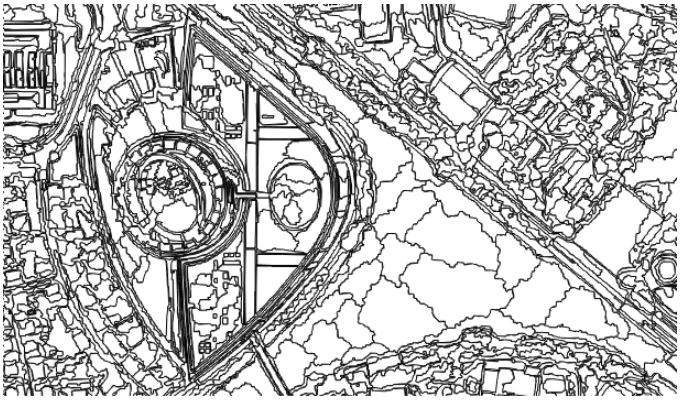
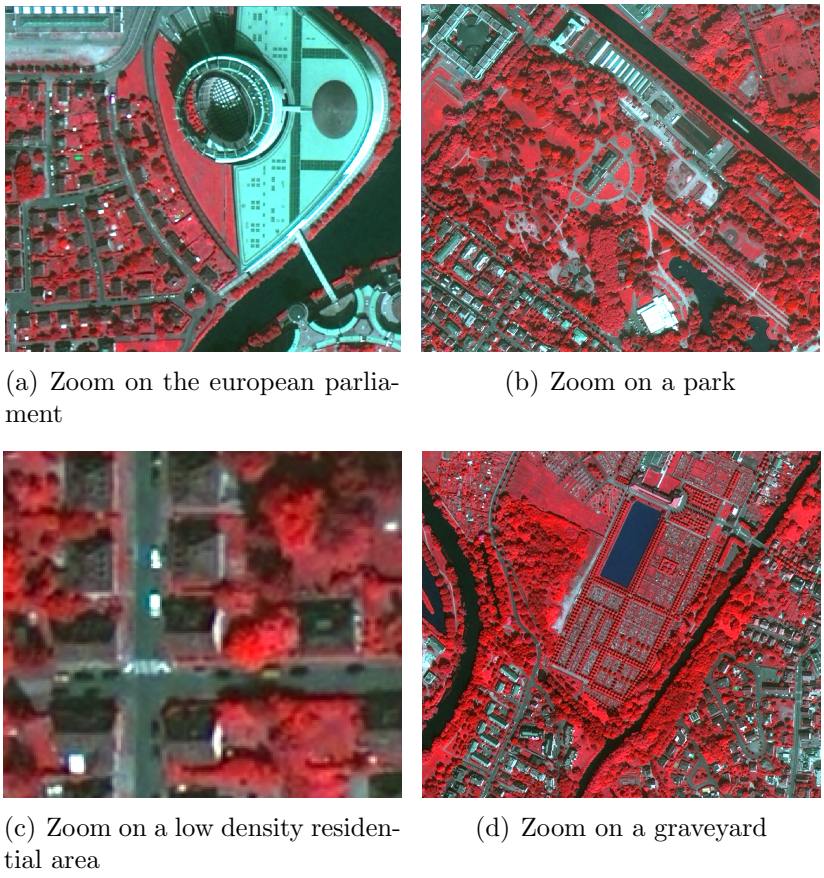


Figure 3.9: Example of segmentation in the central area of Strasbourg. The image is clearly over-segmented

The segments in this data set have highly irregular shapes (cf. Figure 3.9), and consequently the neighborhoods themselves are also irregular. Unlike in the pixel model where 1st order neighborhood usually include 4 or 8 neighbors, in this data set each segment can have 1 to 15 neighbors depending on its shape and position on the image.

3.3.3.1 Ground truth

In order to validate our results, we had to find a ground truth. It was however too tedious a task to manually label the 187058 segments. Therefore we decided to rely on maps of the area made by expert geographers (cf. Figure 3.12(a)). These maps were



(a) Zoom on the european parliament

(b) Zoom on a park

(c) Zoom on a low density residential area

(d) Zoom on a graveyard

Figure 3.10: Extracts from the original VHR image

produced by a hybrid methodology, mixing data from topographic databases for roads and buildings, a supervised classification for different types of water and vegetation, as well as further manual refinements in order to reduce classification errors.

A closer inspection of the provided ground truth shows that the segments boundaries do not align well with the segments boundaries in the image data (cf. Figure 3.12(c)). Because of that the original ground truth was not suitable to assess the results of our method which considerably relies on radiometric attributes. Consequently, we decided to improve the ground truth by projecting the segmentation into the original maps and using a majority vote approach to determine which expert label should be chosen for each segment depending on the percentage of covered surface for each segment. Finally, neighboring regions having the same label where fused together to obtain the modified ground truth map. The whole process was done at the University of Strasbourg using the software ECognition and QGIS.

It is important to mention that some of the labels provided by the expert geographers are very unlikely to be translated into an equivalent cluster by an unsupervised algorithm. Such labels include various types of vegetation and forest areas based on their density and the total size they cover, 2 types of water areas, and different types of buildings. Among these sub-categories of objects, in many cases the difference cannot be seen from the sky and thus cannot be detected by an unsupervised algorithm.

Among the 15 different classes provided by the expert geographers, we identified 7 to 9 classes that could possibly be found by an unsupervised method. We did so by regrouping those that are very similar or that we deemed technically impossible to distinguish for a clustering algorithm.

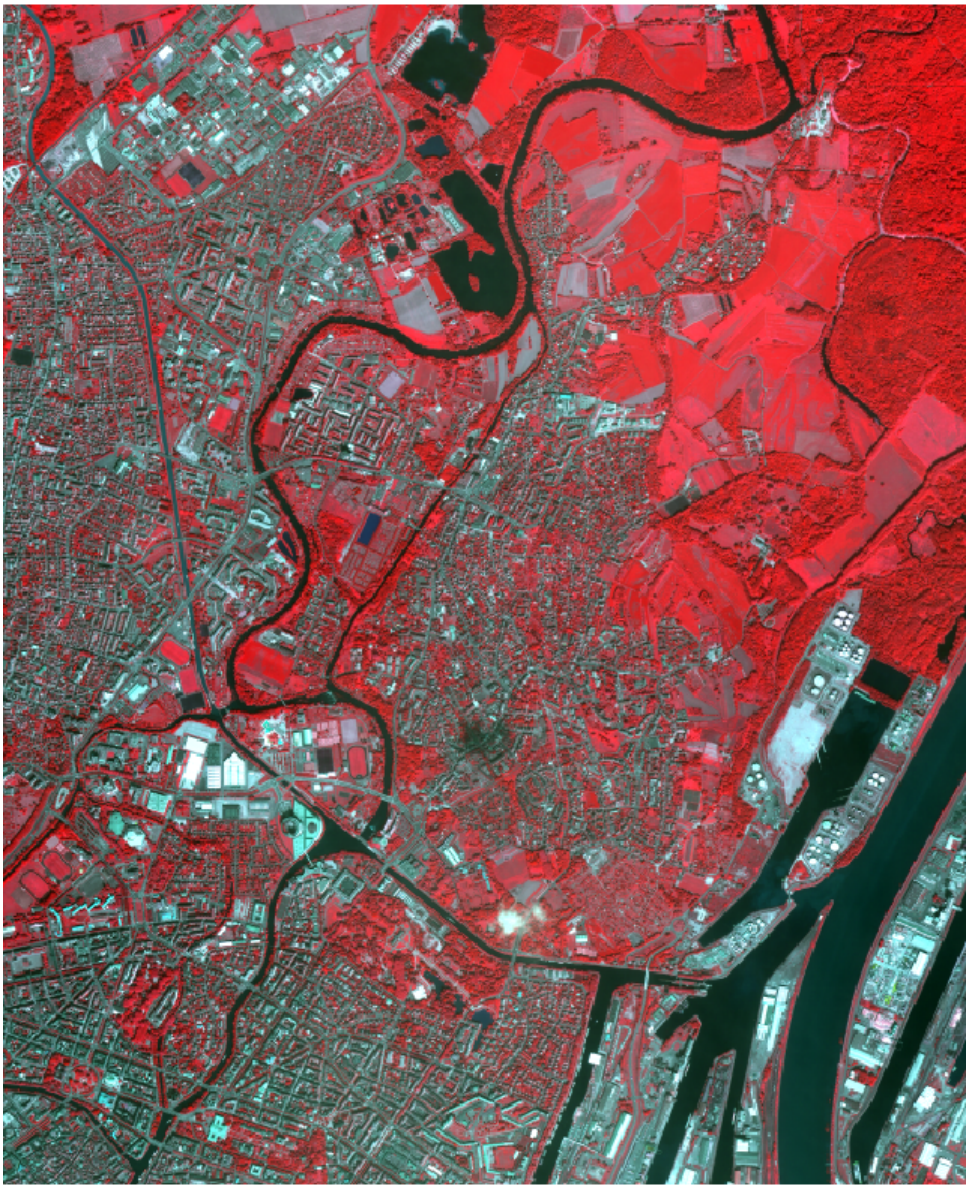


Figure 3.11: Dezoomed version of the full image. ©CNES2012, Distribution Astrium Services / Spot Image S.A., France, All rights reserved.

(a) Original ground truth.

(b) Modified ground truth.

(c) Zoomed boundary representation of the original ground truth maps.

(d) Zoomed boundary representation of the modified ground truth.

Figure 3.12: Extract of the original and improved ground-truth images

As one can see in figures 3.12(b) and 3.12(d) the modified ground truth data corresponds more accurately to the image data. Although, there are still a few errors due to the segmentation which is not optimal. The partner University of Strasbourg has some ongoing work to evaluate and refine this hybrid ground-truth and this ground-truth generation method in general, but the results and conclusions have not been published yet.

3.3.4 Experimental results

3.3.4.1 Experimental results on the VHR Strasbourg data set

In this subsection, we show the results of our MRF-based clustering on the VHR Strasbourg data set using our proposed C-ICM algorithm. To evaluate the quality of our results, we used three different quality measures:

- The Davies-Bouldin Index, see section 2.6.1: an unsupervised quality measure that evaluates the quality of the clusters based on their internal variance and the distance between the prototypes of the different clusters. For this index a lower value is better.

- The Rand Index, see Appendix B: this index compares how close two partitions are. In the context of these experiments we compared our results with our ground-truth made from the expert maps. For the Rand Index, a value close to 1 means a 100% similarity.

- The Probabilistic Rand Index [137, 138]. This index is similar to the original Rand Index, but has been specifically modified to compare partitions from image data sets. Therefore it seemed interesting to use it. This index also takes its values between 0 and 1, with 1 being a full match.

We did not use the adjusted Rand index because of the difference between our clustering results and the ground-truth regarding the number of clusters. This difference tended to bias the adjusted Rand index and to favor solutions featuring a larger number of clusters.

In Table (3.1), we give the results of 3 different instances of the algorithm searching for 7, 8 and 9 clusters. As one can see, the Probabilistic Rand Index results compared with the ground truth are quite good (around 80% of similarity), with the 7-cluster segmentation being the closest one to the ground-truth, followed by the 9-cluster segmentation which is the best one from a clustering point of view (lowest Davies-Bouldin Index). The results of the regular Rand Index are similar except that the 9-cluster segmentation has the best score with this index.

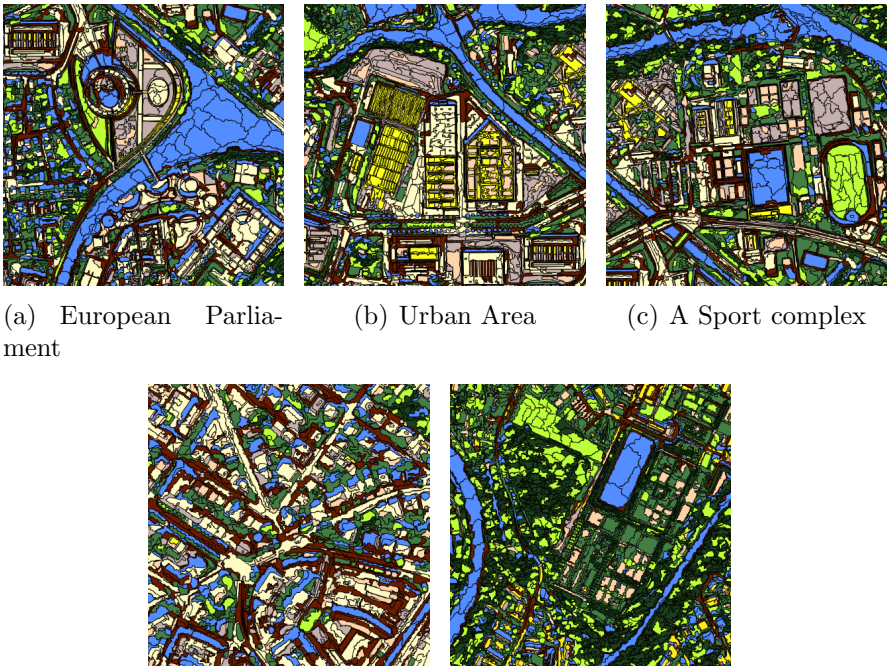
These results are surprisingly good given the fact that the C-ICM is unsupervised, and thus it tends to prove the effectiveness of the semantic rich version of this algorithm to process this type of satellite images. Four extracts of the 9-cluster segmentation are shown in Figure 3.13.

It is however important to emphasize that all 3 clustering results suffer from problems that are very common in image segmentation. Regardless of the 27 attributes, it seems that the original colors still are the most important factors. For instance water areas, shadow areas and dark roofs are often grouped in a single cluster. With our current knowledge of this data set, it is difficult to say whether this problem comes from the segmentation algorithm, from the data preprocessing that created segments

C-ICM	DB Index	Rand Index	Probabilistic Rand Index
7 clusters	2.76 ± 0.31	0.783 ± 0.036	0.826 ± 0.040
8 clusters	3.33 ± 0.24	0.785 ± 0.023	0.796 ± 0.017
9 clusters	3.11 ± 0.27	0.792 ± 0.013	0.817 ± 0.025

Table 3.1: Results achieved by the C-ICM algorithm searching for 7, 8 and 9 clusters: Davies-Bouldin Index, Rand Index, and Probabilistic Rand Index

from these shadow areas, or a bit of both. These segmentation defaults are quite easy to spot on full scale color versions of the image extracts as shown in Figures 3.13(a) and 3.13(d).



(a) European Parliament (b) Urban Area (c) A Sport complex

(d) Strasbourg residential area (e) A graveyard

Figure 3.13: Extracts of the 9-cluster segmentation

On the semantic side, after linking each cluster to the corresponding expert label from the visual results, the following properties were found :

- There is a strong spatial connection between the tree areas and grass areas (dark green and light green respectively), with a transition probability of ≈ 0.65 from a tree segment to a grass segment. This interesting property from the neighborhood matrix A could be translated as "Trees are often surrounded by grass".

- There is a double side low transition probability (< 0.02) from modern urban buildings segments (in yellow) to water segments. It matches with the fact that modern buildings are rarely built directly adjacent to a river, nor in the middle of it.

- The water cluster has an average transition probability to itself (≈ 0.47) . This is consistent with the river and water areas of the city of Strasbourg being rather linear and small.

While these results may seem obvious, considering that they are coming from an unsupervised algorithm that has no external knowledge on the true nature of each cluster, they are still quite impressive.

Cluster Number			Computation time
C-ICM	7	clusters	27,630ms
C-ICM	8	clusters	31,495ms
C-ICM	9	clusters	34,438ms

Table 3.2: C-ICM algorithm average Computation times for different number of clusters

Finally, in Table (3.2), we give the computation times associated with the previous experiments for 10 iterations of the EM algorithm followed by the C-ICM algorithm iterating until convergence. These computation times were acquired using an i5-3210M 2.5GHz processor on an OpenMP parallelized version of our C++ code. The data set processed by the algorithms weighted around 60MB and included : the 187,058 segments' 27 attributes and their neighborhood dependencies graph.

As one can see, the computation times are quite low given the size and complexity of the data set as well as the quality of the results.

3.3.4.2 Experimental results on a pixel-based satellite image

For comparison purposes, we used our algorithm on a pixel-based satellite image, so that we could compare the result of our proposed energy model from Equation (3.8) with the Potts model (Equation (3.3)) and the model adapted for the GMM from Equation (3.7). The original image that we used is made of 1131×575 pixels and can be seen in Figure (3.14) (Copyright 2014 Cnes/Spot Image, DigitalGlobe, Google).

Figure 3.14: From left to right, and from top to bottom : the original image, the result of our algorithm and equation (3.8), the result using energy equation (3.3) and $\beta = 1$, the result using energy equation (3.7) and $\beta = 1$.

As can be seen in Figure (3.14), our energy model achieves a decent segmentation with a fair amount of easily visible elements such as roads, rocky areas and some buildings. On the other hand the classical HMRF-EM algorithm using Equation (3.7) fails to aggregate neighbor pixels from common elements despite a relatively high value for β, while the algorithm based on the Potts model tends to give too coarse results and the different objects of interest in this image are blurry.

While this was not the goal of this experiment, it clearly shows why satellites images should be preprocessed and segmented first, instead of applying the clustering directly to the pixels.

3.4 Discussion

In this chapter, we have introduced the specific clustering application that is image segmentation. In particular, our goal was to propose a method suited for the clustering of data acquired from very high resolution satellite images. To this end, we have studied several of the already existing approaches and we have adapted one of them (the ICM algorithm using a MRF model) to our problem. The resulting algorithm has shown promising results both in a very high resolution satellite image data set and on a regular satellite image data set. Furthermore our proposed methods gives extra semantic information on the clusters that would not be available when using previously existing algorithms. These pieces of information give clues on how the clusters geographically relate to each other in the analyzed image.

We have also shown that while our proposed method and many other methods give relatively good results, image data sets remain difficult to tackle. The whole process from the raw image to the final clustering is a difficult one where initial noise, shadows, deformations, and then segmentation errors and clustering errors accumulate as we move through the different steps. Furthermore, with high resolution satellite images, there are different possible scales of interest available, which makes it difficult to choose parameters as simple as the number of clusters to look for.

In this context, we can see how it could be useful to combine several clustering methods to tackle this kind of image data sets. The benefits would include a higher chance of achieving good results by combining several methods, but could be extended to the possibility of a cross-scale analysis. Various ways of applying these ideas are discussed in details in the next chapter on collaborative clustering.

Chapter 4

Collaborative Clustering

Contents

4.1 Introduction to collaborative clustering

In this section, we introduce the concept of collaborative clustering, a recent topic in the field of unsupervised machine learning. We begin by addressing the origin of the concept of collaborative clustering, its motivations, goals, and applications. Then, we introduce some of the already existing algorithms based on this concept.

4.1.1 Problematic and applications

In Chapter 2 of this PhD thesis, we have shown clustering to be one of the main exploratory task in Machine Learning. As we have already discussed, there is a huge number of existing clustering algorithms that can give very different results with the same data, and choosing between several clustering results is often problematic. This problem can only be solved by asking an expert to choose the most adapted method and the parameters that will work best for a specific data set. This is a very difficult task that can have a heavy influence on the results. Making this kind of decision requires a deep knowledge of both the data to be analyzed, but also of the large amount of algorithms that are available. Furthermore, even with a good expert having a decent knowledge of both the data and the algorithms, it is still difficult to make the *right* choices when it comes to clustering.

In an attempt to solve this problem, the scientific community has suggested several ways of combining the results of different algorithms. The goal was then to use the results and advices of several methods to reach a consensus or a synthesis. These works are based on the intuition that combining results from several sources or experts can help finding a better solution for a given problem.

The marquis of Condorcet was one of the first to formalize this notion [31]. In his "*Condorcet's jury theorem*", he formulates the likelihood that a group of individual reaches an accurate solution. The assumption in the simplest version of the theorem is that a group wishes to reach a decision by majority vote. They all can choose between different choices -one of which is correct-, and each voter has an independent probability p of voting for the correct decision. The theorem then asks how many voters should be included in the group. The answer depends on whether p is greater than or less than $\frac{1}{2}$: if p is greater than $\frac{1}{2}$ (each voter is more likely to vote correctly), then adding more voters increases the probability that the majority decision is correct. In the limit, the probability that the majority votes correctly approaches 1 as the number of voters increases. On the other hand, if p is lower than $\frac{1}{2}$ (each voter is more likely to vote incorrectly), then adding more voters makes things worse: the optimal jury consists of a single voter. In Table 4.1, we show an example of this type of vote where "□" is a right answer and "■" a wrong answer.

Classifier	Prediction	Accuracy
Classifier 1	□■□■■□□□□■□□□□■	10/15 = 0.667
Classifier 2	■■□□□□■□□□□□■□■	10/15 = 0.667
Classifier 3	□□■□□■■□□□□□■■□	10/15 = 0.667
Vote	□■□□□□■□□□□□■□■	11/15 = 0.733

Table 4.1: Example of an improved result when using a simple majority based vote in supervised learning [89]

Approaches based on this idea have been widely studied in supervised learning [117, 150, 81] where they gave birth to the field of *Ensemble Learning*.

While ensemble learning methods and collaborative frameworks have been around for a while in supervised learning (e.g. [25, 27, 7]), in the case of unsupervised learning, collaborative clustering and ensemble clustering are emerging problems and only few works can be found in the literature, (e.g. [103, 105, 39, 51, 42]). The main reason why collaborative and ensemble learning methods are less developed in the domain of clustering is that most supervised methods rely on notions of diversity and quality to decide which algorithms should collaborate, and that the later notion of result quality is much more difficult to evaluate in unsupervised learning.

Due to this inherent difficulty of evaluating the quality of a clustering result, it is only recently that the idea of combining results became popular in the field of unsupervised learning, mainly for two reasons: First, as stated in the beginning of this introduction the vast choice of clustering methods and parameters has made it necessary to use this kind of methods. Second, recent data sets are more complex and led to new clustering paradigms and frameworks (such as the clustering of distributed data) for which the ideas of votes and collaboration constitute elegant solutions. We go into more details on these frameworks in the next subsection.

4.1.1.1 Introduction to modern Clustering Frameworks

We have introduced earlier the concept of clustering as a main exploratory task in Machine Learning. We have discussed the difficulty of this task, reviewed some of the recurrent problems with clustering, and introduced some of the basics algorithms that are commonly used for this task. In this introduction chapter on clustering, we have mostly been concerned with the case of *"easy data sets"* that are not too big, have attributes of the same nature and not too many attributes, that are not split and have usually relatively well formed clusters.

However, this type of *"easy data sets"* does not reflect the reality of the data sets that scientists have been dealing with in the last 20 years. Nowadays, the task of clustering has become even more challenging when the available data sets saw their volume explode with always more data, more features, as well as more multi-view and distributed data sets [161]. Traditional clustering algorithms such as these introduced earlier in this document are either ill adapted to these data sets or simply cannot be used at all.

We introduce thereafter some of the main applications of modern clustering feature complex data to process.

Distributed data clustering: For more than two decades, computing environments and technologies have been evolving towards dynamic and distributed environments that contain massive amounts of heterogeneous, spatially and temporally distributed data sources. This is the case of grid-based or cloud-based environments where the data are distributed between several sources [39]. The most natural solution to this problem would be to centralize all the data in a single site and to apply a regular algorithm. However, this may be impossible for two reasons:

1. The volume of the data. Nowadays data sets may have been split because they are simply too big to be stored and processed in a single computer.

2. Privacy issues. It may not be possible to share or regroup the data split across different sites simply because the said data may contain private information not meant to be read or shared outside of their respective storage site.

To illustrate this case of distributed data, we use the following example [101]: imagine a situation in which we have several organizations or companies that have data concerning the same customers. For instance a bank with banking records, state organizations and hospital with medical information records, retail stores with consumption habits, insurance companies with other information, etc. They are dealing with data on the same individuals but each of them has a data set with descriptors (features) linked to its own activities. All these organizations may want to do a bit of exploratory data mining on there own data set, and they are also aware that the other organizations have similar data sets that could contain information that may or may not be useful to them, for instance to detect patterns that they couldn't see otherwise. Ethical considerations aside, these organizations are forbidden to share their data sets because of privacy and legal issues. However, should any or all of these organization wish to circumvent these legal issues, they would certainly be more comfortable sharing the results of a clustering on their respective data sets, thus keeping their data base private while still sharing anonymous information. Each of them could then use the clustering structures from the other companies to complete their own findings. The typical scheme of a distributed data mining framework is shown in Figure 4.1.

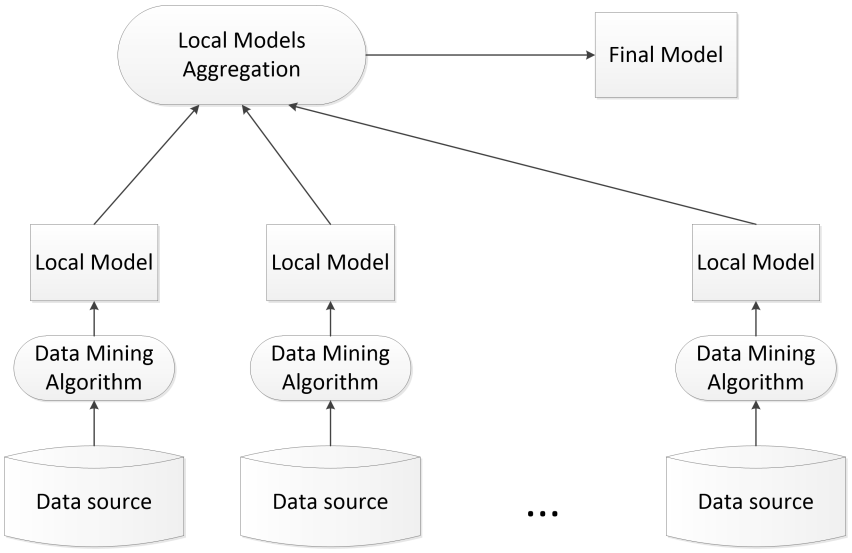


Figure 4.1: Distributed Data Mining Framework

Multi-view clustering: Another very common problem with recent data sets is the huge amount of redundant information available to describe the same objects [15]. This problem is even more complicated when dealing with data described by features of different types: most clustering algorithms have been thought to proceed only one or two (more often one than two) types of attributes, either discrete or continuous, in the form of numbers, text, intervals, or ratios. However, when a single data is described by attributes (features) from several of these categories, defining a distance function (see Section 2.3) becomes a complex tasks require either a deep knowledge of the field, or to assume several biases. There are two possible solutions to this problem: 1) Building a custom distance that will work only for this data set. It usually involves computing a

different distance for all types of attributes and then assessing an average distance based on a weighted combination of all distances. 2) Running different clustering algorithms adapted to each kind of features, and aggregating the results [86].

While the first solution may seem like a good idea, it suffers from a portability issue and usually results in designing a distance and an algorithm that will work for a single data set. Furthermore building the distance will usually require the intervention of one or several experts in order to properly weight the different attributes or sub-distances.

In the case of the second solution, the problem is pretty much equivalent to the previously seen case of distributed clustering where we would have the data attributes split on different artificial sites based on their type. Only this time the different algorithms wouldn't be able to exchange their information directly not because of privacy or legal issue, but because of the incompatible nature of the information itself. Very much like with the clustering of distributed data, there may be redundancies across the different views.

Another issue stemming from data that have attributes of different types and origin is that depending on the observed attributes, different clusters may be found [161]. In this situation, aggregating the results into a global model as shown in Figure 4.1 may not always be the best solution. It may be more advisable to keep all the different clustering results found for the different types of attributes. This later case is not only concerned with the case of attributes of different nature that we have just described. If we consider the example of the VHR Strasbourg data set that we introduced in Section 3.3.3, all the attributes are numerical in nature. However, depending on if we consider only the radiometric attributes, only the geometric ones, or only the attributes added by the segmentation process, different clusters can be found. A clustering based only on the color would probably find different zones such as cities, water areas and vegetation areas, while a clustering based on shapes would be more suited to differentiate between buildings. Still, with this kind of multi-view clustering, each local clustering may or may not benefit from the information sent by other clustering algorithms processing a different view of the same data. In Figure 4.2, we show a typical multi-view data mining framework where the algorithms are not exchanging their information.

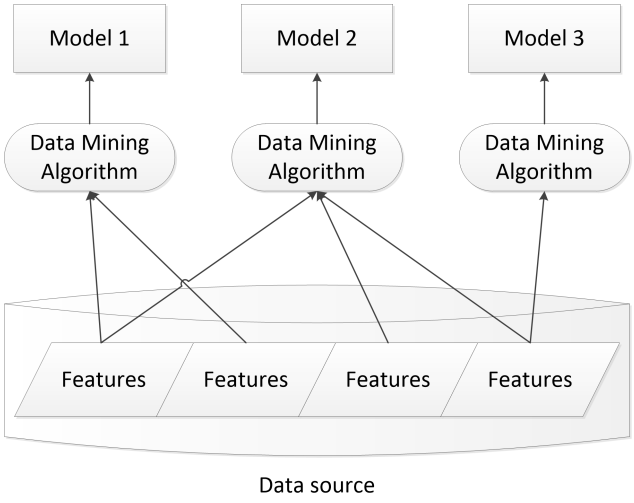


Figure 4.2: Multi-view Data Mining Framework

Clustering of high dimension data sets: In the era of Big Data and huge data sets, another common issue is that data sets tends to be simply too big to be processed

by a single algorithm on a single computer. These issues may come from storage capacity limitations, too many data or too many attributes, or may simply be due to the data arriving at different intervals in time. Processing this kind of data often requires to split the data set along the attributes or to split the full data set into smaller subsets.

In the case where the data are split along the attributes, the solution is to do a multi-view clustering, followed by an a aggregation of the models found, see Figure 4.3.

In the other case, when the data set is separated into several subsets containing less data (because of storage issues or because the data are slowly arriving in time), the problem is a bit different. We will have several algorithms working on different subsets and looking for similar structures. The framework for this case is shown in Figure 4.4.

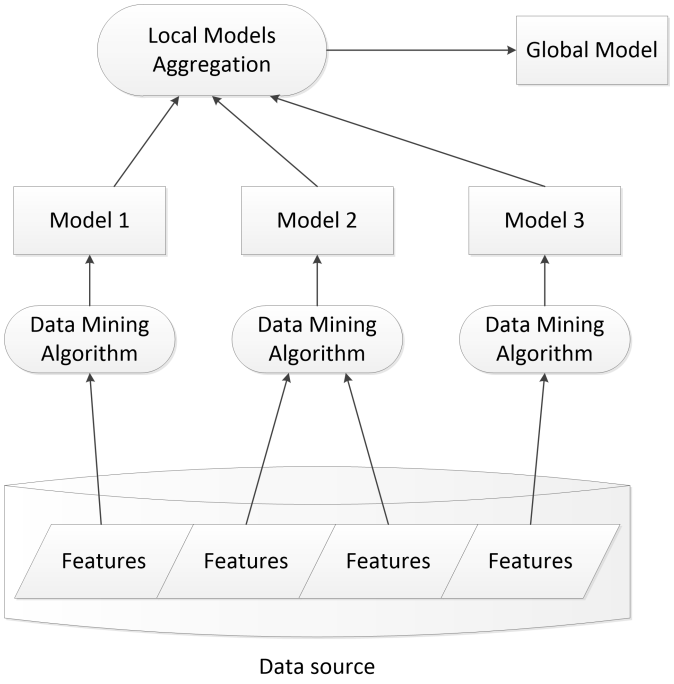


Figure 4.3: Aggregative Multi-view Data Mining Framework

Multi-expert and multi-scale analysis: Finally, even with data sets that are neither distributed, nor big, nor worth a multi-view analysis, data sets in general are complicated and we have the choice between a large number of data mining algorithms that are more or less specialized and may find different structures and patterns for the same data set. There are two sub-cases to this scenario, the two models are illustrated in Figure 4.5:

1. Multi-expert analysis, where we run several algorithms that are known to have different strengths, either with the specific goal of finding different structures, or simply to increase the chances of finding something.

2. Multi-scale analysis, where several algorithms or several instances of the same algorithms are searching for different level of structures, and a hierarchy in the structure [49] (see hierarchical clustering in section 2.3.2).

An example of multi-scale analysis would be to run several algorithms looking for a different number of clusters on a satellite picture. One algorithm would look for

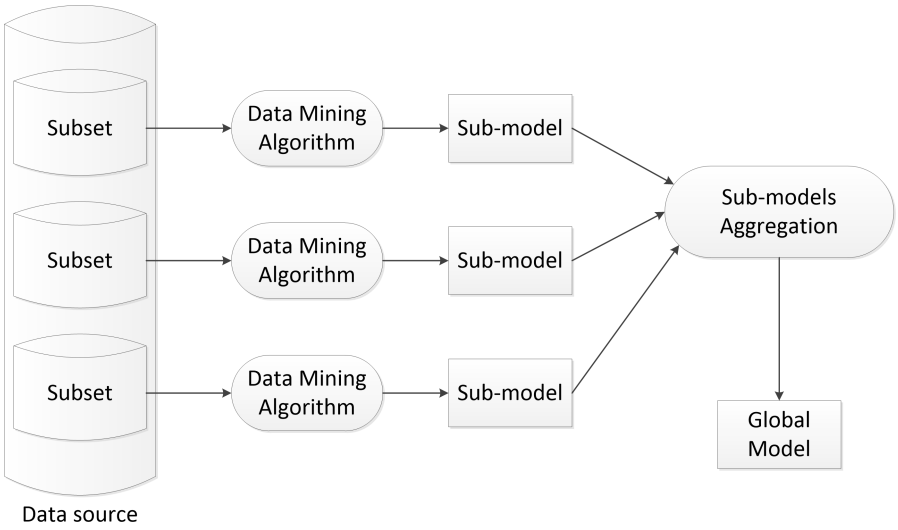


Figure 4.4: Aggregative Framework for Data Mining on several subsets of the same data set

the macro-structures in the landscape (urban areas, water areas, vegetation areas), another algorithm looking for more clusters would zoom a bit and detect more detailed elements (dense urban areas, small density housing areas, rivers, artificial water areas, forest areas, fields, grass areas), and for instance a last algorithm would search on a micro level for more clusters (buildings, roads, houses, trees, bushes, swimming pools, etc.). It is obvious that in such example there is a hierarchical structure between the 3 levels of details.

4.1.1.2 One Clustering Framework to rule them all: Collaborative Clustering

As one can see when looking at Figures 4.1, 4.2, 4.3, 4.4 and 4.5, all the previously introduced complex frameworks have a lot of similarities: we have several algorithms working on the same data or subsets of the same data (subsets of the data themselves or of the attributes), and that will at some point try to aggregate or mutually exploit their results. In all of these cases, one would want these algorithms to be capable of exchanging their information to improve their local results or at least to reduce the diversity of their solutions in order to make the aggregation process smoother. The aggregation process itself is a complex field known as *Ensemble Learning* with several works available in the literature (e.g. [123, 68, 121, 33, 76, 124, 93]). In this thesis, we will only consider the problem of having several algorithms communicating together under various constraints. This problem can be seen as a preliminary step before aggregating results, or can be considered a full process of its own.

Withing this context, *Collaborative Clustering* is an emerging field the goal of which is to have several clustering algorithms working together with a goal of mutual improvement. Collaborative clustering was originally thought to be applied to the specific case of distributed data clustering [101, 102], and later extended to the case of large data sets split into subsets [55]. In this thesis, we argue that given their similarities multi-view clustering, Big Data clustering, multi-expert clustering and multi-scale clustering frameworks can be unified under a common paradigm. In short, we propose a unified framework with common guidelines for the clustering scenarii that we have previously introduced.

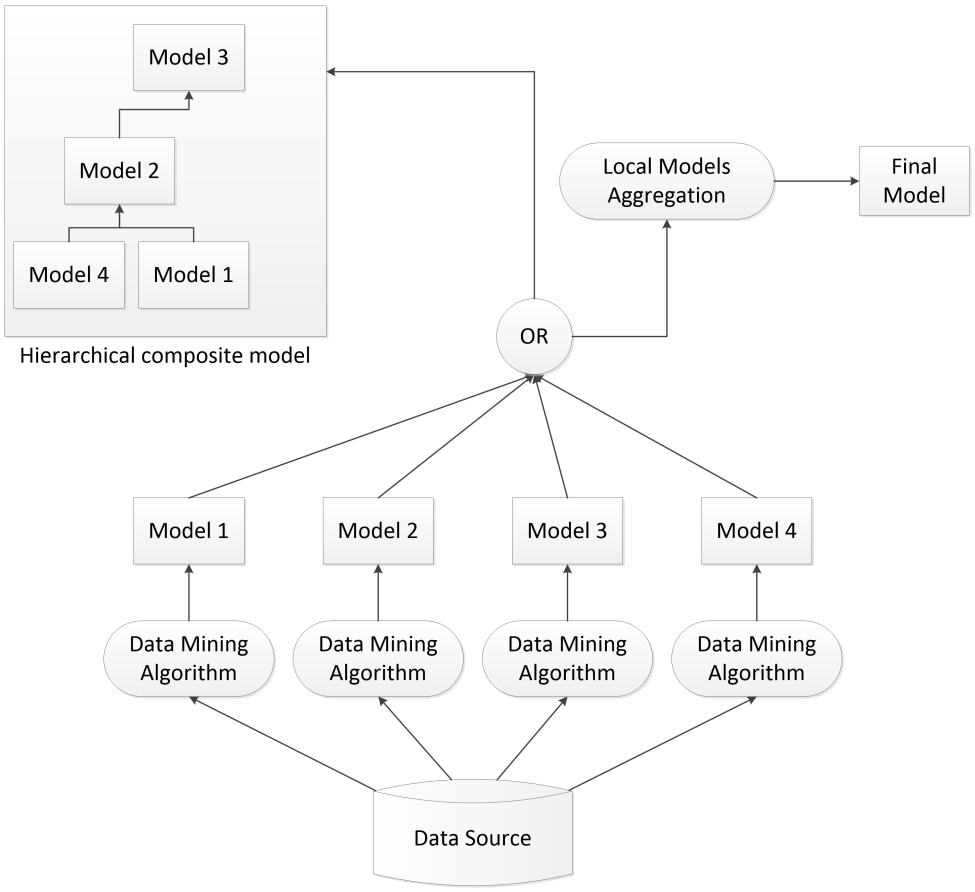


Figure 4.5: Examples of multi-expert and multi-scale Data Mining Framework

Collaborative methods usually follow a two-step procedure [101]:

1. *Local step*: Each algorithm will process the data it has access to and produce a clustering result, e.g. a "summary" and a partition of the data set (this will be more formally explained in the following sections).

2. *Collaborative step*: The algorithms share their results and try to confirm or improve their models with the goal of finding better clustering solutions.

Since collaborative methods and ensemble methods are close and not incompatible, these two steps are sometimes followed by an ensemble learning step (aggregation) which aims at reaching a consensus with the final results after collaboration. In this scenario, the collaborative method is a preliminary step to make the ensemble learning process easier.

The general framework for collaborative clustering is shown in Figure 4.6, where the three possible steps are shown. As we describe it, this framework must be seen as a generic toolbox, where neither the collaborative step nor the eventual consensus steps are mandatory depending on the application. For instance, the collaborative step can be used and repeated as a bloc of its own to improve the local results by exchanging information between the different algorithms. Or it can be used as a preliminary step to improve the solutions and reduce their diversity prior to an aggregation step. While it would appear strange, the collaborative step can also be skipped and the aggregation step plugged directly at the end at the local step.

In this work we will not address the consensus step, and we will instead focus on techniques that can be used during the collaborative step where the algorithms

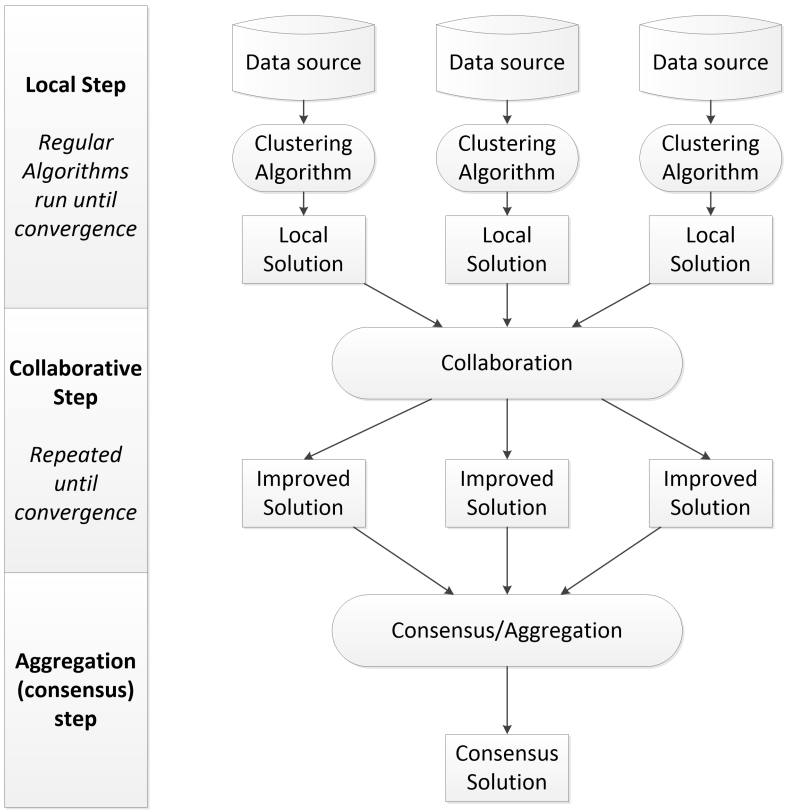


Figure 4.6: Collaborative Clustering: General Framework

are to exchange information with a goal of mutual improvement. Some early works even argue that the aggregation of results acquired from different databases may not be advisable because of potential data redundancies, difference in the structures, and conflicts between the different bases that may result in loss of information when merging the results [157].

As a summary, the problem of collaborative clustering can be defined as follows [50]: Given a finite number of disjoint data sites, collaborative clustering is a scheme of collective development and reconciliation of fundamental cluster structures across these sites.

As one can see, ensemble learning methods and collaborative learning methods are two similar yet different framework families that enable algorithms to work together [105]. The main differences between the two is the following: in collaborative learning, the algorithms share information during the learning process with the goal of improving learning at the level of each participating algorithm; mutual improvement is the goal. With ensemble learning however the objective is to find a global consensus solution by combining the results of local learning, possibly without any exchange during the local learning processes. Thus, ensemble methods aim at finding a single solution using different merging and voting techniques, and collaborative methods aims at sharing the information between several methods and algorithms with the ideal goal of improving all solutions.

4.1.2 Horizontal and Vertical Collaboration

While a large number of applications can fall under the umbrella of a unified collaborative clustering framework, for practical reasons, two main families of frameworks have been coined [104, 55]: *Vertical collaborative clustering* and *Horizontal collaborative clustering*.

- Vertical collaboration refers to the situation where several algorithms look for clusters in different data sets described by the same features but containing different objects. The collaboration is said to be *vertical* because the data set is split alongside the data and thus the information is exchanged vertically, see Figure 4.7. It is closely related to knowledge transfer and transfer learning. It can also be applied to large data sets by splitting them and processing each subset with different algorithms exchanging their information.

- Horizontal collaboration is about several algorithms working on the same objects that may be represented in different feature spaces. The collaboration is said to be *horizontal* because the data set is split alongside the features and thus the information is exchanged horizontally, see Figure 4.8. It can be applied to multi-view clustering, multi-expert clustering, clustering of high dimensional data, or multi-scale clustering.

A third family is sometimes considered in the form of *Hybrid collaborative clustering* [50] where both horizontal and vertical approaches are used at the same time. However, these approaches are often impractical to use due to the lack of common ground between the data bases to exchange their information.

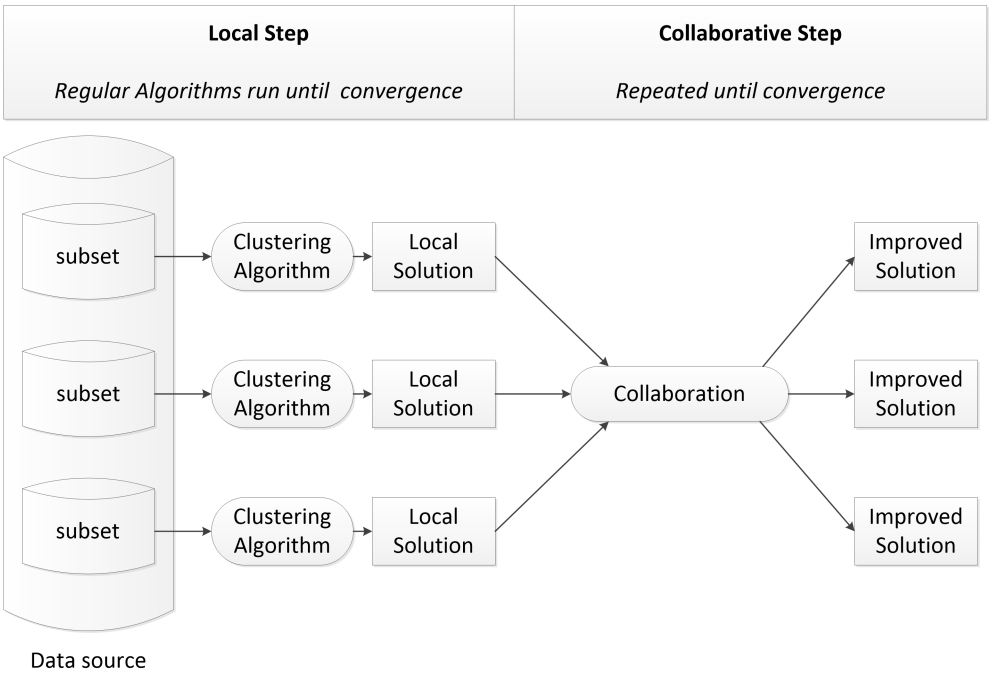


Figure 4.7: Vertical Collaborative Clustering: General Framework

Generally speaking, horizontal collaboration and vertical collaboration are both difficult but in different ways: for horizontal collaboration, the difficulty comes from the fact that the algorithms may not be in the same feature space and thus cannot exchange prototypes or information based on the features of the data. However, since

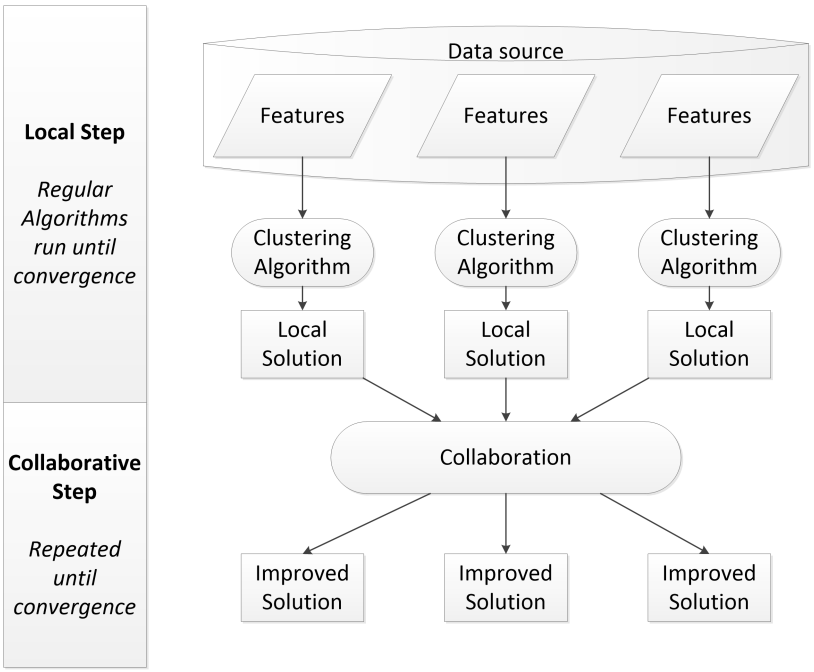


Figure 4.8: Horizontal Collaborative Clustering: General Framework

it is the same data that are split between different sites, comparing the partitions found by different algorithms is quite easy. In the case of vertical collaboration, the difficulty lies precisely in the fact that without any common data, comparing the partitions of different algorithms is impossible and voting systems cannot be used. But, given that all collaborators are prototype-based algorithms and that the data distribution between the different sites are similar enough, vertical collaboration makes it very easy to exchange prototypes and other information on the structures of the clusters between the different collaborators. However, the clusters must first be mapped, and this task can be cumbersome if the distributions are a bit different from one site to another, if the number of cluster is different, or if the algorithms found very different clusters.

4.2 State of the art in collaborative clustering

In this subsection, we introduce several recent collaborative clustering methods.

4.2.1 Collaborative Fuzzy C-Means

One of the first collaborative clustering algorithm was introduced in 2002 by Pedrycz [101, 102] under the name "*Collaborative Fuzzy Clustering*". This method was designed for the specific case of distributed data (See Figure 4.1) where the information cannot be shared between the different sites. This method was based on a modified version of the Fuzzy C-Means algorithm [14] (See Section 2.3.3.2 for the details).

The collaborative Fuzzy C-Means (CoFC) algorithm was the first algorithm introducing the notion of local step and collaborative step:

- **Local step:** Generation of an initial clustering using the regular Fuzzy C-Means algorithm (FCM) on each data site. The FCM algorithm finds c clusters for each data set (it must be the same number for all data sets) and links each data x_n to a cluster i with a certain normalized degree of membership $s_{n,i}$, and stores

this information in a membership matrix S. The FCM algorithm does so by minimizing an objective function $Q[j]$ -see Equation (4.1)- where j refers to the observed data set, $|x_n - \mu_i^j|^2$ the distance between the data x_n and μ_i^j the center of cluster i for the j^{th} data set, and J is the number of data sets.

$$\forall j \in [1..J], \quad Q[j] = \sum_{n=1}^{N} \sum_{i=1}^{C} (s_{n,i}^j)^2 |x_n - \mu_i^j|^2 \tag{4.1}$$

- **Collaborative step:** Each site will exchange its partition matrix S or prototypes μ with the other sites. The FCM algorithm is then run again on each site with a modified version of the objective function that takes the shared information into account. In the case of horizontal collaboration, we use the objective function shown in Equation (4.2) where $\alpha_{j,l}$ is a weighting coefficient that describes the intensity of the collaboration between site j and site l. This coefficient $\alpha_{j,l}$ is usually positive and the matrix α is called the collaboration matrix. In the case of vertical collaboration, the objective function shown in Equation (4.3) is used. In this case $\beta_{j,l}$ is the weighting coefficient that describes the intensity of the collaboration between two sites j and l. This equation also makes the assumption that the clusters of the different algorithms have been mapped and aligned (i.e. Any i^{th} cluster of any site is equivalent to the i^{th} cluster of any other site).

$$\forall j \in [1..J], \quad Q^h[j] = \sum_{n=1}^{N} \sum_{i=1}^{C} (s_{n,i}^j)^2 |x_n - \mu_i^j|^2 + \sum_{\substack{l=1 \\ l \neq j}}^{J} \alpha_{j,l} \sum_{n=1}^{N} \sum_{i=1}^{C} (s_{n,i}^j - s_{n,i}^l)^2 |x_n - \mu_i^j|^2 \tag{4.2}$$

$$\forall j \in [1..J], \quad Q^v[j] = \sum_{n=1}^{N_j} \sum_{i=1}^{C} (s_{n,i}^j)^2 d_{ni}^2[j] + \sum_{\substack{l=1 \\ l \neq j}}^{J} \beta_{j,l} \sum_{n=1}^{N_j} \sum_{i=1}^{C} (s_{n,i}^j)^2 |\mu_i^j - \mu_i^l|^2 \tag{4.3}$$

The main limitation of this proposed approach is that it only enables Fuzzy C-Means algorithms to collaborate together, and furthermore requires that all FCM be looking for the same number of clusters.

The same approach was used to develop a collaborative version of the EM algorithm (CoEM), based on the same equations adapted to the Gaussian Mixture Model [16]. The Collaborative Fuzzy C-Means from Pedrycz was later also improved in the form of the Collaborative Fuzzy K-Means (CoFKM) algorithm [30]. Another collaborative EM-like algorithm was proposed by Hu et al. [67] using a Markov Random Field model with the goal of maximizing the likelihood of a combination of multiple clustering before a consensus.

All these algorithms display similar limitations: the number of clusters and the objective function must be identical in all sites, and the collaboration can only happen between instances of the same algorithm.

4.2.2 Prototype-based collaborative algorithms

The work of Pedrycz on the CoFC algorithm was also extended to be adapted to the Self-Organizing Maps (SOM) by Grozavu et al. [54, 57, 55] and Generative Topographic

Maps (GTM) by Ghassany et al. [50, 51] (More details on regular the SOM [84] and GTM [20] algorithms can be found in Section 6.2).

In [57], the classical SOM objective function is modified by adding to the original objective function a term $R^h[j]$ for horizontal collaboration -See Equation (4.4)- and $R^v[j]$ for vertical collaboration -See Equation (4.5)-. In these Equations, $\mathcal{K}_{j,l}(\cdot)$ is a distant neighborhood function defined as shown in Equation (4.6) where $\mathcal{K}^j_{\sigma(j,\mathcal{X}(x^j_n))}$ is the classical SOM neighborhood function (see Section 6.2.1, Equations (6.1) and (6.3) for more details).

$$\forall j \in [1..J], R^h[j] = \sum_{\substack{l=1 \\ l\neq j}}^{J} \alpha_{j,l} \sum_{n=1}^{N} \sum_{i=1}^{K} \mathcal{K}_{j,l}(x_n)|x_n - \mu^j_i|^2 \tag{4.4}$$

$$\forall j \in [1..J], R^v[j] = \sum_{\substack{l=1 \\ l\neq j}}^{J} \beta_{j,l} \sum_{n=1}^{N_j} \sum_{i=1}^{K} \mathcal{K}_{j,l}(x^j_n)|\mu^j_i - \mu^l_i|^2 \tag{4.5}$$

$$\mathcal{K}_{j,l}(x) = \left(\mathcal{K}^j_{\sigma(j,\mathcal{X}(x^j_n))} - \mathcal{K}^i_{\sigma(l,\mathcal{X}(x^l_n))}\right) \tag{4.6}$$

For the collaborative version of the GTM algorithm [51], the principle is the same with the M-Step of the EM algorithm mapping the neurons to the final clusters being modified by considering the collaborative term as a penalization term based on earlier works on the use of the EM algorithm for penalized likelihood estimation [53].

One problem with these two methods is that they do not really solve the main issue of collaboration between different types of algorithms. Furthermore, while the number of clusters does not matter in the case of the collaborative SOM and collaborative GTM, in both cases the maps must have the same number of neurons and be topologically close from each other. This is actually even more restraining than a requirement on the number of clusters. Additionally, both methods heavily rely on a user input *collaborative confidence parameter* (α or β) which defines the importance of the distant clustering partitions. In the case of unsupervised learning there is no available information on quality of the clusters and this parameter is often tricky to choose, which is a problem since the final results are very dependent on this confidence parameter.

4.2.3 The SAMARAH Method

The SAMARAH method proposed by Wemmert [147] belongs to a different family of collaborative clustering frameworks compared with the methods presented in the previous subsection. It is because of its differences and particular focus on collaborative clustering applied to satellite images that this method was grouped together with the other collaborative frameworks relatively late.

First, while the previous methods oculd be used for both horizontal and vertical objective functions, SAMARAH is a purely horizontal collaborative framework. The SAMARAH method can be applied to several clustering algorithms processing the same data elements, eventually with access to different views of the same elements [44]. The strength of SAMARAH is that this framework can work with any kind of hard clustering algorithm, and is not concerned with issues such as fitness functions, number of clusters, or prototypes. Its goal is very simple: given J clustering results for the same data, the idea is to modify these results in an iterative and collaborative way

with the aim of improving the quality of all results and reduce their diversity in order to make the finding of a consensus solution easier.

The SAMARAH method is therefore a complete framework that follows the 3 steps described in Figure 4.6:

- Local step: generation of the local results using any relevant clustering algorithm.s
- Collaborative step: refinement of the results using collaboration.
- Consensus step: aggregation of the refined results.

Once the results have been generated during the local step, the SAMARAH method maps the clusters of the different algorithms using probabilistic confusion matrices (PCM). Let C^i and C^j be two clustering results from two algorithms $\mathcal{A}^i$ and $\mathcal{A}^j$ looking for K_i and K_j clusters respectively. Then, the probabilistic confusion matrix $\Omega^{i,j}$ that maps the clusters from $\mathcal{A}^i$ to $\mathcal{A}^j$ is defined as shown bellow:

$$\Omega^{i,j} = \begin{pmatrix} \omega^{i,j}_{1,1} & \cdots & \omega^{i,j}_{1,K_j} \\ \vdots & \ddots & \vdots \\ \omega^{i,j}_{K_i,1} & \cdots & \omega^{i,j}_{K_i,K_j} \end{pmatrix} \text{ where } \omega^{i,j}_{a,b} = \frac{|c^i_a \cap c^j_b|}{|c^i_a|} \tag{4.7}$$

where c^i_a is a notation for the a^{th} cluster of algorithm $\mathcal{A}^i$.

The PCM $\Omega^{i,j}$ makes it possible to know whether or not the objects of two results have been grouped in a similar way, or if the two clustering results are dissimilar. The matrix has a key role in the comparison of two clustering results -such as detecting agreements and conflicts-, and has the major advantage of being independent from the clustering algorithm used to generate the results. We note that $\Omega^{i,j} \neq \Omega^{j,i}$ even when $K_i = K_j$.

In its collaborative step, the SAMARAH method uses this matrix to detect pairwise conflicts between the different partitions and solve them by order of perceived importance based on a conflict metric criterion [147]. There are 3 possible ways of solving a conflict by individually tweaking the clustering solutions:

- Splitting a cluster into several sub-clusters. The process is done by applying a simple clustering algorithm (such as the K-Means algorithm) on the data of the cluster to be split.
- Merging several clusters into a single one.
- Deleting a cluster: one cluster is deleted and its elements are attributed to one or several of the remaining clusters.

Once the solutions have all been refined and are consequently quite similar to each other, the SAMARAH method proceeds to aggregate them using a process similar to a majority vote [148].

If we sum up, the SAMARAH method has several strengths: it makes it possible for very different algorithms to collaborate together. It is a very complete framework that covers all 3 steps of local learning, collaborative learning and result aggregation. And it does not heavily rely on user input parameters.

On the weaknesses side: SAMARAH does not cover vertical collaboration. Furthermore, its conflict resolution system certainly is a weak point: it relies on a pairwise conflict criterion, and solves the conflict in an ordered asynchronous fashion, which

makes the process volatile and can lead to sub-optimal results. Finally, while it is also a strong point of the method, the fact that the algorithms parameters or prototypes do not play any role once the local step is over may constitute a weakness, in the sense that the local model is never rebuilt using the new partitions and does not play any active role in either the collaborative step or the consensus step. Whether this is a good thing or not is open to debate.

4.3 Horizontal probabilistic collaborative clustering guided by diversity

In this section we propose our own collaborative framework for horizontal collaboration that combines several advantages of the SAMARAH method -the fact that it can incorporate heterogeneous clustering algorithms with a different number of clusters using a PCM matrix-, while keeping the possibility to incorporate information from external algorithms directly into the local model -like Pedrycz's CoFC algorithm- thus according a role to the local model too during the collaboration process. This section is based on two articles [128, 127].

The theoretical basis of our work is close from the work of Bickel and Scheffer on the estimation of Mixture Models using Co-EM [17, 16]. Our proposed method differs from theirs in the following points: in our case we are treating a context broader than multi-view clustering. Our method makes it possible for algorithms from different families to work together, and once again we do not have the limitation that all algorithms should be searching for the same number of clusters.

In Chapter 6, we show how to use our proposed horizontal collaboration framework for vertical collaboration purposes.

4.3.1 Context and Notations

One general setting in which to cast the participating clustering algorithms is the probabilistic model-based framework. Many clustering techniques can indeed be depicted in this model, e.g. fuzzy C-Means, Gaussian mixtures models (GMM), mixtures of Bernouilli distributions, etc.

For example, in the case where one algorithm would follow a GMM with K clusters, we would have $\Theta = \{\theta_1, ..., \theta_K\}, \theta_c = \{\pi_c, \mu_c, \Sigma_c\}$ where μ_c is the mean value of the c^{th} cluster, Σ_c its covariance matrix and π_c the mixing probability.

If we want to be more generic, each clustering algorithm would assume that the random variable x is distributed according to a mixture of K components. Each of these components (i.e. clusters) c is mathematically represented by a parametric distribution θ_c. The data set can therefore be considered as a mixture of distributions on the samples:

$$p(X|\Theta) \;=\; \sum_{c=1}^{K} \pi_c \, p(X|\theta_c) \tag{4.8}$$

where $\pi_1, \ldots, \pi_K$ are the mixing probabilities estimated by the algorithm, and each θ_c is the set of parameters specifying the characteristics c^{th} component (e.g. mean value and covariance matrix), and $p(x|\theta_c)$ is the component distribution. The mixing

probabilities obey:

$$0 \leq \pi_c \leq 1 \quad (c = 1, \ldots, K), \quad \text{and} \sum_{c=1}^{K} \pi_c = 1$$

Classically, in regular probabilistic clustering, the goal is to estimate the parameters of the probabilistic model entertained by the algorithm when observing the data set X. This is generally done through a *maximum likelihood* estimation:

$$\Theta_{ML} = \underset{\Theta}{\text{Argmax}} \log p(X|\Theta) \tag{4.9}$$

the best estimates being the one that maximizes the probability of generating the observed data set.

If, in addition, there is a priori information $p(\Theta)$ about the parameters, this leads to the *maximum a posteriori* (MAP) estimation:

$$\Theta_{MAP} = \underset{\Theta}{\text{Argmax}} \log p(X|\Theta) + \log p(\Theta) \tag{4.10}$$

Assuming that the observations in X are independent, the estimation problem becomes:

$$\Theta_{MAP} = \underset{\Theta}{\text{Argmax}} \log \left[\sum_{n=1}^{N} p(x_n|\Theta) \right] + \log p(\Theta) \tag{4.11}$$

Because, in clustering, the membership of each data element x_n to the clusters c (latent variables) is unknown, maximizing the above log-likelihood is a complex problem. An elegant and powerful method has been designed for finding solutions to such problems with latent variables: the Expectation-Maximizarion (EM) algorithm [38].

The algorithm starts with a guess about the parameters θ_c of the components as well as with the mixing probabilities π_c. Then, it alternates between two updating steps. In the *expectation* step (E-step), the current parameter values $(\theta_c, \pi_c)_{c=1,\ldots,K}$ are used to compute the responsibilities $s_{n,c}$ (i.e. posterior probabilities) of the clusters (latent variables) for each data element x_n. In the *Maximization* step (M-step), the log-likelihood is maximized by optimizing the auxiliary function thus leading to the updated responsibilities, leading to updated estimations about the parameters and the mixing probabilities. It is proved that the algorithm converges towards a local maximum of the log-likelihood.

The final partition (clustering) produced by an algorithm depends on its probabilistic model Θ (e.g. mixture of K Gaussian distributions in $\mathbb{R}^d$) and on the starting parameter values of the run: θ_c, π_c (for all $c \in \{1, \ldots, K\}$).

4.3.1.1 Adaptation to the case of horizontal collaboration

In horizontal collaborative clustering we consider a group of algorithms that are working on the same data elements, albeit possibly on different views of these data. To this end, let $X = \{x_1, ..., x_N\}, x_n \in \mathbb{R}^d$ be a data set containing N elements, each of them with d features, $d \in \mathbb{R}$. Let $\boldsymbol{\mathcal{A}} = \{\mathcal{A}^1, ..., \mathcal{A}^J\}$ be a system of J clustering algorithms working on this data set. Each clustering algorithm will have its own parameters Θ^j to describe either the clusters or its model, and will produce its own clustering partition C^i made of K_i clusters, based on the features of the data set it has access to $X^i \subseteq X$. The solutions C^i are computed for the data X using parameters Θ^i. In the case of hard

clustering C^i can be translated into a solution vector of size N, and for fuzzy clustering into a matrix of size $N \times K_i$. We call this later matrix S^i.

$$S^i = \begin{pmatrix} s^i_{1,1} & \cdots & s^i_{1,K_i} \\ \vdots & \ddots & \vdots \\ s^i_{N,1} & \cdots & s^i_{N,K_i} \end{pmatrix} \tag{4.12}$$

The solutions S^i output by the algorithms are therefore two-dimensional matrices of size $N \times K_i$ where each element $s^i_{n,c}$ expresses the responsibility given by algorithm $\mathcal{A}^i$ to a cluster c for the data element x_n. In soft clustering we have $s^i_{n,c} \in (0,1)$, while in hard partitioning $s^i_{n,c} \in \{0,1\}$ and $s^i_{n,c} = 1$ for only one cluster c for each data element x_n. In this last case, we will use $s^i_n = c$ to express the fact that data element x_n has been (entirely) allocated to cluster c by algorithm $\mathcal{A}^i$ (i.e. $s^i_{n,c} = 1$).

Thus we have : $\forall i$, $\mathcal{A}^i = \{X^i, \Theta^i, S^i, K_i\}$.

In order to simplify the notation, we will most often use X and x_n to describe the data set and the data indifferently for all algorithms, but the differences in viewpoints should still be kept in mind.

For regular probabilistic-based clustering algorithms, the Equation to maximize was the following:

$$(\Theta_{opt}, S_{opt}) = \underset{(\Theta,S)}{\text{Argmax}}\, P(X, S|\Theta) \tag{4.13}$$

However, in the context of collaborative clustering, the search for the local parameters Θ^i and the associated solutions S^i must depend on the solutions and eventual prototypes of the other algorithms. Furthermore, not all collaborating algorithms may be probabilistic or prototype-based. Therefore, the probabilistic notations and probabilistic context cannot be used anymore. Therefore, the problem to solve for a given algorithm $\mathcal{A}^i$ becomes the following:

$$S^i_{opt} = \underset{S^i}{\text{Argmax}}\, f(X^i, \mathbf{S}, \mathbf{\Theta}) \tag{4.14}$$

Where $f(\cdot)$ is an objective function that considers the local visible data X^i and all the solutions $\mathbf{S}$ from the other algorithms as well as their eventual parameters $\mathbf{\Theta}$. This function is detailed in the next section, as well as the optimization process to optimize it.

4.3.2 Algorithm

In this section, we study the modifications made on the algorithms used during the local step so that they can use information from the other collaborators during the collaborative step.

4.3.2.1 Objective function

We now describe the modification of the current partition $S^i|_t$ of algorithm $\mathcal{A}^i$ at an iteration t depending on the information received from the other collaborators. The fundamental question is the following: Given that $\mathcal{A}^i$ has computed a partition $S^i|_t$, and knowing that the other algorithms have computed their own partitions, how do we bias S^i so that $S^i|_{t+1}$ take into account both the local information and the information from the other collaborators?

Since we consider a case where we have different algorithms, potentially looking for a different number of clusters, with different models (prototype-based on not prototype-based), it is impossible for these algorithms to collaborate by exchanging their prototypes or parameters. However, like in the SAMARAH method, they can collaborate based on their solutions S^i.

The idea of our method is to optimize a modified fitness function that considers both the local partition and parameters, and the information coming from the other algorithms solutions. From the point of view of a given algorithm $\mathcal{A}^i$, during the collaborative step this can be translated into Equation (4.15), where $s^i_{n,c}|_{t+1}$ is the new responsibility given locally to cluster c by algorithm $\mathcal{A}^i$ for the data element x^i_n after taking into account the solutions of the other algorithms, $s^i_{n,c}$ denotes the responsibility given locally to cluster $c \in [1..K_i]$ by algorithm $\mathcal{A}^i$ depending on its parameters Θ^i, λ_i is a parameter that controls the degree to which each algorithm accepts information from the other collaborators, $\mathcal{Q}$ is the ensemble of all possibles combinations of clusters that can be entertained by the different algorithms for a given data x_n, $\boldsymbol{q} = \{q_1, ..., q_J\}, q_j \in [1..K_j]$ one of this possible combination of clusters, and $g^i(\boldsymbol{q}, c)$ is a combination function chosen to assess the likelihood of having $q_i = c$ given the other clusters in the combination $\boldsymbol{q}$. In Equation (4.15), for any given data x_n, we consider all the possible combinations $\boldsymbol{q} \in \mathcal{Q}$ so that $q_i = c$.

$$s^i_{n,c}|_{t+1} = (1 - \lambda_i) \cdot s^i_{n,c}(\Theta^i|_t) + \lambda_i \sum_{\boldsymbol{q} \in \mathcal{Q} | q_i = c} \left(g^i(\boldsymbol{q}, c) \cdot \prod_{\substack{j=1 \\ j \neq i}}^{J} s^j_{n,q_j}(\Theta^j|_t) \right) \tag{4.15}$$

Equation (4.15) can be interpreted as follows: the first term $(1 - \lambda_i) \cdot s^i_{n,c}(\Theta^i|_t)$ is a *local term* weighted by the parameter λ_i. It gives the likelihood for element x_n to be linked to a cluster c based only on the local parameters of the algorithm $\mathcal{A}^i$. The function $s^i_{n,c}(\Theta^i|_t)$ is a local term computed using Equation (4.16). In the case where the local algorithm is not prototype based, it is directly equal to the responsibility during the previous iteration. Otherwise, it is a recomputed responsibility based on the local model and the prototypes Θ^i. In this case $s^i_{n,c}(\Theta^i|_t)$ may be slightly different from $s^i_{n,c}|_t$. With these probabilistic models, we mostly have $P(s^i_n = c|x^i_n, \Theta^i|_t) \propto P(x^i_n|\theta^i_c)$.

We want to highlight that in Equation (4.15), $s^i_{n,c}|_{t+1}$ is a term from the solution matrix that considers both local and collaborative terms, while $s^i_{n,c}(\Theta^i|_t)$ with a parameter is a responsibility function computed only from the local algorithm $\mathcal{A}^i$ using the parameters Θ^i if any.

$$s^i_{n,c}(\Theta^i|_t) = \begin{cases} s^i_{n,c}|_t \text{ if } \Theta^i = \emptyset \\ P(s^i_n = c|x^i_n, \Theta^i|_t) \text{ otherwise} \end{cases} \tag{4.16}$$

The second term of Equation (4.15) is called a *collaborative term* that evaluates the likelihood for the element x_n to be linked to a cluster c based on the other algorithms partitions and their choice of cluster for the same data x_n, weighted by λ_i. To do so, in the case of fuzzy clustering this likelihood must be evaluated for all possible combinations of clusters where $q_i = c$ -with $g^i(\boldsymbol{q}, c)$- and is weighted based on the likelihood of this combination to occur for the data x_n assuming that all algorithms are independent from each other - hence the $\prod_j s^j_{n,q_j}(\Theta^j|_t)$-.

This collaborative term can be interpreted as follows: for each cluster combination $\boldsymbol{q} \in \mathcal{Q}$, $g^i(\boldsymbol{q}, c)$ assesses the likelihood of having $q_i = c$ with this combination $\boldsymbol{q}$. Then

$g(\cdot)$ is weighted by the likelihood of such combination with the other collaborators $\mathcal{A}^j \neq \mathcal{A}^i$ for the data x_n. The sum over all the $\boldsymbol{q} \in \mathcal{Q}$ makes it possible to take into account the fuzziness of the solutions by summing up the likelihood of having $q_i = c$ for all possible combinations.

Equation (4.15) can be rewritten in the following probabilistic form:

$$s^i_{n,c}|_{t+1} = (1-\lambda_i) \cdot P(s^i_n = c|x^i_n, \Theta^i_t) + \lambda_i \sum_{\boldsymbol{q} \in \mathcal{Q}|q_i=c} \Big[P(q_i = c|\boldsymbol{q}, \boldsymbol{S}_t) \cdot P(\boldsymbol{q}|n, \boldsymbol{S}_t \backslash S^i) \Big] \tag{4.17}$$

$$= (1-\lambda_i) \cdot P(s^i_n = c|x^i_n, \Theta^i_t) + \lambda_i \cdot P(s^i_n = c|\boldsymbol{S}_t) \tag{4.18}$$

For hard clustering or when $\mathcal{Q}$ contains too many combinations ($\bar{K}^J$ on average) for Equation (4.15) to be computed in a reasonable time for all the data, a semi-hard version is given in Equation (4.19) where $\boldsymbol{q}^{max_n}$ is the most likely combination of clusters for x_n so that $\boldsymbol{q}^{max_n} \in \mathcal{Q}$, $q_i = c$ and $\forall j \neq i, q_j = argmax\left(s^j_{n,q_j}(\Theta^j|_t)\right)$. Since Equation (4.19) considers only the most likely combination of cluster for each element x_n instead of all possible combinations, it can be seen as an approximation of Equation (4.15) that is much easier to compute. With this approximation the product $\prod_{j\neq i}^{J} s^j_{n,q_j}(\Theta^j|_t)$ from Equation (4.15) is different from 0 only when $\boldsymbol{q} = \boldsymbol{q}^{max_n}$, and is therefore removed from Equation (4.19).

$$s^i_{n,c}|_{t+1} = (1-\lambda_i) \cdot s^i_{n,c}(\Theta^i|_t) + \lambda_i \cdot g^i(\boldsymbol{q}^{max_n}, c) \tag{4.19}$$

4.3.2.2 Combination functions

For Equations (4.15) and (4.19) to be correct, the function $g(\cdot)$ needs to be chosen so that it has specific properties that can be summed up in a set of axioms:

Axiom-1. $g^i(\boldsymbol{q}, c)$ needs to increase strictly between 0 and 1 when the consensus between the different algorithms grows on the likelihood of having $q_i = c$ for a given combination q.

Axiom-2. $g^i(\boldsymbol{q}, c)$ needs to be normalized so that for any cluster combination $\boldsymbol{q}$ that occurs at least once, we have: $\sum_{c\in[1..K_i]} g^i(\boldsymbol{q}, c) = 1$.

Axiom-3. When the algorithms have the exact same partitions and $c = argmax_{q_i}(s^i_{n,q_i})$, then: $g^i(\boldsymbol{q}^{max_n}, c) = 1$.

To be more precise on the computation and increase property of g, let i be a fixed algorithm, c a fixed cluster, and $\boldsymbol{q}$ be a fixed cluster combination such that $q_i = c$. The value $g^i(\boldsymbol{q}, c)$ is computed by considering $\boldsymbol{S}$ the set of all partitions, in the following way: we compute the likelihood of $q_i = c$ with respect to all the other $q_j, j \neq i$ based on a given partition $S \in \boldsymbol{S}$. This likelihood is computed directly from the cardinal of the intersections of all involved clusters. The increase property should be understood in the following way: $g^i(\boldsymbol{q}, c)$ defines a function $\tilde{g} : \boldsymbol{S} \to \mathbb{R}$ in which for each $S \in \boldsymbol{S}$ we associate the value $g^i(\boldsymbol{q}, c)$ computed as explained above. This function $\tilde{g}$ increases relatively fast with respect to the consensus between the different algorithms that $q_i = c$ is likely a good choice.

To better understand what $g(\cdot)$ is all about, let's consider a simple example of hard partitioning: in Figure 4.9 we show a simple case with three partitions of data sample

X by three algorithms. In this example, the data element $\mathbf{x}_n$ has been put in cluster a' by algorithm $\mathcal{A}^{red}$, in cluster b'' by algorithm $\mathcal{A}^{blue}$ and in cluster c' by algorithm $\mathcal{A}^{green}$.

In this very simple example, we want to analyze whether or not the collaborative term would push algorithm $\mathcal{A}^{red}$ to change the cluster it linked to x_n depending on the partitions produced by $\mathcal{A}^{blue}$ and $\mathcal{A}^{green}$.

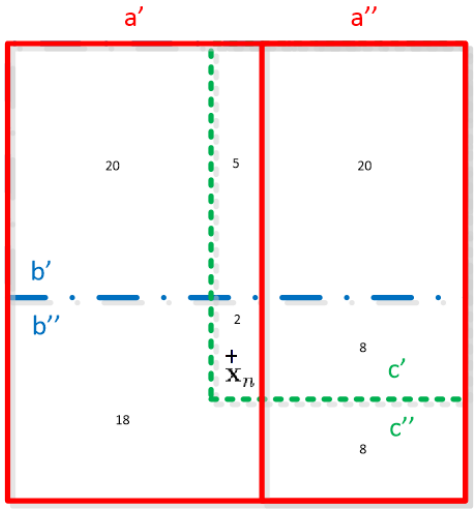


Figure 4.9: Illustration of multiple partitions of a data set X on a grid of size 9×9 by 3 algorithms.

During the course of this PhD thesis, we found several possible combination functions $g(\cdot)$ abiding by the axioms exposed before, and that all have different strengths and weaknesses. Three of them are shown in Equations (4.20), (4.21) and (4.22).

$$g^i_{\cap}(\boldsymbol{q}, c) = \frac{|\bigcap_{j \neq i} q_i \cap q_j|}{|\bigcap_{j \neq i} q_j|}, \quad q_i = c \tag{4.20}$$

The formula from Equation (4.20) assesses consensus between the local algorithm and the other algorithms divided by the consensus between the other algorithms. Plus this combination function is already normalized. However, it is costly to compute due to the K^J possible intersections. It is also worthy to mention that this combination function does not allow to weight the influence of the different algorithms.

$$g^i_{+}(\boldsymbol{q}, c) = \frac{1}{Z} \sum_{j \neq i} \tau_{j,i} \frac{|q_i \cap q_j|}{|q_j|} = \frac{1}{Z} \sum_{j \neq i} \tau_{j,i} \cdot \omega^{j,i}_{q_j, q_i}, \quad q_i = c \tag{4.21}$$

In Equation (4.21) we compute the mean pairwise consensus between the partitions, and in (4.22) the geometric mean consensus. In both Equations, the $\tau_{j,i}$ are weights that can be set to different values in order to change the influence of the algorithms of each other, and Z is a normalization constant that is needed to respect axiom 2. Both equations are based on the same PCM Matrices from the SAMARAH method described in Equation (4.7) and which are relatively cheap to compute. Beyond the fact that both combination functions require a normalization, g_* also has the issue that it always returns 0 whenever one of the intersection is empty.

$$g^i_{*}(\boldsymbol{q}, c) = \frac{1}{Z} \prod_{j \neq i} \left(\frac{|q_i \cap q_j|}{|q_j|} \right)^{\tau_{j,i}} = \frac{1}{Z} \prod_{j \neq i} \left(\omega^{j,i}_{q_j, q_i} \right)^{\tau_{j,i}}, \quad q_i = c \tag{4.22}$$

Given that all 3 combination functions have their pros and cons, picking one is context dependent. For instance, $g_\cap$ certainly is the most interesting one to have a global consensus combination function, but should be avoided when using a large number of collaborators due to its complexity, and it is unpractical when weighting the collaborators is a requirement. On the other hand, g_+ and g_* exhibit a somewhat similar behavior than $g_\cap$ and scale better with a large number of collaborators. However, g_* may behave in an erratic way with collaborators the diversity of which is too big (because of many null intersections), and may require to be used in its logarithm form when the number of collaborators increases.

$g_\cap^{red}(\boldsymbol{q}, a') = \frac{\lvert a' \cap b'' \cap c' \rvert}{\lvert b'' \cap c' \rvert} = \frac{2}{10} = 0.2$
$g_\cap^{red}(\boldsymbol{q}, a'') = \frac{\lvert a'' \cap b'' \cap c' \rvert}{\lvert b'' \cap c' \rvert} = \frac{8}{10} = 0.8$
$g_+^{red}(\boldsymbol{q}, a') = \frac{1}{Z}\left(\frac{\lvert a' \cap b'' \rvert}{\lvert b'' \rvert} + \frac{\lvert a' \cap c' \rvert}{\lvert c' \rvert}\right) = \frac{1}{Z}\left(\frac{20}{36} + \frac{7}{35}\right) \approx 0.38$
$g_+^{red}(\boldsymbol{q}, a'') = \frac{1}{Z}\left(\frac{\lvert a'' \cap b'' \rvert}{\lvert b'' \rvert} + \frac{\lvert a'' \cap c' \rvert}{\lvert c' \rvert}\right) = \frac{1}{Z}\left(\frac{16}{36} + \frac{28}{35}\right) \approx 0.62$
$g_*^{red}(\boldsymbol{q}, a') = \frac{1}{Z}\left(\frac{\lvert a' \cap b'' \rvert}{\lvert b'' \rvert} \times \frac{\lvert a' \cap c' \rvert}{\lvert c' \rvert}\right) = \frac{1}{Z}\left(\frac{20}{36} \times \frac{7}{35}\right) \approx 0.24$
$g_*^{red}(\boldsymbol{q}, a'') = \frac{1}{Z}\left(\frac{\lvert a'' \cap b'' \rvert}{\lvert b'' \rvert} \times \frac{\lvert a'' \cap c' \rvert}{\lvert c' \rvert}\right) = \frac{1}{Z}\left(\frac{16}{36} \times \frac{28}{35}\right) \approx 0.76$

Table 4.2: Example of results for different combination functions

For comparison purposes, in Table 4.2 we show the numerical results returned by the 3 combination functions in the case of the easy problem shown in Figure 4.9 with $\tau_{j,i} = 1 \quad \forall (i,j)$. In this example, the data x_n has been assigned to b'' by $\mathcal{A}^{blue}$ and c' by $\mathcal{A}^{green}$. Therefore, we are weighting the likelihood of $\boldsymbol{q} = \{a', b'', c'\}$ against $\boldsymbol{q} = \{a'', b'', c'\}$.

As one can see, based on the cluster surface area, all three combination functions agree that the collaborative term would encourage the red algorithm $\mathcal{A}^{red}$ to put the data x_n into a'' instead of a'.

4.3.2.3 Algorithm

An elegant way to maximize Equation (4.15) or Equation (4.15), and thus to increase the likelihood of the solutions, is to use a parallelized version of the EM algorithm [38]. The general framework of our collaborative algorithm is shown in Algorithm 9. This algorithm includes the local step where each algorithm works on its own, and the collaborative step which is equivalent to a meta-EM algorithm.

We now want to analyze the computational complexity of the collaborative step. We consider a data set of size $N \times d$, J collaborators looking for an average of K clusters, and that the collaborative step will do around I iterations before convergence.

Using Equation (4.15), we have a complexity of $I \times J^2 \times N \times K^J \times cpx(g) \times d^2$. As one can guess, this is way too much to be practical with almost any type of data set. With Equation (4.19), the complexity becomes: $I \times J^2 \times N \times cpx(g) \times K \times d^2$.

All three functions $g(\cdot)$ have a complexity of $J \times N$, therefore with Equation (4.19) the final complexity of our proposed algorithm is $I \times J^3 \times N^2 \times K \times d^2$. However, in the case of g_+ and g_*, the complexity can be reduced to $I \times J^2 \times N \times K \times d^2$ by computing the PCM $\Omega^{i,j}$ once at the beginning of each iteration and storing them in an array of size $J^2 \times K^2$ instead of computing g_+ and g_* on the flight. However, this is not an option with the $J \times K^{J-1}$ possible combinations that would have to be stored for $g_\cap$.

Algorithm 9: Horizontal Probabilistic Collaborative Clustering Guided by Diversity

Local step:
forall clustering algorithms $\mathcal{A}^i$ **do**
  Apply $\mathcal{A}^i$ on the data X^i.
  $\rightarrow$ Learn the local parameters Θ^i
  $\rightarrow$ Compute $s^i_{n,c}$ for all data point x_n and all clusters c considered by $\mathcal{A}^i$
end
Collaborative step:
Set the λ_i
$t = 0$
while the system's global entropy $\mathcal{H}$ is not stable **do**
  Meta E-Step:
  forall clustering algorithms $\mathcal{A}^i$ **do**
    forall $x^i_n \in X^i$ **do**
      Assess $s^i_{n,c}|_{t+1}$ using Equation (4.15) or (4.19)
    end
  end
  Meta M-Step:
  forall clustering algorithms $\mathcal{A}^i$ **do**
    Update the local parameters Θ^i -if any, and when it is relevant- by using a maximum likelihood estimation:

$$\Theta^i|_{t+1} = \underset{\Theta}{\text{Argmax}}\, p(X^i|\Theta) \tag{4.23}$$

  end
  $t++$
end
Return $\boldsymbol{S}_{t-1}$

4.3.2.4 Stopping Criterion

Earlier works have shown the importance of diversity in collaborative clustering ([6, 56]). Intuitively, the way our proposed method works is that it tries to reduce the diversity between the algorithms' solutions. Therefore, studying the variation of diversity between the different algorithms appeared to be the best stopping criterion.

We want to mention that because of the way it was designed based on the clusters intersections, our method tends to quickly reduce the diversity between solutions that were already similar. This phenomenon can result in a "*meta-clustering*" of the collaborators where groups of algorithms with similar results will quickly have a low diversity between their solutions, while the cross-diversity between solutions of different groups will tend to remain relatively high.

There are several common methods to assess the diversity of two algorithms results. In the case of our framework, we needed a diversity measure that meets several requirements: (1) Fast to compute, as we need to assess the diversity of multiple algorithms at the end of each iteration of our collaborative framework. (2) Capable of comparing results that have a different number of clusters. (3) Ideally matrix-based and asym-

metrical, since we already use probabilistic correspondence matrices that happen to be asymmetrical in our collaborative algorithm.

Given these criterion, we quickly ruled out pairwise-counting based methods such as the rand index [108] because of their quadratic complexity. We then considered mutual information based methods [123], as well as matrix based methods from ensemble learning [70]. We ultimately decided to do a mix of both by using a mutual information based method computed from the intersection of the different algorithms clusters. To do so, we adapted the *probabilistic confusion entropy* formula [143, 145] to our problem. We later found out that another team working on collaborative clustering had a similar idea roughly at the same time than us [154].

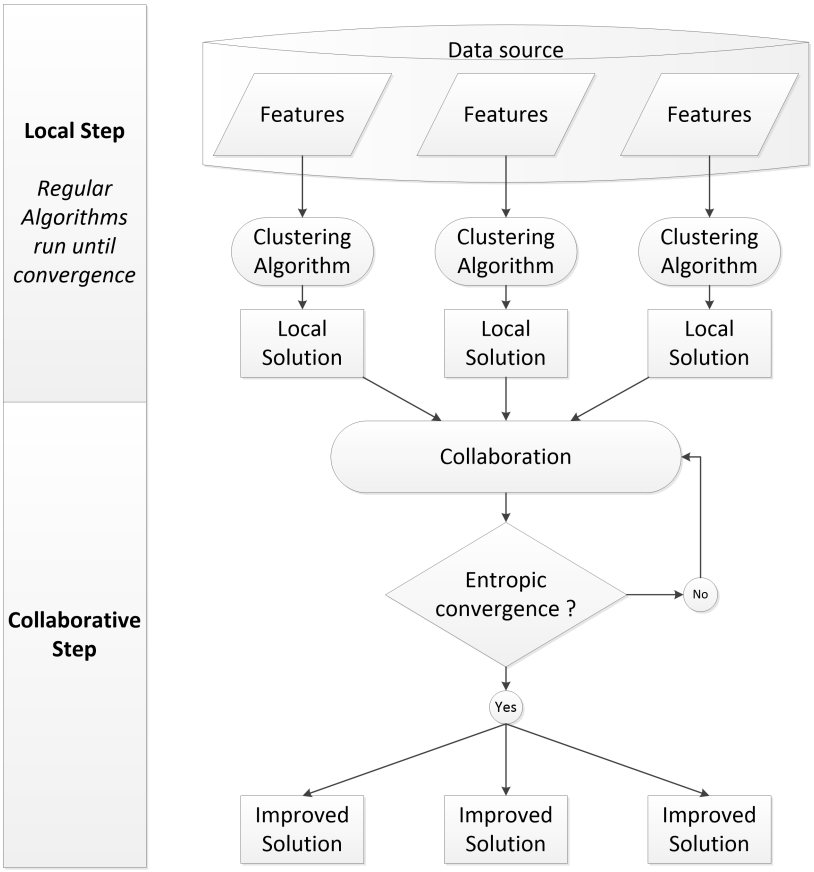


Figure 4.10: Horizontal probabilistic clustering guided by diversity

The global entropy mentioned in Algorithm (9) is described in Equation (4.24) bellow:

$$\mathcal{H} = \sum_{i=1}^{J} \sum_{j \neq i}^{J} \frac{-1}{K_i \times \ln(K_j)} \sum_{l=1}^{K_i} \sum_{m=1}^{K_j} \frac{|c_l^i \cap c_m^j|}{|c_l^i|} \ln\left(\frac{|c_l^i \cap c_m^j|}{|c_l^i|}\right) \tag{4.24}$$

In Equation (4.24), the notation c_l^i refers to the l^{th} cluster of algorithm $\mathcal{A}^i$, $|c_l^i|$ is the number of data in this cluster, and $|c_l^i \cap c_m^j|$ is the number of data linked to the l^{th} cluster of $\mathcal{A}^i$ and the m^{th} cluster of $\mathcal{A}^j$ at the same time. Using the PCM matrix notation from Equation (4.7), Equation (4.24) becomes:

$$\mathcal{H} = \sum_{i=1}^{J} \sum_{j \neq i}^{J} \frac{-1}{K_i \times \ln(K_j)} \sum_{l=1}^{K_i} \sum_{m=1}^{K_j} \omega_{l,m}^{i,j} \ln(\omega_{l,m}^{i,j}) \tag{4.25}$$

Since the global entropy is updated after each iteration of the collaborative step, if the entropy remains stable between two iterations, it means that the algorithms are not learning from each other anymore. As a consequence, the global entropy seems like a valid stopping criterion.

A schematic version of the complete algorithm is shown in Figure 4.10.

4.3.2.5 Parameters

We now want to discuss the role of the weighting parameter λ_i in Equations (4.15) and (4.19). As we mentioned earlier, this is a user input parameter that for a given algorithm $\mathcal{A}^i$ determines the balance between the weight the local algorithm model and the weight of the collaborative term. Examples of possibles set-ups for this parameters and their effects are given thereafter:

$\lambda_\mathbf{i} = \mathbf{0}$ In this case, no collaboration happens at all, and our proposed framework is Equivalent to running all the algorithms in parallel without any interaction.

$\lambda_\mathbf{i} = \mathbf{0.5}$ In this case, the local term is given the same importance than the collaborative term. It means, that the local model weights as much as all combined advises from the collaborators. This set-up is interesting when using model-based fuzzy clustering algorithms, but is unfit for hard clustering algorithms because the collaborative term will never weight enough to change anything.

$\lambda_\mathbf{i} = \mathbf{1} - \frac{\mathbf{1}}{\mathbf{J}}$ This set-up gives an equal weight to all the algorithms and collaborators. It means that the local term weights $\frac{1}{J}$ and so do each vote from every single collaborator. This is a good set-up for hard clustering algorithms.

$\lambda_\mathbf{i} = \mathbf{1}$ In this case, the local term is completely removed. With this set-up, our framework is equivalent to the collaborative step of the SAMARAH method but with a simultaneous parallel conflict resolution system. While this set-up may seem a bit extreme, it works very well with non model-based clustering algorithms.

We now briefly discuss the choice of the $\tau_{j,i}$ parameters to weight the influence of the algorithms inside the collaborative term in this section. There are three possible scenarii:

- We know from an expert which algorithms can help each other or not and set-up the $\tau_{j,i}$ manually accordingly.
- Without any prior knowledge on the algorithms and there results, all $\tau_{j,i}$ are given the same value (1 for instance).
- The values for the $\tau_{j,i}$ are computed to enhance the collaboration process. Examples of such optimization of the $\tau_{j,i}$ are shown in Chapter 5.

4.3.3 Convergence Properties

We are concerned here with convergence results obtained under conditions that are applicable to many practical situations.

Let $L(X, \mathbf{S}, \mathbf{\Theta})$ given by Equation (4.26) be the global likelihood function, defined from Equation (4.15), over a system with J collaborating algorithms. As it is defined in Algorithm 9, our proposed framework tries to maximize this likelihood function.

$$L(X, \mathbf{S}, \mathbf{\Theta}) = \sum_{i=1}^{J}\sum_{n=1}^{N}\sum_{c=1}^{K_i}(1-\lambda_i)\cdot s_{n,c}^{i}(\Theta^i) + \lambda_i \sum_{\boldsymbol{q}\in\mathcal{Q}|q_i=c}\left(g^i(\boldsymbol{q},c)\cdot\prod_{\substack{j=1\\ j\neq i}}^{J} s_{n,q_j}^{j}(\Theta^j)\right) \quad (4.26)$$

In Equation (4.26), $\mathbf{\Theta} = \{\Theta^1, ...\Theta^J\}$ is a set containing all the parameters from all the algorithms.

This global likelihood has two components: $L(X, \mathbf{S}, \boldsymbol{\theta}) = \mathcal{L}(X, \mathbf{S}, \mathbf{\Theta}) + \mathcal{C}(X, \mathbf{S}, \mathbf{\Theta})$, where $\mathcal{L}(X, \mathbf{S}, \mathbf{\Theta})$ is the local term of the likelihood and $\mathcal{C}(X, \mathbf{S}, \mathbf{\Theta})$ is the collaborative term. More precisely, we have:

$$\mathcal{L}(X, \mathbf{S}, \mathbf{\Theta}) \;=\; \sum_{i=1}^{J}\sum_{n=1}^{N}\sum_{c=1}^{K_i}(1-\lambda_i)\cdot s_{n,c}^{i}(\Theta^i) \quad (4.27)$$

and

$$\mathcal{C}(X, \mathbf{S}, \mathbf{\Theta}) \;=\; \sum_{i=1}^{J}\sum_{n=1}^{N}\sum_{c=1}^{K_i}\lambda_i \sum_{\boldsymbol{q}\in\mathcal{Q}|q_i=c}\left(g^i(\boldsymbol{q},c)\cdot\prod_{\substack{j=1\\ j\neq i}}^{J} s_{n,q_j}^{j}(\Theta^j)\right) \quad (4.28)$$

The proposed method, described in Algorithm (9), can be seen as some kind of meta-EM algorithm trying to optimize Equation (4.26). In Algorithm (10), we demonstrate how our algorithm can be considered as a meta-EM algorithm. Please note the presence of the $\mathbf{S}|\mathbf{S}_t$ in the Meta E-Step. This is due to the fact that when the S_{t+1}^i are locally computed in Algorithm (9), their value in the collaborative term depends on the solutions or parameters of the previous iteration -$\mathbf{S}_t$ and $\mathbf{\Theta}_t$- as explained in Equation (4.16).

Algorithm 10: Collaborative "Meta-EM"

Initialize, $t = 0$ and all $\mathbf{S}_0$ and $\mathbf{\Theta}_0$ with the local step
while the system global entropy $\mathcal{H}$ is not stable **do**
  Meta E-Step: $\mathbf{S}_{t+1} = \arg\max_S L(X, \mathbf{S}|\mathbf{S}_t, \mathbf{\Theta}_t)$ using Equation (4.15) or (4.19)
  Meta M-Step: $\mathbf{\Theta}_{t+1} = \arg\max_{\mathbf{\Theta}} \mathcal{L}(X, \mathbf{S}_{t+1}, \mathbf{\Theta})$ using Equation (4.23)
  $t++$
end
Return $\boldsymbol{S}_{t-1}$

Let $\mathcal{L}^i(X^i, S^i)$ be the local likelihood function of algorithm $\mathcal{A}^i$ summed over all the data and all the clusters, and $\mathcal{C}^i(X^i, S^i)$ its collaborative likelihood function. Note that Θ^i and S^i can be interchanged in the equations since one can be computed from the other. However, because of non prototype-based algorithms, we prefer to use S^i.

We put ourselves under the following hypotheses:

- We consider that at the beginning of the collaborative step, all local terms are optimized: at $t = 0, \forall i, S_0^i = \arg\max_S \mathcal{L}^i(X^i, S)$.
- We suppose that all algorithms $\mathcal{A}^i$ either have access to all attributes of the data X, or that the attributes subsets $X^i \subset X$ to which they have access have *"similar distributions"*.

Given two algorithms $\mathcal{A}^i$ and $\mathcal{A}^j$, then the two subsets X^i and X^j are said to have *"similar distributions"* if and only if for any diversity measure $D(S^i, S^j)$ the following condition is true:

$$\begin{aligned} h_i : \mathcal{L}^i(X^i, S^i_a) &\geq \mathcal{L}^i(X^i, S^i_b) \\ \Delta_{ij}(a,b) = D(S^i_a, S^j_a) &- D(S^i_b, S^j_b) \\ \forall \eta > 0, \exists \epsilon \geq 0 \text{ small enough, so that } P(\Delta_{ij}(a,b) < \eta | h_i \wedge h_j) &\geq 1 - \epsilon \end{aligned} \tag{4.29}$$

where $\Delta_{ij}(a,b)$ compares the difference in diversity between two solutions S^i and S^j at two given states a and b (which could be for instance 2 different iterations).

This can be translated as follows: *Two algorithms are said to have similar distributions when an increase in the likelihood (local) of both of them induces a reduction in the diversity of their solutions.*

With these two hypotheses, we now consider the Meta E-Step of our proposed framework and give the following propositions:

Proposition 1. *On average, optimizing the global local term $\mathcal{L}$ will result in increasing the value of the global collaborative term $\mathcal{C}$.*

Proof. From Equation (4.27), we have:

$$\mathcal{L}_{t+1} \geq \mathcal{L}_t \Leftrightarrow \sum_{i=1}^{J}\sum_{n=1}^{N}\sum_{c=1}^{K_i} s^i_{n,c}(\Theta^i_{t+1}) \geq \sum_{i=1}^{J}\sum_{n=1}^{N}\sum_{c=1}^{K_i} s^i_{n,c}(\Theta^i_t) \tag{4.30}$$

Then, because of the hypothesis that all distributions are similar from Equation (4.29), we know that when the local term increases, for any given similarity measure, the similarity between the solutions of the collaborative process increases with a very high probability. We also know from the first Axiom of section 4.3.2.2 that $g(\cdot)$ behaves like a similarity measure (which is the complementary of a diversity measure). Therefore we have the following property with a very high probability:

$$\mathcal{L}_{t+1} \geq \mathcal{L}_t \implies \sum_{i=1}^{J}\sum_{n=1}^{N}\sum_{c=1}^{K_i}\sum_{\boldsymbol{q}\in\mathcal{Q}|q_i=c} g^i(\boldsymbol{q}, c)_{t+1} \geq \sum_{i=1}^{J}\sum_{n=1}^{N}\sum_{c=1}^{K_i}\sum_{\boldsymbol{q}\in\mathcal{Q}|q_i=c} g^i(\boldsymbol{q}, c)_t \tag{4.31}$$

And then, given the definition of $\mathcal{C}$ in Equation (4.28), using Equation (4.30) to remove the product, with Equations (4.29) and (4.31), we have:

$$\forall \eta > 0, \exists \epsilon \geq 0 \text{ so that } P\left(\mathcal{L}_{t+1} \geq \mathcal{L}_t \Rightarrow (\mathcal{C}_{t+1} - \mathcal{C}_t) > \eta\right) \geq 1 - \epsilon \tag{4.32}$$

□

Corollary 1. *Under the same hypotheses that in Proposition 1, we have that it is unlikely to have a situation where the local term increases and the collaborative one decreases.*

Proof.

$$\forall \eta > 0, \exists \epsilon \geq 0 \text{ so that } P\left(\mathcal{L}_{t+1} \geq \mathcal{L}_t \Rightarrow (\mathcal{C}_{t+1} - \mathcal{C}_t) > \eta\right) \geq 1 - \epsilon \tag{4.33}$$

$$\Leftrightarrow \forall \eta > 0, \exists \epsilon \geq 0 \text{ so that } P\left(\mathcal{L}_{t+1} \leq \mathcal{L}_t \vee (\mathcal{C}_{t+1} - \mathcal{C}_t) > \eta\right) \geq 1 - \epsilon \tag{4.34}$$

$$\Leftrightarrow \forall \eta > 0, \exists \epsilon \geq 0 \text{ so that } P\left(\mathcal{L}_{t+1} \geq \mathcal{L}_t \wedge (\mathcal{C}_{t+1} - \mathcal{C}_t) < \eta\right) \leq \epsilon \tag{4.35}$$

□

Proposition 2. *As long as the global entropy $\mathcal{H}$ is not stable, the global likelihood L increases on average during the Meta E-Step of our proposed framework.*

Proof. Let's make the hypothesis that the global entropy $\mathcal{H}$ is not stable and that we have the global likelihood L decreasing during the Meta E-Step of our proposed framework.

- By construction, we know that both $\mathcal{L}$ and $\mathcal{C}$ cannot be decreasing at the same time because $\mathbf{S}_{t+1}$ is locally computed from an $argmax$.
- The case $\mathcal{C}_t = \mathcal{C}_{t+1}$ whatever the variations of $\mathcal{L}$ implies that $\mathcal{H}$ is constant. This case is for instance possible when the collaborators swap partitions thus affecting $\mathcal{L}$ without changing $\mathcal{C}$.
- The case $\mathcal{L}_t < \mathcal{L}_{t+1}$ and $\mathcal{C}_t > \mathcal{C}_{t+1}$ is unlikely -yet not impossible- during the Meta E-Step because of Corollary 1.
- When $\mathcal{L}_t < \mathcal{L}_{t+1}$ and $\mathcal{C}_t < \mathcal{C}_{t+1}$, we immediately have $L_{t+1} \geq L_t$.
- The case $\mathcal{L}_t > \mathcal{L}_{t+1}$ and $\mathcal{C}_t < \mathcal{C}_{t+1}$ may occur when the local term is already optimized, but not the collaborative term (hypothesis 1). In this case, the collaborative term helps the local term getting out of a local minimum. As a consequence, whenever this event occurs it is followed by at least one subsequent iteration where both terms are again optimized (subcase 4). Thus, we have $P(\mathcal{L}_t > \mathcal{L}_{t+1} \wedge \mathcal{C}_t < \mathcal{C}_{t+1}) \leq 0.5$. There are two sub-cases when this event occurs:
  1. $L_{t+1} \geq L_t$ despite the local term decreasing. This happens when the collaborative term is dominant over the local one.
  2. $L_{t+1} \lesssim L_t$ and the global likelihood slightly decreases. In this case, because $P(\mathcal{L}_t > \mathcal{L}_{t+1} \wedge \mathcal{C}_t < \mathcal{C}_{t+1}) \leq 0.5$ and $\tilde{g}$ increases fast, the decrease will be compensated in the following iterations. Thus L still increases on average.

Since the hypothesis of a decreasing global likelihood with an unstable global entropy $\mathcal{H}$ is unlikely, then Proposition 2 is verified.

□

Using Propositions 1 and 2, we have the following theorem:

Theorem 1. *When using only prototype based algorithms during the collaborative process of our proposed method, the global likelihood function L increases on average toward a stationary point.*

Proof. Using Proposition 1, we know that improving the local term $\mathcal{L}$ during the Meta M-Step will improve the collaborative term $\mathcal{C}$ on average. Therefore the global likelihood L increases on average during the Meta M-Step.

In the same way, using Proposition 2, we know that as long as the global entropy $\mathcal{H}$ is not stable, the global likelihood L also increases on average during the Meta E-Step.

Therefore, the global likelihood L converges on average toward a stationary point.

□

Non Prototype-based algorithms:

Proposition 3. *When the collaborative process includes only clustering algorithms that are non-prototype based, the global likelihood of the system as defined in Equation* (4.26) *increases strictly toward a stationary point through the algorithm's iterations.*

Proof. With non-prototype based algorithms, we have: $\forall i, \Theta^i = \emptyset$. As a consequence the Meta M-Step is voided, and using Proposition 2 we directly have that the global likelihood of the framework converges on average toward a stationary point. □

It is worth mentioning that since the Meta E-Steps optimizes $\mathbf{S}_{t+1}$ based on $\mathbf{S}_t$, it may still take several iterations before reaching convergence.

Remark: We would like to mention that regardless of whether the collaborators are prototype-based or not, the global convergence of the system does not say that all algorithms will converge locally. In fact, it is very possible that several collaborators may enter in a cycle of swapping their respective partitions, or in a local oscillatory states where the variations compensate each other. Such behaviors have been observed during the experiments. The same issues have been mentioned by Bickel and Sheffer in their work on multi-view clustering [16]. This is due to the local nature of the the parameters optimization during the Meta M-Step.

4.3.4 Experimental results

In order to evaluate our approach, we have applied our algorithm to seven data sets of different sizes and complexity: the Iris data set, the Wine data set, the Wisconsin Diagnostic Breast Cancer data set, the Escherichia coli data set, the Image Segmentation data set and the Spam Base data set from the UCI repository [47] (See Appendix C), as well as the VHR Strasbourg data set [111] (See Section 3.3.3). All data have been normalized before the experiments.

As criteria to validate our approach we have used two internal validation criteria (See Section 2.6.1): the Davies-Bouldin Index [36], and the Silhouette Index [113]; And the Adjusted Rand Index [69] as an external criterion to compare the solution vectors with known solutions and ground truths (See Appendix B).

The Davies-Bouldin index and the Silhouette index have both been considered because they do not assess the same clustering properties: the Davies-Bouldin index searches for the worst scatter/separation ratio between all the clusters, while the Silhouette index gives an average score on whether or not each data has been attributed to the right cluster based on the distance between the each data and the clusters centroids. The Davies-Bouldin index is a positive non-normalized index, the value of which is better when it is lower. The Silhouette Index is a normalized index bound between -1 and 1, with 1 indicating compact and well differentiated clusters, while -1 indicates very poor clustering results.

In our experiments, the distance metric used for both the Davies-Bouldin index and the Silhouette index is the Euclidian distance.

4.3.4.1 Full collaboration

In this section, we show the results acquired from collaboration processes with between 5 to 10 algorithms. The algorithms used in the collaborative process were the K-Means and EM algorithms using the Gaussian mixture model for the Iris, WDBC and Wine

data sets, EM algorithms and GTM algorithms for the Spam base data set, and EM algorithms and several instances of our SR-ICM algorithm (See Section 3.3) for the VHR Strasbourg data set.

Experimental protocol: For all data sets the attributes were randomly split among the collaborators, with redundancies in the case of data sets having too few attributes.

For each data set, we ran several simulations using different real results from various algorithms, as well as artificial solutions created from real solutions where we added noise. For each simulation, we computed several indexes before and after the collaboration.

The results for each data sets are shown in Table 4.3. For each index we show the average results for all simulations, as well as the best and worst results we had (shown between parentheses).

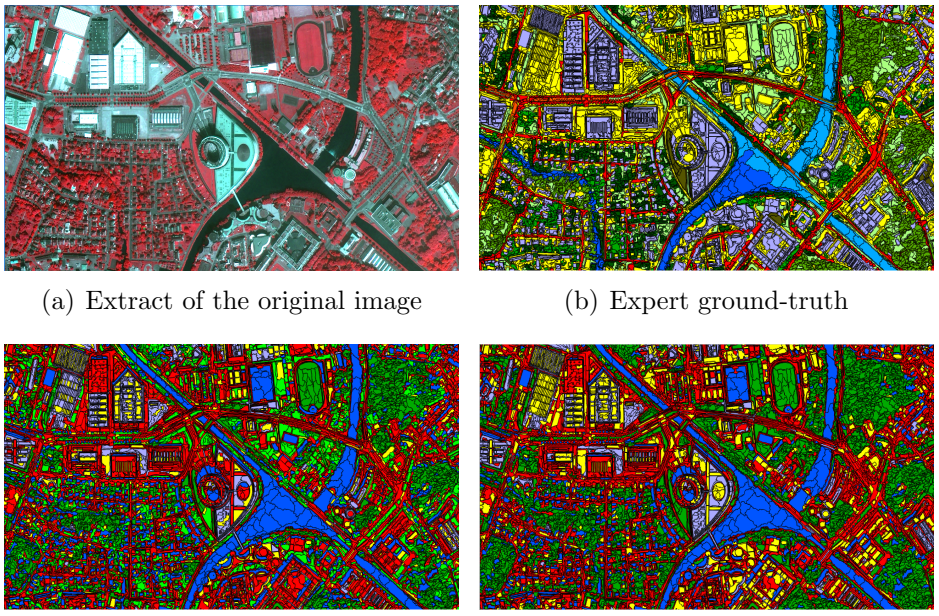
This experiment was conducted with all collaborators having the same weight ($\lambda = 1 - \frac{1}{J}$). We used the $g_\cap(\cdot)$ combination function for all data sets except for the VHR Strasbourg data set for which we tried both $g_+(\cdot)$ and $g_*(\cdot)$ due to high computational complexity issues.

Data Set	Simulations	Avg. Sil. gain	Avg. DB gain	Avg. ARI gain
Iris $g_\cap$	200×10	$0.116 \begin{pmatrix} +0.398 \\ -0.530 \end{pmatrix}$	$0.1273 \begin{pmatrix} +0.597 \\ -0.26 \end{pmatrix}$	$5\% \begin{pmatrix} +24\% \\ -18\% \end{pmatrix}$
Wine $g_\cap$	100×10	$0.063 \begin{pmatrix} +0.204 \\ -0.176 \end{pmatrix}$	$0.045 \begin{pmatrix} +0.369 \\ -0.09 \end{pmatrix}$	$5\% \begin{pmatrix} +16\% \\ -3\% \end{pmatrix}$
WDBC $g_\cap$	100×10	$0.122 \begin{pmatrix} +0.241 \\ -0.09 \end{pmatrix}$	$-0.013 \begin{pmatrix} +0.042 \\ -0.036 \end{pmatrix}$	$2\% \begin{pmatrix} +3\% \\ -2\% \end{pmatrix}$
EColi $g_\cap$	100×10	$0.114 \begin{pmatrix} +0.212 \\ -0.099 \end{pmatrix}$	$0.29 \begin{pmatrix} +0.837 \\ -0.324 \end{pmatrix}$	$4\% \begin{pmatrix} +8\% \\ -5\% \end{pmatrix}$
ImgSeg $g_\cap$	100×7	$0.091 \begin{pmatrix} +0.133 \\ -0.017 \end{pmatrix}$	$0.102 \begin{pmatrix} +0.251 \\ -0.101 \end{pmatrix}$	$-2\% \begin{pmatrix} +5\% \\ -6\% \end{pmatrix}$
Spam Base $g_\cap$	100×5	$0.037 \begin{pmatrix} +0.081 \\ -0.122 \end{pmatrix}$	$0.165 \begin{pmatrix} +0.440 \\ -0.106 \end{pmatrix}$	$10\% \begin{pmatrix} +22\% \\ -4\% \end{pmatrix}$
VHR Stras. g_+	20×5	$-0.001 \begin{pmatrix} +0.0204 \\ -0.0179 \end{pmatrix}$	$0.141 \begin{pmatrix} +0.6982 \\ -0.1992 \end{pmatrix}$	$\approx 0\% \begin{pmatrix} < 1\% \\ -1\% \end{pmatrix}$
VHR Stras. g_*	35×5	$0.027 \begin{pmatrix} +0.0372 \\ -0.049 \end{pmatrix}$	$0.377 \begin{pmatrix} +0.475 \\ -0.094 \end{pmatrix}$	$-8\% \begin{pmatrix} +6\% \\ -20\% \end{pmatrix}$

Table 4.3: Results achieved by our proposed collaborative method on UCI data sets and the VHR Strasbourg data set: imporvements on the Silhouette index, Davies-Bouldin index and Adjusted Rand index

Comments on the results: The first striking result of this experiment is that the gain for the internal quality indexes (Silhouette and Davies-Bouldin) have a lot of variations. The explanation lies in the fact that while our proposed framework aims at improving all the results, in practice the best collaborators see the quality of their results remaining constant or slightly diminishing. Nevertheless, we can see than the collaboration results for the Silhouette and Davies-Bouldin indexes remain positive on average, which tends to prove the robustness of our proposed collaborative framework.

The second point highlighted by this experiment is that our proposed collaborative framework does not solve the issue of achieving good results on external indexes (the Adjusted Rand Index here) with purely unsupervised clustering algorithms. The weaker performances achieved on the Adjusted rand index can be explained by two factors:



(a) Extract of the original image (b) Expert ground-truth

(c) ICM 7 clusters before collaboration (d) ICM 7 clusters after collaboration

Figure 4.11: Visual results on the VHR Strasbourg data set

- First, without external knowledge, there is no reason for the collaborative process to converge toward the ground truth. The idea of adding external knowledge into our collaborative process may be considered in our future works.
- Second, in the case of the VHR Strasbourg data set, the ground expert truth contains 15 clusters covering only 90% of the data set, most of them very unlikely to be found by a clustering algorithm. As a consequence, the collaborative process accentuated the situation where the clustering algorithm found only a reduced number of clusters, thus boosting indexes such as the Davies-Bouldin index, which is very high for this data set, while severely worsening the results on the Adjusted Rand Index.

In Figure 4.11, we show an example of visual results from our collaborative process on the VHR Strasbourg data set. Figure 4.11(a) is an extract of the original image and Figure 4.11(b) is a projection of the expert ground-truth for this extract. In Figure 4.11(c) we show the visual result before collaboration, and in Figure 4.11(d) we show the result after collaboration (7 clusters). As one can see, while the collaboration slightly deteriorates the Rand Index, the result after collaboration shows more homogeneous areas that make more sense when compared with the original picture. Other visual results after collaboration have displayed similar improvements in the form of better looking homogeneous areas and will be shown in the next subsection. While this visual approach is somewhat subjective, it illustrate the quality of our proposed method from an application perspective.

4.3.4.2 Hierarchical clustering on the VHR Strasbourg data set

We now propose a second experiment in which we use our proposed collaborative framework for hierarchical clustering purposes [127]. In very high resolution satellite images such as the one introduced earlier in this document, depending on the scale there may be different types of elements of interest: At the first broad level, we can

usually distinguish three main types of objects, namely water areas, vegetation areas and urban areas. At a second level with more details, we can separate different types of urban blocs, different types of vegetation areas, and start to distinguish elements such as roads. When zooming even more, very high resolutions images enable detecting small urban elements such as individual houses, cars, trees, or swimming pools.

As one can see, there is an obvious hierarchical relation between the different objects of interest that can be detected when searching at different scales for a different number of clusters. However, the huge size of these data sets usually makes them ineligible for hierarchical clustering algorithms because of their high computational complexity. We therefore propose an experiment in which we use our collaborative framework with several instances of our previously proposed SR-ICM algorithm (See Section 3.3) looking for 3, 6 and 9 clusters. In this experiment, we use the g_+ combination function and $\lambda = \frac{1}{2}$.

In our experiment, we compare our results with these of two other algorithms: the EM algorithm for the Gaussian Mixture Model [38], and the regular SR-ICM algorithm [131] both looking for 3, 6 and 9 clusters.

The results were assessed using the Davies-Bouldin index as a an internal criterion. This index assesses the compactness of the clusters and how well they are separated. It is worth mentioning that the Davies-Bouldin index usually gives better results with less clusters. As for the external index, we used the Rand Index to compare our results with the expert ground truth. We did not used the Adjusted version of the Rand index because it was giving really poor results for the clustering results with 3 and 6 clusters which is much lower than in the ground truth. The Rand index expresses in percentage how much the result matches with the expert ground-truth.

The results of this experiments over a dozen simulations for each algorithm are shown in Table 4.4, where the best result for each number of cluster is highlighted in bold.

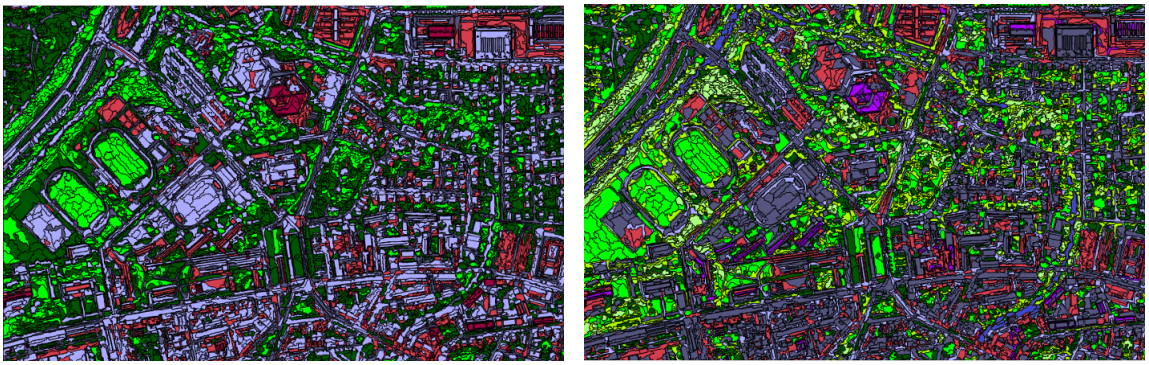
Algorithm	Davies-Bouldin Index	Rand Index
EM 3	2.36928	0.67454
SR-ICM 3	2.32855	**0.6760**
Co SR-ICM 3	**2.32674**	0.6743
EM 6	2.88014	0.7586
SR-ICM 6	2.67816	0.7693
Co SR-ICM 6	**2.49726**	**0.7706**
EM 9	2.62786	0.7822
SR-ICM 9	2.94065	0.7906
Co SR-ICM 9	**2.58836**	**0.79218**

Table 4.4: *"Hierarchical"* collaborative clustering version of our SR-ICM algorithm compared with the regular SR-ICM and the EM algorithm searching for 3, 6 and 9 clusters.

As one can see, our proposed method performs better in 5 cases out of 6. We can also mention that the improvement on the Davies-Bouldin Index are more remarkable than these on the Rand Index which cannot be considered to be significant considering the lack of strong difference between the results of the 3 algorithms. These results were to be expected since our proposed method does not change the unsupervised nature of the base clustering algorithms that we use. It is therefore logical that -just like in our previous experiment- we have better results with unsupervised indexes.

In figures 4.12(a) and 4.12(b), we show extracts of the results for the clustering results with 6 and 9 clusters. As can be seen on a color large scale version of the images, we achieve a decent segmentation at both scales with most elements being recognizable. An interesting remark is that while we did get a hierarchical structure between the clusters at different scales, the 3 initial clusters were not the ones that we expected, had we used a supervised method. The 3 clusters found at the first scale by all algorithms (collaborative or not) were: vegetation areas, industrial buildings, and a cluster containing both urban elements and water areas. Actually, water areas only became a distinct cluster at the scale with 9 clusters. As we see it, this may be due to the fact that on the 27 attributes of the VHR Strasbourg data set, the color attributes remain the most important ones.

Nevertheless, the unsupervised methods that we used found clusters that overall made sense. In the case of our collaborative clustering hierarchical method, the results were not only better when using internal and external criteria, we also had a better hierarchical structure between the clusters from different scales.



(a) Collaborative SR-ICM result for 6 clusters (b) Collaborative SR-ICM result for 9 clusters

Figure 4.12: Clustering extracts

4.4 Discussion

In this Chapter, we have been studying the problematic of using several clustering methods together to address difficult data sets. Our goal was to share the solutions found by different algorithms or experts and to use them with the aim of finding better solutions and results. We have shown a large range of potential applications for this type of multi-algorithm frameworks ranging from multi-view clustering to consensus-based methods.

We have studied several existing methods available in the literature, and we have proposed a method of our own applicable to the case of *horizontal collaboration* where several algorithms tackle different views or different scales of the same data. Our proposed method is generic and can work with almost any type of clustering algorithm. We have also demonstrated its efficiency when applied to different data sets and problems. Furthermore, we have also studied its convergence properties.

In the next two chapters, we will address two main issues with our method (issues that can also be found in some other collaborative clustering frameworks):

- In the next chapter, we will address the issue of predicting and optimizing the results of a collaboration, and reducing the risk of negative collaboration where the results of most algorithms are deteriorated after collaboration.
- In Chapter 6, we will deal with the case of *vertical collaboration* where several algorithms work on different data with similar distributions. The collaborative method that we have proposed in the current chapter can not be used for this kind of application, and we will show how it can be modified to tackle this type of collaborative clustering problem.

Chapter 5

Optimizing the collaboration process

Contents

While the collaborative algorithm proposed in the previous Chapter (see section 4.3) has convergence properties somewhat similar with these of the EM algorithm, in this chapter we address an issue that we have not yet discussed: The fact that the algorithm converges does not allow to make any hypothesis on the quality of the results. In fact as we have seen the experiments, the risk of negative collaboration -where poorly performing algorithms drag down the others- remains quite likely.

We are therefore interested in studying the factors that may influence the final result of a collaboration with the goal of determining which collaborations should be favored and which shouldn't be in order to predict and maximize the outcome of the collaborative process. Within this context, in this chapter we propose two different ways of optimizing the weights $\tau_{j,i}$ from Equations (4.21) and (4.22), a strategy that will effectively allow favoring some collaborations over some others.

To do so, we propose two methods that we will introduce and discuss in the coming sections:

- An optimization strategy aiming at choosing weights that maximize the likelihood function of our algorithm.
- Another weighting strategy based on empirical and regression results.

5.1 Optimization of the collaborative term under KKT conditions

In our first approach, we study how optimizing the weights of the combination function $g_+(\cdot)$ can lead to a higher value of the likelihood function and reduce the risk of negative collaboration by further optimizing weight factors between the algorithms.

$$g_+^i(\boldsymbol{q}, c) = \frac{1}{Z} \sum_{j \neq i} \tau_{j,i} \frac{|q_i \cap q_j|}{|q_j|}, \quad q_i = c$$

To do so, we propose to use the combination function $g(\cdot)$ described in Equation (4.21) (reminded above) that allows to weight the different algorithms. This function sums the pairwise likelihood of having $q_i = c$ based on the *a posteriori* intersections of the clusters in the different solutions. Each likelihood is then weighted by the term $\tau_{j,i}$ that describes the degree to which algorithm $\mathcal{A}^j$ can influence algorithm $\mathcal{A}^i$. This function is equivalent to a weighted vote where each algorithm gives a degree of agreement between its cluster q_j and the local algorithm cluster $q_i = c$. The term Z is a normalization constant so that $\sum_{c=1}^{K_i} g_+^i(\boldsymbol{q}, c) = 1$.

$$Z(i, \boldsymbol{q}) = \sum_{c=1}^{K_i} \sum_{j \neq i} \tau_{j,i} \frac{|c \cap q_j|}{|q_j|}$$

We also remind that $|q_j|$ is the number of elements that are in the cluster $q^j \in [1..K_j]$ of algorithm $\mathcal{A}^j$, and that $|c \cap q_j|$ is the number of elements that are in both the cluster $q_i = c$ of algorithm $\mathcal{A}^i$ and the cluster q_j of algorithm $\mathcal{A}^j$.

Given this Equation, the weights $\tau_{j,i}$ should obviously be chosen to maximize the likelihood function L which in turn should ensure better results. Using Equations (4.26)

and (4.21), this is equivalent to find the $\tau_{j,i}$ that maximize the collaborative term for all data and all algorithms and thus to maximizing Equation (5.1):

$$\mathcal{C}(X, \mathbf{S}, \mathbf{\Theta}) = \sum_{i=1}^{J} \sum_{n=1}^{N} \sum_{c=1}^{K_i} \lambda_i \sum_{\boldsymbol{q} \in \mathcal{Q} | q_i = c} \left[\frac{1}{Z(i, \boldsymbol{q})} \sum_{j \neq i} \left(\tau_{j,i} \frac{|c \cap q_j|}{|q_j|} \right) \cdot \prod_{\substack{j=1 \\ j \neq i}}^{J} s^j_{n,q_j}(\Theta^j) \right] \tag{5.1}$$

In order to compute the optimal $\tau_{j,i}$, we change Equation (5.1) into Equation (5.2), where we simply changed the position of the sum $\sum_{j \neq i} \tau_{j,i}$ and where $R(n, \boldsymbol{q}, i, \mathbf{\Theta}) = \prod_{j \neq i} s^j_{n,q_j}(\Theta^j)$.

$$\mathcal{C}(X, \mathbf{S}, \mathbf{\Theta}) = \sum_{i=1}^{J} \sum_{j \neq i} \tau_{j,i} \cdot \lambda_i \sum_{n=1}^{N} \sum_{c=1}^{K_i} \sum_{\boldsymbol{q} \in \mathcal{Q} | q_i = c} \left(\frac{1}{Z(i, \boldsymbol{q})} \frac{|c \cap q_j|}{|q_j|} \cdot R(n, \boldsymbol{q}, i, \mathbf{\Theta}) \right) \tag{5.2}$$

In turn, Equation (5.2) simplifies into Equation (5.3) where $\beta_{j,i}$ can be interpreted as a non-normalized criterion assessing the degree of agreement of algorithm $\mathcal{A}^j$ with algorithm $\mathcal{A}^i$. $\beta_{j,i}$ is an asymmetrical information based similarity measure between two different clustering solutions.

$$\mathcal{C}(X, \mathbf{S}, \mathbf{\Theta}) = \sum_{i=1}^{J} \sum_{j \neq i} \tau_{j,i} \cdot \beta_{j,i}(X, \mathbf{S}, \mathbf{\Theta}) \tag{5.3}$$

5.1.1 Theoretical results

We now want to find the $\tau_{j,i}$ that maximize Equation (5.3). We do so under the Karush-Kuhn-Tucker conditions (KKT) [87] assuming that the weights $\tau_{j,i}$ respect the condition given in Equation (5.4).

$$\forall i \quad \sum_{j \neq i}^{J} (\tau_{j,i})^p = 1, \qquad p \in \mathbb{N}^* \tag{5.4}$$

The results of the optimization under the Karush-Kuhn-Tucker conditions are shown bellow for different values of p in Equations (5.5), (5.6) and (5.7). The complete calculus for these results is available in Appendix D.

- If $p = 1 : \forall j \neq i, \tau_{j,i} = \begin{cases} \frac{1}{Card(\beta_{j,i} = max_k(\beta_{k,i}))} & \text{if } \beta_{j,i} = max_k(\beta_{k,i}) \\ 0 & \text{otherwise} \end{cases}$ (5.5)
- For $p > 1 : \forall j \neq i, \tau_{j,i} = \frac{|\beta_{j,i}|^{\frac{1}{p-1}}}{(\sum_{k=1}^{J} |\beta_{k,i}|^{\frac{p}{p-1}})^{\frac{1}{p}}}$ (5.6)
- When $p \to \infty : \forall (i, j), \tau_{j,i} = Cte$ (5.7)

The interpretation of these results is the following: in the context of horizontal collaborative clustering, the global results should be better if each individual algorithm gives higher weights to algorithms that have the most similar solutions compared with the local one (high $\beta_{j,i}$ value for a given $\mathcal{A}^i$).

Going deeper, we see that the degree to which one algorithm should collaborate with other collaborators that have dissimilar solutions depends on the degree of normalization p in Equation (5.4). For $p = 1$ (Equation (5.5)), each algorithm would only

collaborate with the algorithm that has the most similar solution. If several algorithms have the same most similar solution, they would be given the same weight. When using a higher degree of normalization (Equation (5.6)), the algorithms with the most similar solutions would still be favored to optimize the likelihood of the global collaborative framework. But algorithms the solutions of which have a lesser degree of similarity would still be taken into consideration locally. In fact as p gets higher, the solutions from dissimilar algorithms would have a heavier and heavier weight, and at some point they would matter just as much as any other solution. In this later case, when the value of p is high enough, this would be equivalent to give the same weight to all the algorithms (Equation (5.7)).

5.1.1.1 Extension to other combination functions

It is our strong belief that the results of the previous section are applicable to any function that uses weights in the context of horizontal collaboration. To prove our point, we first take the example of the simplified collaboration function with hard or semi-fuzzy clusters. Then we use a different weighted combination function $g(\cdot)$ from one of our earlier work. In both cases, we show that the best weights still rely on giving more importance to solutions that are similar to the local partition.

In the case of the simplified collaboration with hard or semi-fuzzy clusters using Equation (4.19) and the combination function $g_+(\cdot)$ from Equation (4.21). In this case, we can find the exact same results after the KKT optimization but with a simplified $\beta_{j,i}$, see Equation (5.8).

$$\beta^+_{j,i} = \lambda_i \sum_{n=1}^{N} \frac{1}{Z(i,n)} \frac{|s^{max}_{ni} \cap s^{max}_{nj}|}{|s^{max}_{nj}|} \tag{5.8}$$

The same properties can also be found with other combination functions. For example with Equation (4.22) that defines $g_*(\cdot)$, we use a product for the combination function instead of a sum. The weighted version of this combination function combined with the simplified model from Equation (4.19) would lead to the same optimal weights as in Equations (5.5), (5.6) and (5.7), with a slightly different $\beta_{j,i}$ that clearly still is a similarity index. This $\beta_{j,i}$ is shown in Equation (5.9).

$$g^i_*(\boldsymbol{q}, c) = \frac{1}{Z} \prod_{j \neq i} \left(\frac{|c \cap q_j|}{|q_j|} \right)^{\tau_{j,i}}$$

$$\beta^*_{j,i} = \sum_{n=1}^{N} \ln \left(\frac{|s^{max}_{ni} \cap s^{max}_{nj}|}{|s^{max}_{nj}|} \right) - \ln Z(i,n) \tag{5.9}$$

To further demonstrate that our proposed optimization method and results are generic, we argue that they can easily be adapted to other frameworks. For instance, in the case of the algorithm proposed by Pedrycz [101] the prototype-based collaborative methods proposed by Grozavu et al. [55] and Ghassany et al. [50] the α and β in their equations are equivalent to our τ. The only difference is that they want to minimize a penalty function instead of maximizing a likelihood.

5.1.1.2 Interpretation

If we think about the goal of collaboration, these mathematical results make sense: the goal of collaboration is to have the algorithms mutually helping each others. In

this context, when several algorithms find solutions that are similar, it is quite likely that they have actually found a structure in the data. Therefore, collaborating with algorithms that have solutions similar to the local partitioning is a convenient way to avoid the risk of negative collaboration. There are actually two good reasons not to collaborate with an algorithm the results of which are too different from the local partition:

1. This collaborator may be in a feature space where the clusters to be found are completely different, even for the same objects.

2. If this collaborator has a solution that is dissimilar with these of all other algorithms, maybe it is just a bad solution.

These results can also be linked to recent works on clustering stability [11, 94]. A clustering is said to be stable if the partitioning remains similar when the data set or the clustering process are perturbed. In the context of collaborative clustering, the perturbations would be that (1) we observe the same data in different feature spaces, and (2) we use different algorithms and models. With this proposed weighting method, the algorithms that will have the highest weights will be these with solutions that are the most often similar to the other algorithms' solutions. It matches with the definition of stability: such solutions that features common structures and clusters through several feature spaces with different algorithms are the most stable.

As a conclusion to this theoretical section and based on the previous results, we make the following proposition:

Proposition 4. *In the context of horizontal collaboration between several heterogeneous algorithms, the most efficient way for a given algorithm $\mathcal{A}^i$ to collaborate is to favor exchanges with other collaborators $\mathcal{A}^j$ that have similar and stable solutions.*

If this proposition may seem somewhat counter intuitive, we are unaware of any existing mathematical counter-proof for collaborative clustering.

5.1.2 Experimental results

5.1.2.1 Evolution of the weights

In the following experiment, we sought to assess the behavior of our proposed weighting method with two goals:

1. Checking that the weights worked as intended with heavier weights given to algorithms with a lower diversity.

2. Assessing that the dynamic weights did stabilize at some point and thus would not hinder the convergence process of the collaborative framework.

To do so, we used the Waveform data set from the UCI website. Our experiment was the following: we ran 5 regular EM algorithms (GMM) in parallel over different versions of the data set: two with almost no noisy variables, two others with a moderate number of noisy variables, and one with a lot of noisy variables. At the end of this initial step we had results that were more or less good and close to each others depending on the number of noisy variables. We then started the collaborative step from these results and ran our proposed method with five EM algorithms collaborating together. During

each iteration and for each algorithm, we observed the absolute average evolution of the weights $\tau_{j,i}$ between the different collaborators. For this experiment, we used $p = 2$ (See Equation (5.4)) and $\lambda = 0.5$.

In Figure 5.1 we show the absolute average weight evolution per iteration for the five collaborative EM algorithms. Each curve represents one of the collaborator.

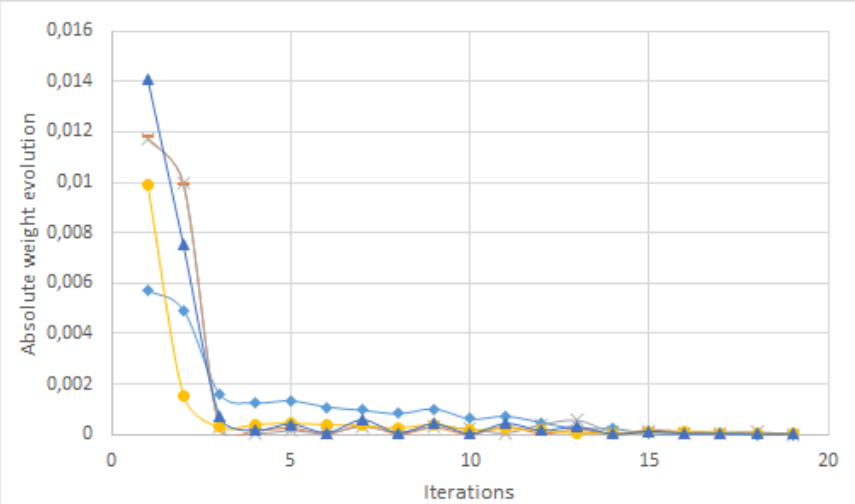


Figure 5.1: Absolute average weight evolution per iteration for five EM algorithms using the Waveform data set

As one can see, the weight slowly stops evolving over time. As displayed on the diagram this is not a strict convergence, but a convergence on average. This result makes sense since the weights are based on a similarity measure between the solution and that in our algorithm the global entropy of the system also converges on average. Another interesting remark is that we can see that the algorithms that had the most noisy variable (the 4 curves starting the highest values) had their weights changing a lot during the two first iterations but converged fast overall. On the other hand, the weights for the collaborator that had no noise evolved less in the beginning but took more time to stabilize as the other solutions became slowly more similar. The internal behavior of our weighting system was the one expected from the KKT optimization, with bigger weights being given to more similar solutions.

5.1.2.2 Performance raw results

In a second experiment, we applied our proposed framework to several other data sets from the UCI website and assessed 2 indexes before and after collaboration while using our weighting method. To do so, we ran a few simulations on several split data sets and checked the values of the Silhouette index [113] and the Davies-Bouldin index [36] at the end of the local step and the end of the collaborative step.

In our experimental protocol, for each data set, we created 5 subsets by removing some of the attributes randomly. Each subset was then assigned to an EM algorithm using the Gaussian mixture model. We then ran our collaborative framework with the parameters $p = 2$ and $\lambda = 0.5$.

The results of this experiment are shown in Table 5.1 where the average result over a dozen simulations are shown in the main cells, as well as the best and worst results over all simulations that are shown between brackets.

Data Set	Silhouette Index	DB-Index
Wine	$+1\% \begin{pmatrix} +3\% \\ -2\% \end{pmatrix}$	$+1\% \begin{pmatrix} +2\% \\ -1\% \end{pmatrix}$
WDBC	$+1\% \begin{pmatrix} +2\% \\ -1\% \end{pmatrix}$	$+1\% \begin{pmatrix} +3\% \\ +1\% \end{pmatrix}$
Waveform	$+4\% \begin{pmatrix} +15\% \\ +1\% \end{pmatrix}$	$+1\% \begin{pmatrix} +2\% \\ -1\% \end{pmatrix}$
EColi	$+9\% \begin{pmatrix} +34\% \\ -2\% \end{pmatrix}$	$+4\% \begin{pmatrix} +14\% \\ +1\% \end{pmatrix}$
Image Segmentation	$+4\% \begin{pmatrix} +12\% \\ 0\% \end{pmatrix}$	$+15\% \begin{pmatrix} +45\% \\ -3\% \end{pmatrix}$

Table 5.1: Average Improvement after collaboration using the KKT weighting system in our collaborative framework

As one can see, our proposed framework has overall positive results for two different clustering indexes. While it is true that the improvement after collaboration is not huge, our proposed weighting method has the advantage of resulting in fewer cases of negative collaborations (results getting worst after collaboration) when compared with the version that that had no weighting system in Table 4.3. These results highlight that our weighting system has the positive aspect of reducing the cases of negative collaboration, but also the flaw that it may also lead to results that on average are less impressive.

5.2 Empirical analysis of the impact of Quality and Diversity to optimize a given quality criterion

In this section, we discuss our second weighting approach. Our focus here is to try to empirically determine criteria to optimize the results of collaborative clustering based on known properties (quality indexes and diversity) of the collaborators participating in the process.

Our methodology is the following: we use the original version of our horizontal collaborative framework as it was introduced in Section 4.3, **without the KKT weighting system**. We first evaluate the performances of our collaborative framework depending on the initial quality and diversity of the solutions. This is done in Section 5.2.2 where we use point clouds to display the results of a large number of simulations over several data sets. From there we make an analysis of the influence of quality indexes and diversity to predict the outcome of a collaboration. We also use this section to assess some extra properties of our collaborative framework. Then in Section 5.2.3, we use the point clouds to build a regression model to predict the outcome of a collaboration in term of internal quality criterion. Finally, we use this regression model to propose new weights τ for the collaborators, and we evaluate the effects of such a weighting system in a short experiment.

5.2.1 Collaboration vocabulary

Before starting with the experimental evaluation of our collaborative framework, we want to give some specific vocabulary and elements that will be useful to interpret the point clouds from Section 4.3.

First of all, in the following sections we will evaluate the evolution of quality and diversity from the point of view of individual algorithms: If we consider Q a quality index (internal or external), then GQ the raw quality gain will be evaluated individually from the point of view of each algorithm. In short anything related to a quality index (Q, GQ, ΔQ) will be computed from the point of view of a *local algorithm* in the *local feature space*, while diversity measures will be computed based on the other *collaborating algorithms*.

Based on these definitions, earlier works have defined a collaboration to be a "positive collaboration" when $GQ > 0$ and a "negative collaboration" otherwise.

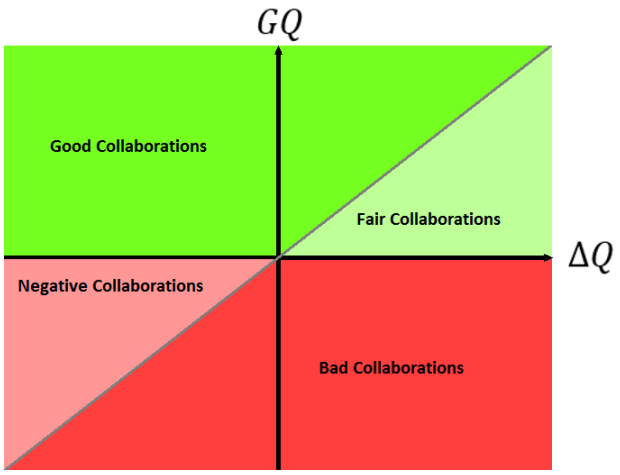


Figure 5.2: The different types of outcome for a collaboration

If we consider ΔQ the average quality difference before collaboration between two given algorithms solutions in the feature space of the observed algorithm, Figure 5.2 shows what a typical $GQ = f(\Delta Q)$ point cloud diagram will look like. Points in the green area would be positive collaborations while points in the red area would be negative ones.

We extend this vocabulary as follows by adding some contrast to the possible outcomes of a collaboration:

- A collaboration outcome where the local result is improved so that it becomes better than the average results of the other collaborators (grey diagonal line) before collaboration will called a *good collaboration*.
- If the result improves but remain lower than than the average quality of the other collaborators before collaboration, it is a *fair collaboration*.
- If the result gets worse while remaining above the average quality before collaboration, then it is a *negative collaboration*.
- Finally, if the result are worse and are bellow the the average quality before collaboration, such collaboration is not just a negative collaboration, it is a *bad collaboration*. These terms are illustrated in the diagram shown in Figure 5.2.

5.2.2 Point clouds results

In this section, we analyze the influence of the diversity and the quality of the collaborators on our proposed framework. In our case, we choose to use a local oriented entropy measure (Equation (5.10)) derived from the global entropy in Equation (4.25) as a diversity index, and both the Silhouette index and the Davies-Bouldin index as quality indexes.

$$H_{i,j} = \frac{-1}{K_i \times \ln(K_j)} \sum_{l=1}^{K_i} \sum_{m=1}^{K_j} \omega_{l,m}^{i,j} \ln(\omega_{l,m}^{i,j}) \tag{5.10}$$

The experimental protocol is the following: we randomly modified already existing solution vectors that we had found for all respective subsets and we analyzed the evolution of the Davies-Bouldin index and Silhouette index for 1 iteration of our collaborative process using two or three of these random solution vectors collaborating together. The data sets are randomly split to have 2 or 3 algorithms working on different attributes. A similar protocol is used with the VHR Strasbourg data set, but in addition to that, the number of cluster to be found being unknown the collaborators are looking for a different number of clusters. The algorithms used is the collaborative EM algorithm for all data sets. All experiments were realized with $\lambda = 1 - \frac{1}{J}$.

The results are shown in the form of four point clouds for all data sets:

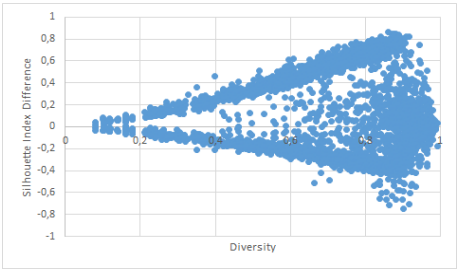
- The raw Silhouette index difference between the two collaborators depending on the initial diversity.
- The Silhouette index raw improvement depending on the initial entropy between the collaborators.
- The Silhouette index raw improvement depending on the initial Silhouette index raw difference between the two collaborators.
- The Davies-Bouldin index raw improvement depending on the initial Davies-Bouldin index raw difference between the two collaborators.

Note that the first point cloud is not an experimental result and is just here to show that our simulation covered most possible cases of differences in quality and diversity prior to the collaborative process.

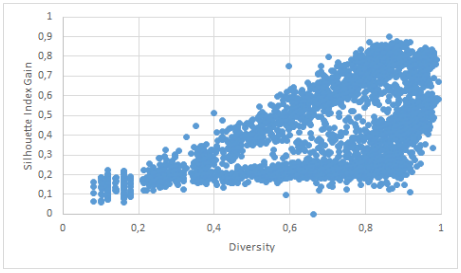
We used point clouds containing 1500 simulated collaborations (3000 points since the collaboration is evaluated both ways) for the Iris, WDBC, Wine and Image data sets, and 250 simulated collaborations (500 points) for the VHR Strasbourg data set because of computation time issues. The point clouds for these experiments are shown in Figures 5.3, 5.4, 5.5, 5.6 and 5.7.

We first want to analyze Figures 5.3(a), 5.4(a), 5.5(a), 5.6(a) and 5.7(a). These figures show the initial characteristics of the collaborators before collaboration by comparing the initial quality difference using the silhouette Index with the initial diversity.

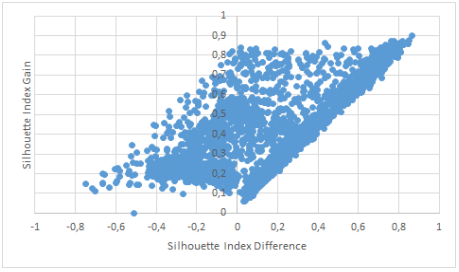
As one can see, in the case of the easy data sets -small in size and with few attributes- we were able to generate random example that are representative of the full range of possible couples, see Figures 5.3(a), 5.4(a) and 5.5(a). However, for more complex data sets we were unable to cover the whole range of possible couples, see Figures 5.6(a) and 5.7(a). Specifically, the random generator could not find clustering results that have a similar quality index but a high diversity. Nevertheless, we believe that our results are still significant even for complex data sets.



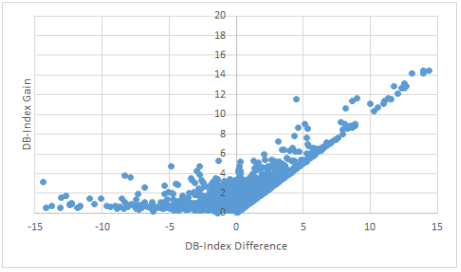
(a) $\Delta Silhouette = f(Diversity)$



(b) $SilhouetteGain = f(Diversity)$

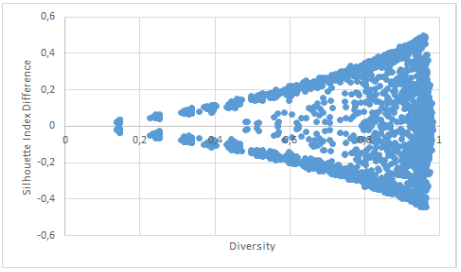


(c) $SilhouetteGain = f(\Delta Silhouette)$

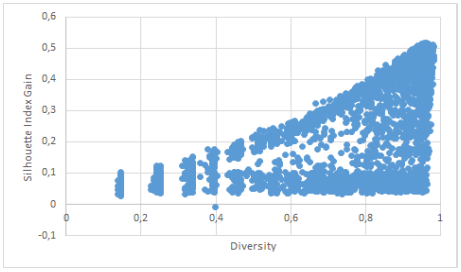


(d) $DBGain = f(\Delta DB)$

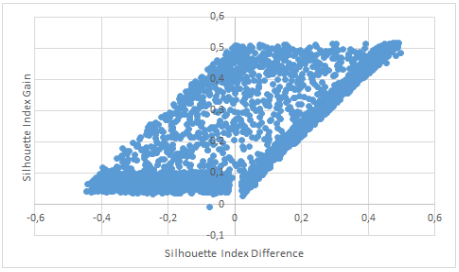
Figure 5.3: Iris Data Set Point Clouds



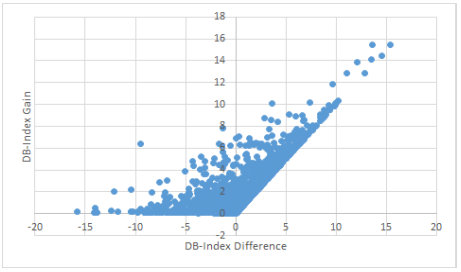
(a) $\Delta Silhouette = f(Diversity)$



(b) $SilhouetteGain = f(Diversity)$



(c) $SilhouetteGain = f(\Delta Silhouette)$

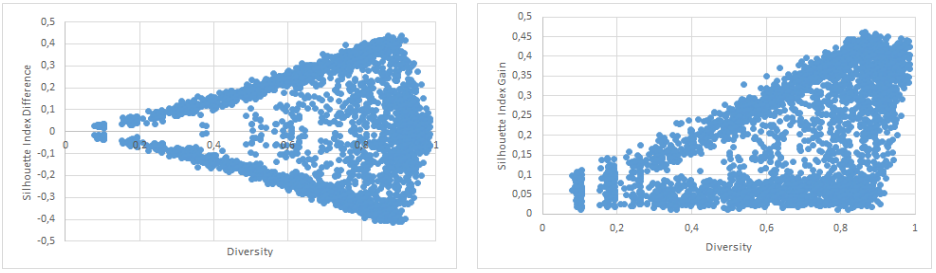


(d) $DBGain = f(\Delta DB)$

Figure 5.4: WDBC Data Set Point Clouds

We now move to the analysis of the influence of Diversity on the collaboration results. In figures 5.3(b), 5.4(b), 5.5(b), 5.6(b) and 5.7(b), we show how the value of the Silhouette Index evolved during the collaboration depending on the initial diversity between the two clustering solutions before collaboration.

The influence of diversity has already been the subject of an earlier study in our team, see [56], which after experiments similar to ours with another collaborative framework and supervised indexes concluded that a diversity around 50% has the best potential to give good collaboration results. However, as can be seen in our figures, we do not find the same results. While both studies agree that a low diversity induces a

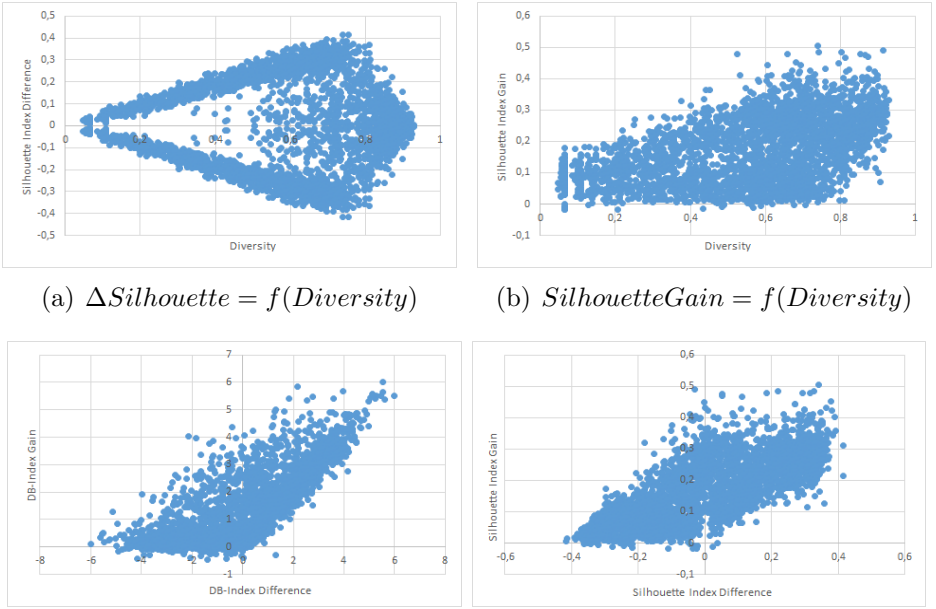


(a) $\Delta Silhouette = f(Diversity)$ (b) $SilhouetteGain = f(Diversity)$

(c) $SilhouetteGain = f(\Delta Silhouette)$ (d) $DBGain = f(\Delta DB)$

Figure 5.5: Wine Data Set Point Clouds

small potential for a good collaboration, our experiments do not show any performance spike around 50%. In our case, a higher diversity always induces a stronger potential for better results. By potential we mean that the best results were always achieved when the diversity was very high, but that a high diversity does not always imply good collaboration results. These divergences between our empirical results, the results of previous empirical studies, and the theoretical results are discussed in Section 5.3.

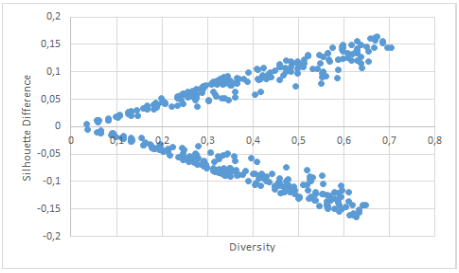


(a) $\Delta Silhouette = f(Diversity)$ (b) $SilhouetteGain = f(Diversity)$

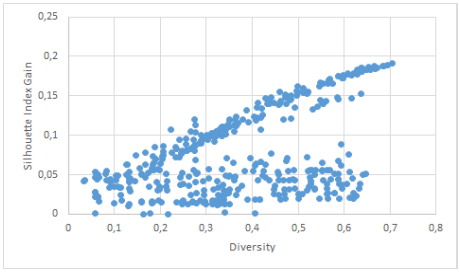
(c) $SilhouetteGain = f(\Delta Silhouette)$ (d) $DBGain = f(\Delta DB)$

Figure 5.6: Image Segmentation Data Set Point Clouds

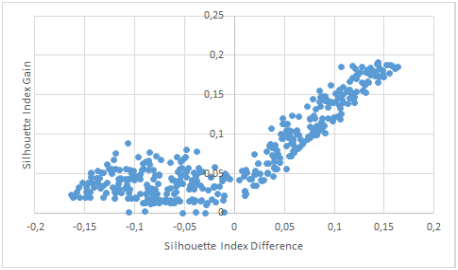
Finally, we now move to Figures 5.3(c), 5.3(d), 5.4(c), 5.4(d), 5.5(c), 5.5(d), 5.6(c), 5.6(c), 5.7(c) and 5.7(d). In these figures, we show how the initial quality difference



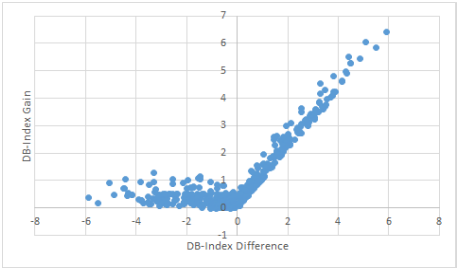
(a) $\Delta Silhouette = f(Diversity)$



(b) $SilhouetteGain = f(Diversity)$



(c) $SilhouetteGain = f(\Delta Silhouette)$



(d) $DBGain = f(\Delta DB)$

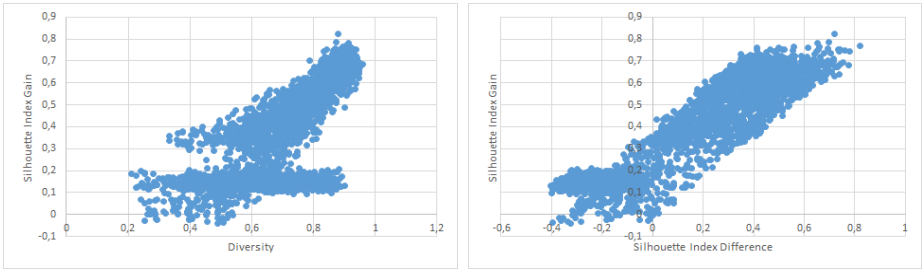
Figure 5.7: VHR Strasbourg Data Set Point Clouds

between two results before collaboration can influence the outcome of the collaboration. We conducted this study using two indexes: the Silhouette Index and the Davies-Bouldin Index.

In term of global performances of our original algorithm, if we analyze our results using the diagram that we introduced in Figure 5.2, we can see that our proposed collaboration frameworks gives very good results: for all data sets, most of the collaboration results fall into the *good collaboration* area: when receiving information from a better collaborator, the results after collaboration are almost always improves (fair collaboration) and in many cases they even beat the initially best collaborator (good collaboration). And when receiving information from a lower quality collaborator, the results are still improved.

We can see that for all data sets, most cases of negative and bad collaborations occur either when collaborating with solutions that have a similar or lower quality. When we cross these results with these on the influence of diversity, we can highlight that these negative collaboration areas are found when both the diversity and quality of the collaborators are low. We can clearly see that collaborators that have a very low comparative quality but a high diversity give better collaborative results than those that also have a lower quality but a low diversity. Our explanation for this result is that given our collaborative model, in the case of a very weak collaborator having a high diversity (i.e. an almost random solution vector), the collaborative term looses all its influence and only the local algorithm term matters. This therefore results an improvement despite the negative collaborator. However, in the case of a weak collaborator but a low diversity (weak local result), the collaborative term still has a lot of influence in the process and thus may lead to a deterioration of the results.

We then applied the same protocol to have 3 random collaborators instead of only 2 working together, with the goal of determining whether or not the previous results would remain true with a higher number of algorithms. What we found out is that a higher number of collaborators increases the chances of conflicts and thus decreases the overall chances of achieving the best collaboration results, see Figures 5.8(b) and



(a) $SilhouetteGain = f(Diversity)$ (b) $SilhouetteGain = f(\Delta Silhouette)$

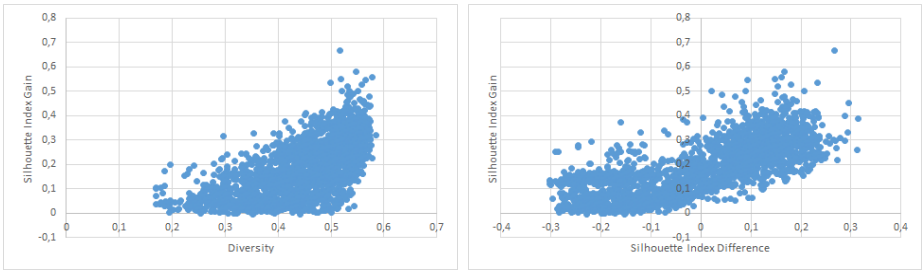
Figure 5.8: Iris Data Set Point Clouds: 3 collaborators

5.9(b). Our interpretation is that we have two sub-cases when the collaborators are better on average than the local algorithm:

1. All collaborators are better than the local algorithm, in which case the results are the same than for the first part of the experiments with only 1 external collaborator. The result after collaboration in this case is better than the average results proposed by the collaborators, and it falls in the "good collaboration" area.

2. The collaborators are better than the local algorithm only on average: some of them are still weaker than the local algorithm and negatively influence the results, thus leading to weaker performances after collaboration. This issue could therefore be solved by giving a lower weight to weaker collaborators given a specific quality criterion.

However, despite slightly weaker performances, most properties that we found with 2 collaborators remained true with 3: a higher diversity still leads to potentially better results (Figures 5.8(a) and 5.9(a)), collaborating with algorithms that are weaker on average does not necessarily leads a negative collaboration as long as the diversity is high enough, and finally the results of our proposed framework remain satisfactory with most collaboration falling into the "good collaboration" or "fair collaboration" area. We found the same tendencies with all data sets.



(a) $SilhouetteGain = f(Diversity)$ (b) $SilhouetteGain = f(\Delta Silhouette)$

Figure 5.9: EColi Data Set Point Clouds: 3 collaborators

The conclusions that we can draw from this experiment is that while our proposed framework is tolerant to weak collaborators and mostly negates their influence, the ideal conditions for good collaborations would require algorithms that already give high quality results, as well as a diversity as high as possible between these results. Therefore most risks can probably be averted by detecting algorithms with solutions

that have low quality and diversity indexes. And we will show on what basis this can be done in the next section.

It would also be possible to reduce the computational complexity of the framework by removing them early in the collaborative process.

One possible weakness of this experiment is that it is not easy to determine whether the improvements on the results come mostly from the local term, the collaborative one, or both. A possible answer to this question could lie in Figures 5.6(b) and 5.6(c), as well as Figures 5.7(b) and 5.7(c), where we can see the the quality gain on the Silhouette index based on the initial diversity is a folded version of the quality gain depending on the initial quality difference. Based on these point clouds, and if we assume for a moment that all improvements with a weaker collaborator come only from the local term, the real influence of the collaborative term can be approximated on the diversity based diagrams and is still significant.

5.2.3 Extrapolation of the results

In this subsection, we show how to use point clouds similar to these of the previous section with the goal of extracting a model that could predict the outcome of a collaboration.

We Consider the following variables :

- Q_{loc}, the quality of the local clustering result before collaboration using the silhouette index. It is computed in the local feature space.

- Q_{col}, the quality of the collaborator clustering result before collaboration. It is computed using the Silhouette index in the local feature space of the algorithm receiving the collaboration.

- $\Delta Q = Q_{col} - Q_{loc}$, the quality difference between the receiving algorithm and the collaborating algorithm.

- H, the diversity with the collaborator before collaboration . We use our oriented entropy $H_{i,j}$ to get the potential information gain that the collaborator can provide.

- GQ, the quality improvement during the collaboration. It is the raw improvement on the silhouette index for the local algorithm. If the Silhouette index after collaboration is better than the one before, $GQ > 0$. Otherwise, in case of a negative collaboration $GQ \leq 0$.

For our point clouds, we re-used the same clouds than in the previous experiment with only two collaborators working together and we generated some extra points under the same conditions with the goal of having enough data to build a decent modem. In Table 5.2, we show the number of points used per data set to create this model.

Using WEKA (The Waikato Environment for Knowledge Analysis), we applied a Linear Regression algorithm to all point clouds with the goal of predicting GQ as described in Equation (5.11). We tried other regression models of higher orders, but the linear regression was ultimately the one that gave the best results.

$$GQ = f(H, \Delta Q) = a \times H + b \times \Delta Q + c \tag{5.11}$$

Data Set	Number of collaborations
Iris	3000
Wine	3000
WDBC	3000
EColi	3000
ImgSeg	3000
VHR Strasbourg	500

Table 5.2: Subsamples characteristics

We first tried to use Q_{loc} and Q_{col} as regression parameters, but the two always ended up with opposite regression factors which prompted us to use ΔQ as a regression factor instead. The resulting regression factors using ΔQ and H to predict the GQ the gain in quality are shown in Table 5.3.

Data Set	D	ΔQ	Constant c	Correlation Coeff.	Abs. Mean Error
Iris	0.5612	0.5405	-0.0723	0.9683	0.0499
Wine	0.3262	0.4763	-0.0309	0.9718	0.0231
WDBC	0.3410	0.5337	-0.0032	0.9499	0.0302
Ecoli	0.3352	0.4518	0.0419	0.9510	0.0292
ImgSeg	0.2907	0.4438	0.0032	0.9221	0.0302
VHR Strasbourg	0.2456	0.4858	0.0212	0.9519	0.0100
Average	0.3363	0.4911	-0.0295	0.9513	0.03374

Table 5.3: Regression factors for the prediction of GQ

Following these results, we ran again the linear regression algorithm on a global set grouping the simulations from all data sets. We obtained the results shown in Equations (5.12). The correlation coefficient for GQ was evaluated to 89.80% with an absolute mean error of 0.0562. The mean errors and correlation coefficients using these equations on the individual data sets is shown in table 5.4

$$GQ = 0.3443 \times D + 0.5195 \times \Delta Q - 0.0208 \tag{5.12}$$

Data Set	GQ Mean Error	GQ Correlation
Iris	0.0999	0.9547
Wine	0.0332	0.9717
WDBC	0.0526	0.9499
EColi	0.0361	0.9499
ImgSeg	0.0376	0.9223
VHR Strasbourg	0.0385	0.9090

Table 5.4: Linear Regression Absolute Mean Errors and Correlation Coefficients

As one can see that the quality gain can be correctly approximated by a simple linear regression. Furthermore, this regression gives us a good idea of which parameters are the most important, with the quality being the most determining parameter, quickly followed by diversity.

From now on we will focus on the results for the prediction of the gain in quality GQ shown in Equation (5.12), Table 5.3 and Table 5.4.

First, we can see in Table 5.3 that diversity and quality can be used to predict the potential gain in quality after collaboration. The correlation coefficient after the linear regression is around 95%, and the average error on predicting the improvement of the silhouette index is bellow 5%. Moreover, we can clearly see that the regression factors for all the data sets are quite similar, with the notable exception of the Iris data set where the diversity and the quality difference have nearly the same weights. Finally, the regression factors were similar enough to run another linear regression on a merged version of all point clouds from all data sets. The resulting factors shown in Equation (5.12) gave similarly good performances on both the individual data sets, see Table 5.4 with correlation coefficients still above 90% and absolute mean errors between 3% and 10%, as well as one the merged data point clouds with a correlation coefficient around 90% and an average mean error around 6%. These facts tend to prove that it is possible to have a reliable idea of what the result of a collaboration will be in term of quality based on the characteristics of the different collaborators.

We conducted a similar experiment attempting to predict the gain in stability given two initial collaborators, but the results were much less conclusive. The results of this experiment with stability are summed up in Appendix E.

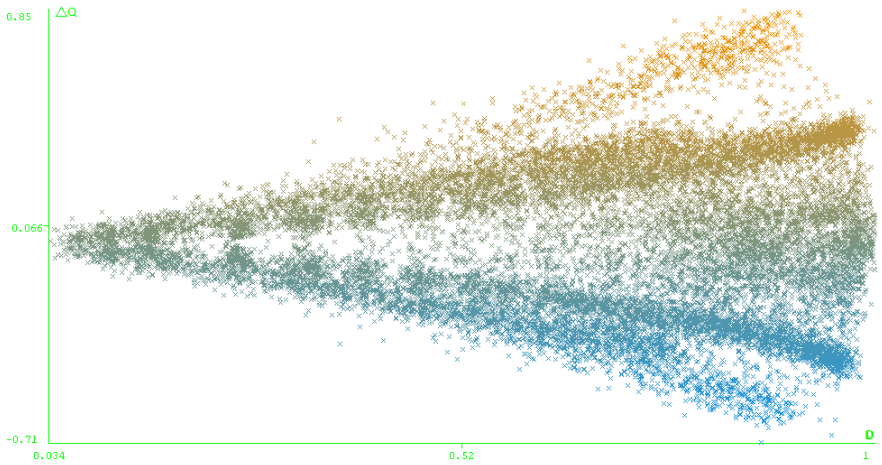


Figure 5.10: $\Delta Q = f(Diversity)$ for the merged data point clouds

The factors themselves indicates that the signed quality difference has the most important impact followed by diversity. They have a weight of approximately $\frac{1}{2}$ and $\frac{1}{3}$ respectively. Finally, all linear regression results featured a constant factor close to zero (between -0.08 and +0.05), the value of which may be dependent on the data set.

It is worth mentioning that the quality difference ΔQ and the diversity D that link two algorithms' solutions are not completely independent from each others, see Figure 5.10. The potential to have a higher absolute quality difference between two clustering results grows when the diversity increases. It means that a low diversity will always mean a low difference in quality, but that the opposite is not true.

5.2.3.1 Weighting proposition

Based on the previous empirical results, we now introduce a weighting method using the factors from the linear regression. As a consequence, for the $\tau_{j,i}$ from Equations (4.21) and (4.22), we can extrapolate the model shown in Equation (5.13) where Q is an internal quality index such as the Silhouette index, $H_{j,i}$ is the oriented entropy

between the solutions of the algorithms $\mathcal{A}^j$ and $\mathcal{A}^i$ as defined in Equation (5.10), and λ is a user input trade-off parameter.

$$\tau_{j,i} = \max\left(0, \frac{1}{2}(Q_i - Q_j) + \frac{1}{3}H_{j,i} - \lambda\right) \tag{5.13}$$

We now propose a short experiment where we analyze the influence of the risk parameter λ. To do so, we use the same data sets than in the previous sections. The collaborative framework is as follows: 10 clustering algorithms (a mix of EM algorithms using the Gaussian mixture model and some K-Means algorithms) working on different features of the data sets. We repeat this procedure using the same algorithms results for several values of λ. For each value of λ, we assess the average quality gain during the collaboration. The quality index used in this experiment is the Silhouette index.

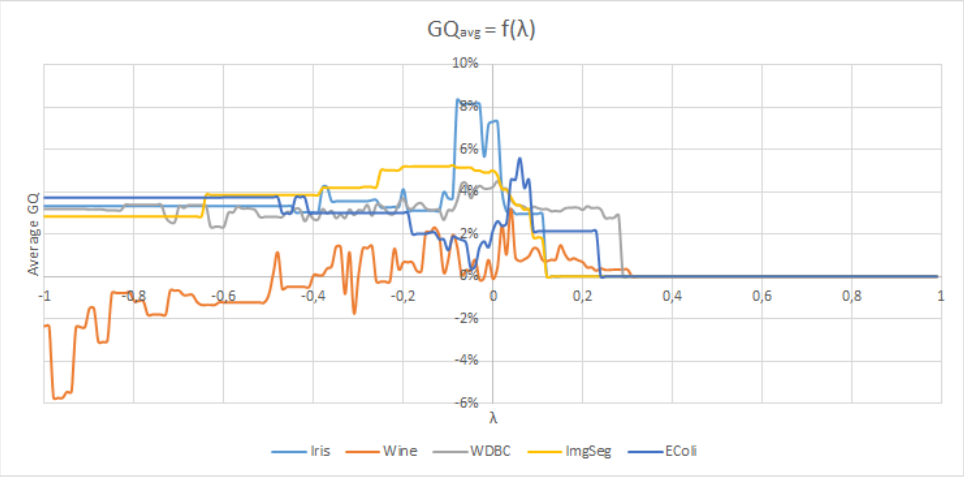


Figure 5.11: Evolution of the average improvement on the Silhouette Index (GQ_{avg}) depending on the chosen parameter λ

In Figure 5.11, we show the results of this experiments for the Iris, Wine, WDBC, Image Segmentation and EColi data sets. As one can see, the average quality gain during the collaboration depending on the parameter risk λ follows a similar pattern for all data sets:

- For the lower values of λ (near -1) all collaborators are participating in the process with a relatively significant weight. In this case the collaboration results are poor (Wine data set) or average (the other data sets).
- When λ increases and gets closer to zero, only the best collaborators are given an important weight in the collaborative process and work together. For all the data sets, this range of values results in a sharp increase of the average quality gain during the collaboration.
- Finally, when λ becomes too high, there is no collaborator left that is considered good enough by the collaborative framework and the collaboration stops, hence the 0% average gain. For most data sets, this occurs around $\lambda \approx 0.3$.

From this experiment we can draw the conclusion that the quality prediction model introduced in Equation (5.12) is relatively accurate for several data sets and can be used to define the weights given to each algorithm during the collaboration. However, the performances of this model may vary greatly depending on the choice of the risk parameter λ, a parameter that has shown to be different depending on the data set.

5.3 Discussion

In this chapter, we have proposed two methods in an attempt to predict and optimize the results of a collaboration. While our results were focused on the optimization of our own collaborative method introduced in the previous chapter, we believe that both studies are generic and can be applied to other collaborative frameworks, and compared with earlier studies.

We now want to discuss the apparently contradictory results found in sections 5.1 and 5.2, as well as the results from earlier studies. In particular, we want to address the diverging results regarding the influence of diversity: In our KKT optimization study from Section 5.1, we concluded that the best way to maximize the likelihood function of our algorithm is for each algorithm to favor solutions with which they have a low diversity. Yet, in Section 5.2 the empirical study shows that while this is not the main factor, a higher diversity should be regarded as something desirable. Finally, in their own study Grozavu et al. [56] reached the conclusion from their own experimental results that an average diversity between the solutions is the best option.

While these results may seem irreconcilable, one must look at them with pragmatism: our analysis for these divergences is that the three experiments are trying to determine optimal collaboration conditions for set ups that are very different.

In the case of the linear regression done in Section 5.2 and of the experiments from Grozavu et al., the goal was to find a setup that maximizes the collaborative outcome for an internal criterion in our case (the silhouette index), and an external criterion in their case (the Adjusted Rand Index). What our result says is that given a quality criterion Q, the best way to maximize this criterion during the collaboration is to collaborate with other algorithms that are as good as possible with this criterion, preferably much better than the local algorithm, thus favoring collaborators with a high quality and a high diversity. This result is very unsurprising and quite logical.

However, when it comes to optimizing an external criterion such as the Adjusted Rand Index based on diversity and internal criterion, the strategy is quite different. Except for very easy data set, there is usually only a mild connection between internal and external criteria. Therefore choosing other collaborators based on the internal quality of their clusters makes little sense. Choosing collaborators with an average diversity is a safe compromise between collaborating with very different solutions that may be unstable and lead to negative collaboration, and collaborating with solutions that are too close and from which less will be learned toward improvement.

For the KKT approach in Section 5.1, there was no criterion to optimize and nothing was known about the quality of the collaborators. Absent an internal or external quality criterion, the only thing that remains is stability [94]: if several algorithms come out with similar solutions, there is a good chance that the structures highlighted by these solutions are neither random nor some artefact. Therefore the mathematical result saying that in this set up it is best to collaborate with algorithms that have similar solutions makes sense because it tends to maximize the only criterion available: stability. Furthermore, for $p > 1$, the results do not say to collaborate only with similar solutions, but merely to favor them.

This result highlights a conservative approach when trying to optimize a collaborative process with no specific criterion. As shown in the experimental results, this

approach takes less risks thus leading to fewer cases of negative collaboration. But it also leads to weaker improvements because of the same lack of risk taking.

As a conclusion to this chapter on the optimization of the collaborative process, we can say that the right method to weight the collaborators heavily depends on the criterion that one tries to maximize. While there may not be a universal answer, we want to highlight that our proposed weighting method based on the KKT optimization is in our opinion the most convenient one: it relies on no specific quality index, it has an adjustable parameter p that allows to choose between conservatism and risk taking when weighting the collaborators, and it can be adapted to other collaborative frameworks.

Chapter 6

From Horizontal to Vertical collaborative clustering

Contents

6.1 Introduction to vertical collaboration

In Chapter 4, we introduced collaborative clustering in its different forms and suggested a framework that works for a broad range of applications in the context of horizontal collaboration (same data, potentially different representations). However, it is limited to horizontal collaboration.

The purpose of this chapter is to explain how our previously introduced horizontal framework can be used for vertical collaboration: while our horizontal framework is very generic, and can be adapted to almost any type of clustering algorithms for a wide range of applications, it comes with the cost that their is no easy way to turn it into a vertical collaboration framework while keeping it generic.

As we have discussed in Section 4.1.2, vertical collaboration refers to the situation where several algorithms look for clusters in different data sets containing different objects while having similar distributions. Since the method that we have introduced earlier entirely relies on clusters intersections between the different algorithms (see section 4.3.2.2), it is obvious that there is no easy way to adapt this method to the case where the collaborating algorithms are processing different data.

Our proposal to solve this issue is to take advantage of the neural network structure of two specific prototype-based clustering algorithms, namely the Self-Organizing Maps (SOM) and the Generative Topographic Map (GTM). Using these structures, we will show how to perform vertical collaboration with our previously introduced method. The work presented in this Chapter is based on one of our conference article and its journal extension [129, 130].

This chapter is organized as follows: First we introduce the SOM and GTM algorithms. Then, we explain how to use their structure for vertical collaboration purposes. Finally, we give some experimental results.

6.2 Neural Network based clustering algorithms

In Machine Learning, *artificial neural networks* are a broad family of models and algorithms inspired by biological neural networks and are used to learn or estimate models that have multiple inputs. Artificial neural networks are generally presented as systems of interconnected "neurons", several layers of which may be used in the learning process. These models are used in both supervised and unsupervised learning.

One of the most known neural network clustering algorithm is the Self-Organizing Maps (SOM) introduced by Kohonen [82, 84]. It has been and still is widely used for unsupervised classification and visualization of multidimensional data sets. The fundamental principle of the Self-Organizing Maps is to represent and describe the data using a set of prototypes in a process called *vector quantization*. In short the SOM algorithm creates a layer of interconnected prototypes (that play the role of neurons) that can be used to represent or visualize the structures of a data set.

In this section we introduce the Self-Organizing Maps algorithm and its probabilistic equivalent the Generative Topographic Map algorithm (GTM) [18].

6.2.1 The SOM Algorithm

The SOM algorithm introduced by Kohonen [84] is a vector quantization algorithm and data visualization method that is widely used as a first step for two-step clustering. As we will explain later, the SOM algorithm generates a layer of interconnected prototype

on which another clustering algorithm can by applied to obtain the final clustering result with the desired number of clusters.

Self-Organizing maps are part of non-linear unsupervised methods using artificial neurons as prototypes to recognize and learn from data. It abides by the principle of *"Winner takes all"* in which each observation will activate a single neuron (prototype) that will be reinforced while the other neurons will be inhibited. Using this principle each neuron will specialize in recognizing a certain type of data through the learning process.

The main differences between the SOM algorithms and other prototype-based algorithms using the principle of vector quantization such as the K-Means algorithm are the following: First the number of prototypes in a self-organizing map is normally much higher than the number of expected clusters (hence the necessary second step to get the final clusters). Second, the SOM prototypes are linked together into a layer that we call a *"map"*. This map has a discrete topology defined by an undirected graph linking the prototypes (neurons) together. The most common topology for self-organizing is to put the neurons in a two-dimensional grid. This grid is used in a way that neighbor prototypes can influence each other depending on their proximity and can thus be distorted and conveniently used for representation purposes. The influence of a prototype w_i on a prototype w_j is defined by a symmetrical and positive kernel function $\mathcal{K}_{i,j}$ as described in Equation (6.1) where d_1^2 is the squared value of the Manhattan distance (See Table 2.2) computed based on the 2-dimensional coordinates of the prototypes w_i and w_j on the map.

$$\mathcal{K}_{i,j} = \frac{1}{\lambda(t)} \exp\left(-\frac{d_1^2(i,j)}{\lambda^2(t)}\right) \tag{6.1}$$

$\lambda(t)$ is a temperature decay function that reduces the influence of the farthest neurons of the map (see Figure 6.1) with each iteration t of the algorithm. As it is shown in Equation (6.2), $\lambda(t)$ scales between λ_i and λ_f.

$$\lambda(t) = \lambda_i \left(\frac{\lambda^f}{\lambda^i}\right)^{\frac{t}{t_{max}}} \tag{6.2}$$

Given this context, we use the following notations: let $X = \{x_1, ..., x_N\}, x_n \in \mathbb{R}^d$ be a data set of size N. We note $W = \{w_1, ..., w_C\}, w_i \in \mathbb{R}^d$ the prototypes of the topological map. Each of them is also described by a set of coordinates on the grid. Let $\mathcal{X}(x_n)$ be the function that assigns each data x_n to its closest prototype. Then, the criterion that the SOM algorithm tries to minimize is the following:

$$R(W) = \sum_{n=1}^{N} \sum_{j=1}^{C} \mathcal{K}_{j,\mathcal{X}(x_n)} ||x_n - w_j||^2 \tag{6.3}$$

In Equation (6.3), the function $\mathcal{X}(\cdot)$ can be a simple assignation based on the Euclidean distance $\mathcal{X}(x_n) = \text{argmin}_i ||x_n - w_i||^2$, but it can also be replaced by any kind of distance or even distribution depending on the context.

6.2.1.1 Algorithm

On possible way to minimize Equation (6.3), is to use a simple gradient descent algorithm [5]. The SOM method can then be summed up as shown in Algorithm 11 where eta_t is the learning rate parameter of the gradient algorithm.

Algorithm 11: The SOM Algorithm

Define the map topology (size and shape)
Randomly initialize the prototypes W
do
  forall $x_n \in X$ picked in a random order **do**
    Compute $\mathcal{X}(x_n) = \text{argmin}_i \, ||x_n - w_i||^2$
    forall $w_i \in W$ **do**
      $w_i|_{t+1} = w_i|_t - \eta_t \cdot \mathcal{K}_{i,\mathcal{X}(x_n)}||w_i|_t - x_n||^2$
    end
  end
while $||W_{t+1} - W_t|| > \epsilon$;
return $(\mathcal{X}, W)$

As stated in the introduction, one advantage of this algorithm is that it is useful to visualize the characteristics of a data set. Doing so requires to deform the prototype layer grid based on the prototypes' distances in the data space. This process which results in wrapping the prototypes around the input space is called *mapping the data space*. An example is shown in Figure 6.1.

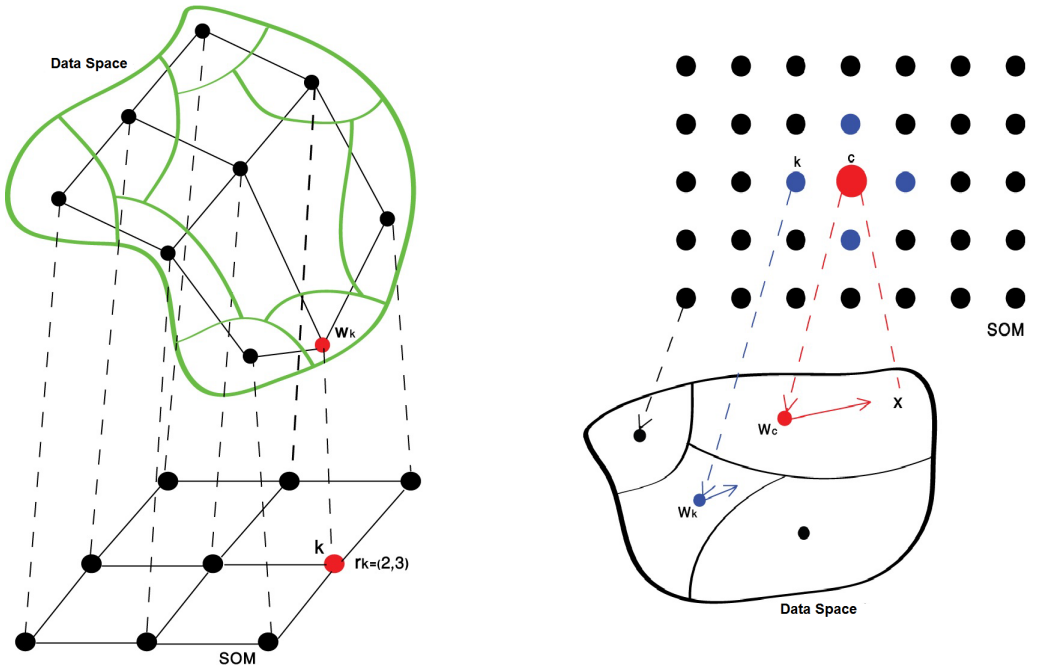


Figure 6.1: Architecture of SOM: Mapping the data space to the self-organizing map (left). Prototypes update (right). [60]

As stated before, creating the map based on the prototypes W is only a first step of a larger process. Indeed, the number of prototype is usually much higher than the expected number of clusters to be found. In short, in addition to provide a visualization method what the SOM algorithm does is providing a large set of micro-clusters in which each prototype of the map is a representative for a micro-cluster and is linked to a subset

of the data. By the end of the learning process the prototypes produced by the SOM algorithm are a fair representation of the data set structure.

The second step to extract the real clusters from the newly acquired prototype consists in using a regular clustering method applied to the map's prototypes. This can be done using very simple methods such as the K-Means algorithm or hierarchical clustering methods that are eligible due to the low number of prototypes compared with the original number of data. Each prototype will then be linked to a cluster of the final partition, and each data can therefore be linked to a cluster too through the prototype to which it is attached.

This kind of two-step approach, with a first step to find a reduced representation of the data and a second step to extract the clustering, is part of what made SOM a popular algorithm. Due to its non-linear nature, the SOM algorithm often leads to better results than simpler algorithms like the K-Means [28].

6.2.1.2 Weaknesses

The two-step approach to extract the final clusters using the SOM algorithms is not without faults. One major weakness of this approach is that it is not optimal in a sense that part of the information is lost during the construction of the map. This loss of information is inevitable since there are less prototypes than data and given that the prototypes used in the final clustering live in a two dimensional space (sometimes three). This amount of lost information is usually not without consequences when doing the final partitioning on the prototypes during the second step.

Another issue is that in its original form, the SOM algorithm is ill-adapted to dynamic data that quickly evolve in time. This problem however can be tackled by tweaking the temperature decay function from Equation (6.2).

6.2.2 The GTM Algorithm

6.2.2.1 Model

The Generative Topographic Map [18, 20] were designed as an alternative to the SOM algorithm. Due to its heuristic nature, the SOM algorithm has several deficiencies [83] such as the absence of a cost function, the lack of theoretical basis to choose a learning rate, or the absence of convergence proof. The GTM algorithm addresses most of these issues by proposing a generative model that maps a set of prototypes *from* a low dimensional latent space *into* the data space. For visualization purposes, the mapping can be inversed using Baye's theorem, thus giving access to a posterior distribution in the latent space.

The goal of the GTM latent variable model is to find a mapping of a set of prototypes $Z = \{z_1, ..., z_K\}, z_k \in \mathbb{R}^l$ from a l-dimensional space into their equivalent prototypes $Y = \{y_1, ..., y_K\}, y_k \in \mathbb{R}^d$ in the data d-dimensional space. The mapping is governed by a set of parameters $\boldsymbol{W}$ and could consist for instance in the parameters of a feed-forward neural networks. $\boldsymbol{W}$ would therefore be a matrix of size $d \times M$ containing the weights and biases of the network. Then, given y a point in the data space, and z a point in the latent space, the mapping is defined as shown in Equation (6.4) where Φ is a matrix containing M basis functions $(\phi_1(z), ...\phi_M(z))$.

$$y = f(\boldsymbol{W}, z) = \boldsymbol{W}\Phi(z) \tag{6.4}$$

Traditionally, the GTM algorithm uses symmetric Gaussian basis functions where μ_m is the center of the basis function and σ its variance :

$$\phi_m(z) = \exp\left(-\frac{||z - \mu_m||^2}{2\sigma^2}\right) \tag{6.5}$$

A visual example of a mapping is shown in Figure 6.2 bellow. As we can see, this mapping process is very similar with the latent space wrapping achieved by the SOM algorithm.

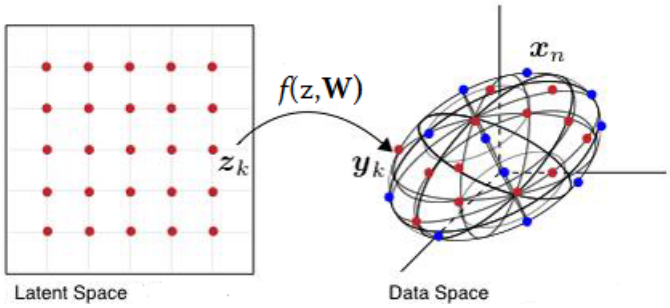


Figure 6.2: Example of mapping of the prototypes from the latent space into the data space. Each node z_k is mapped into the corresponding prototype $y_k = \boldsymbol{W}\Phi(z_k)$ into the data space. The y_k themselves are prototypes used to represent the data X. [20]

Given $X = \{x_1, ..., x_N\}, x_n \in \mathbb{R}^d$ a data set containing N points, the probability distribution of a data point x_n is then defined as follows using a Gaussian distribution where β is the inverse variance:

$$p(x_n|z, \boldsymbol{W}, \beta) = \mathcal{N}(\boldsymbol{W}\phi(z), \beta) \tag{6.6}$$

$$= \left(\frac{\beta}{2\pi}\right)^{d/2} \exp\left(-\frac{\beta}{2}||x_n - \boldsymbol{W}\phi(z)||^2\right) \tag{6.7}$$

Other models are possible for $p(x|z)$. Depending on the context, one may want to use Bernouilli for Binary variables (with a sigmoid transformation of y), or a multinomial for mutually exclusive classes (with a 'softmax', or normalized exponential transformation of y [19]). It is also possible to use combinations of different models.

For a given matrix $\boldsymbol{W}$, the distribution in the data space is obtained by integration over the z-distribution :

$$p(x|\boldsymbol{W}, \beta) = \int p(x|z, \boldsymbol{W}, \beta)p(z)dz \tag{6.8}$$

The parameter matrix $\boldsymbol{W}$ and the inverse variance β can be determined using a maximum likelihood estimation overall the data set by maximizing the log-likelihood given by Equation (6.9).

$$\mathcal{L}(\boldsymbol{W}, \beta) = \sum_{n=1}^{N} \ln p(x_n|\boldsymbol{W}, \beta) \tag{6.9}$$

However, one issue lies in the fact that in most cases, Equation (6.8) will be intractable. This problem can be solved by choosing $f(\boldsymbol{W}, z)$ from Equation (6.4) to be a linear function of $\boldsymbol{W}$. Then, if we mimic the SOM model and choose $p(z)$ to be a set

of K equally weighted delta functions centered on the nodes of a regular grid in the latent space (See Figure 6.3), we have:

$$p(z) = \frac{1}{K} \sum_{k=1}^{K} \delta(z - z_k) \tag{6.10}$$

In this case, each point z_k is mapped to a corresponding point $y_k = f(z_k, \boldsymbol{W})$ in the data space, where each y_k is the center of a Gaussian density function. Furthermore, using Equation (6.10), the integral in Equation (6.8) become tractable using the form shown in Equation (6.11) bellow:

$$p(z|\boldsymbol{W}, \beta) = \frac{1}{K} \sum_{k=1}^{K} p(x|z_k, \boldsymbol{W}, \beta) \tag{6.11}$$

It follows that Equation (6.9) becomes:

$$\mathcal{L}(\boldsymbol{W}, \beta) = \sum_{n=1}^{N} \ln \left(\frac{1}{K} \sum_{k=1}^{K} p(x_n|z_k, \boldsymbol{W}, \beta) \right) \tag{6.12}$$

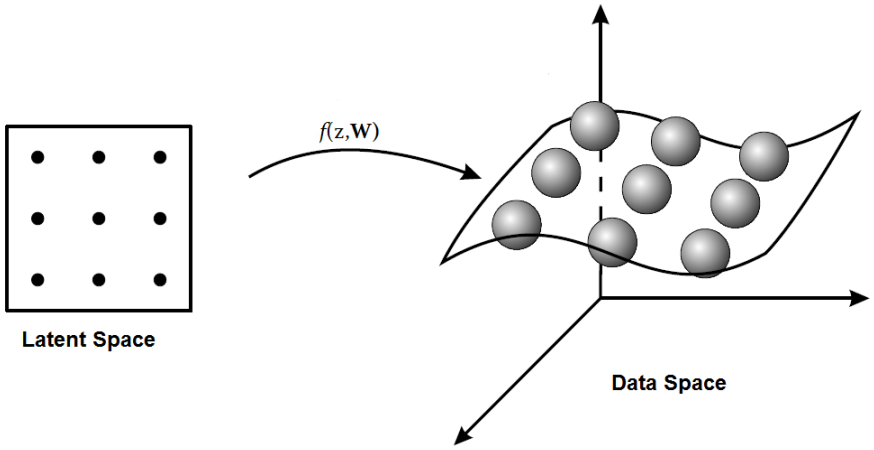


Figure 6.3: Example of mapping of the prototypes from the latent space into the data space that shows the Gaussian distributions around the prototypes y_k in the data space. [20]

6.2.2.2 Algorithm

The maximization of Equation (6.9) can be regarded as a missing data problem where the i^{th} component that generated each point x_n is unknown. This problem can be solved using the EM algorithm [38] (See Section 2.5.1). The E-Step and M-Step for the GTM algorithm are formulated thereafter.

E-Step Let $R = (r_{i,n})_{K \times N}$ be the responsibility matrix containing the posterior probabilities that each data x_n belongs to a Gaussian component i. Using Bayes theorem, these probability are computed during the E-Step as shown in Equation (6.14) where $Z(n)$ is a normalization constant: $Z(n) = \sum_{i=1}^{K} \exp\left(-\frac{\beta}{2}||x_n - \boldsymbol{W}\phi(z_i)||^2\right)$.

$$r_{i,n} = p(z_i|x_n, \boldsymbol{W}_{old}, \beta_{old}) \quad (6.13)$$

$$= \frac{1}{Z(n)} \exp\left(-\frac{\beta}{2}||x_n - \boldsymbol{W}\phi(z_i)||^2\right) \quad (6.14)$$

M-Step We now consider the expectation of the complete data log-likelihood as shown bellow:

$$\mathbf{E}[\mathcal{L}_{comp}(\boldsymbol{W}, \beta)] = \sum_{n=1}^{N}\sum_{i=1}^{K} r_{i,n} \ln\left(p(x_n|z_i, \boldsymbol{W}, \beta)\right) \quad (6.15)$$

The parameters $\boldsymbol{W}$ and β can then be estimated by maximizing Equation (6.15). In particular, the weight matrix $\boldsymbol{W}$ is updated by solving the system proposed in Equation (6.16) where Φ is the $K \times M$ matrix of basis functions so that $\Phi_{i,j} = \phi_j(z_i)$, R is the $K \times N$ responsibility matrix, X is the $N \times d$ matrix containing the data, and G is a diagonal matrix so that $g_{i,i} = \sum_{n=1}^{N} r_{i,n}$.

$$\Phi^T G \Phi \boldsymbol{W}_{new}^T = \Phi^T R X \quad (6.16)$$

The variance β^{-1} is updated following the regular method and using the newly computed matrix $\boldsymbol{W}$:

$$\beta^{-1} = \frac{1}{ND}\sum_{n=1}^{N}\sum_{i=1}^{K} r_{i,n}||x_n - \boldsymbol{W}^{new}\phi(z_i)||^2 \quad (6.17)$$

The GTM algorithm is summed-up in Algorithm 12 bellow:

Algorithm 12: The GTM Algorithm

```
Generate the latent grid Z
Initialize the basis function centres {μ_m}_{m∈[1..M]}, pick σ, compute the matrix Φ
Initialize W randomly or using a PCA, initialize β
Define the map topology (size and shape)
Randomly initialize the prototypes W
do
    E-Step
    Compute R using Equation (6.14)
    M-Step
    Update W using Equation (6.16)
    Update β using Equation (6.17)
while ||W_{t+1} − W_t|| > ε;
```

Very much like the SOM algorithm, the GTM algorithm usually uses a latent grid Z in a low dimensional latent space (2 or 3) for visualization purposes. Likewise, the number of prototypes in this latent space is almost always higher than the expected number of clusters and the algorithm outputs a map Z and not a clustering result. As a consequence, the same 2-step technique is applied to get the final clusters: another clustering algorithm (K-Means or preferably EM) is run on the prototypes from the latent spaces. The responsibility matrix R that makes the link between the data and the prototypes can then be used to pinpoint the final clusters to which each of the prototype will be linked in the exact same way than for the SOM algorithm.

6.2.3 Comparing SOM and GTM

There are several studies available to compare the SOM and GTM algorithms [20, 9, 139]. The general consensus among these studies is that the GTM algorithm has the advantage of having a generative model and a convergence proof thanks to the EM algorithm. The GTM method also yields smaller quantization errors [84]. However, the SOM algorithm yields less topological errors [140], which tends to result in a better final clustering at the end of the second step when extracting the clusters from the topological map. It is commonly accepted, that while the SOM algorithm have less topological errors, the generated topological maps from both algorithms look similar. Both algorithms have a similar speed of training.

In table 6.1, we highlight more differences between the two methods.

	SOM	GTM
Latent space representation	Nodes $\{w_j\}_{j=1}^{C}$ in a L-dimensional array, held together by a neighborhood function	Point grid $\{z_i\}_{i=1}^{K}$ that keeps its topology via smooth mapping
Definition of the manifold in the data space	Indirectly by the location of the prototypes	Continuous mapping with the function f
Objective function	No	Yes : the log-likelihood
Self-organization	Difficult to assess	The smooth mapping preserves the topology
Convergence	No warranty	Yes, using the EM algorithm proof
Smoothness of the manifold	Depends on the neighborhood function $\mathcal{K}$	Depends on the basis functions parameters and the prior distribution $p(x)$
Generative model	No	Yes
User input parameters	$\mathcal{K}$ and size of the map	size of the map
Magnification factors	Approximated by the difference between the prototype vectors	Exactly computable anywhere

Table 6.1: Comparison between SOM and GTM [50]

6.3 From Horizontal to Vertical collaborative clustering with the GTM architecture

In this section, we explain how the architecture from the GTM algorithm can be used to adapt our horizontal collaboration framework from the previous chapter (see section 4.3) to the case of vertical collaboration where several algorithms are working on different data with similar distributions.

6.3.1 Vertical collaborative clustering using the GTM algorithm

Let us consider several data sets $X^1, ..., X^J$ that do not contain the same number of data, but are in the same feature spaces. Their other common point is that the data in each data sets have similar distributions and consequently that the same clusters should be found in all of them.

Each of these data set is processed by either the SOM algorithm or the GTM algorithm resulting in the production of an equal number of topographic maps $W^1, ..., W^J$. Under the hypothesis that all topographic maps have the same number of prototypes, underwent the same initialization, if we suppose that the different data sets have similar distributions, the maps from the different algorithms should be similar, and as a consequence the prototypes outputted by the different algorithms can be seen as a data set the attributes of which have been split between the different SOM or GTM algorithm maps. Therefore, since each prototype has a unique equivalent in each other topographic map, we can apply the collaborative framework for Heterogeneous algorithms.

Our idea here is to apply the previously proposed collaborative framework to the second step of the GTM or SOM algorithm: the clustering of the final prototypes using the EM algorithm. To do so, we use the prototypes vectors $\mathbf{W} = \{W^1, ..., W^J\}$ as input data sets for our collaborative model. A simple example of this idea is shown in Figure 6.4 where we have 3 results in the form of 3 small topographic maps generated by the GTM algorithm.

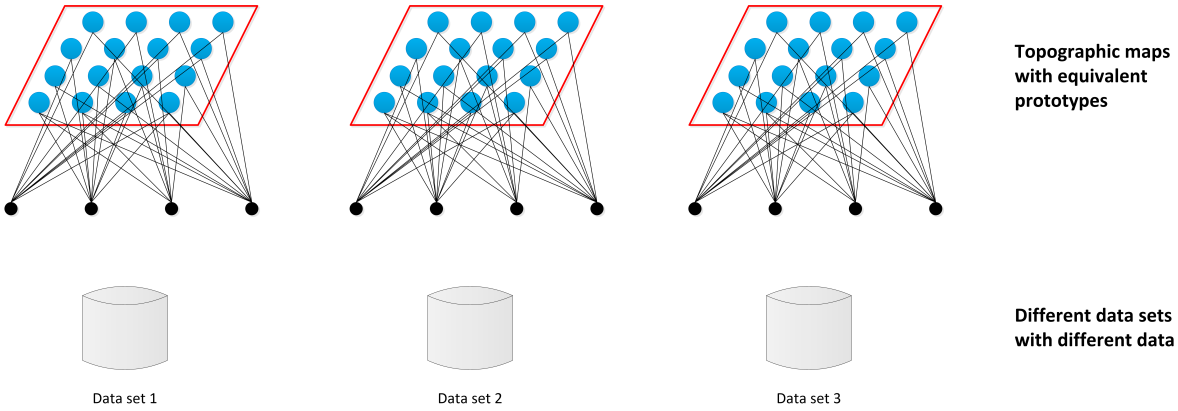


Figure 6.4: From different data sets to similar prototypes

Based on a collaborative version of the EM algorithm using our earlier proposed framework, the transfer learning algorithm with Generative Topographic Maps using Collaborative Clustering is described in Algorithm 13. Figure 6.5 is an illustration of

the kind of result we can expect from this framework applied to topographic maps. The colors highlight the clusters found during the second step.

Algorithm 13: Vertical Collaborative Clustering using GTM : V2C-GTM

Data transformation
forall Data sets X^i **do**
  Apply the regular GTM algorithm on the data X^i.
  Run a first instance of the EM algorithm on the prototypes $\mathbf{W^i}$
end
Retrieve the prototypes $\mathbf{W^i}$ and their clustering labels S^i
Local step:
forall clustering algorithms **do**
  Apply the regular EM algorithm on the prototypes matrix $\mathbf{W}$.
  $\rightarrow$ Learn the local parameters Θ
end
Collaborative step:
while the system global entropy is not stable **do**
  forall EM algorithms $\mathcal{A}_i$ **do**
    forall $w_q \in \mathbf{W^i}$ **do**
      Find s_q^i that maximizes the collaborative EM log-likelihood
    end
  end
  Update the solution vectors S
  Update the local parameters Θ
end

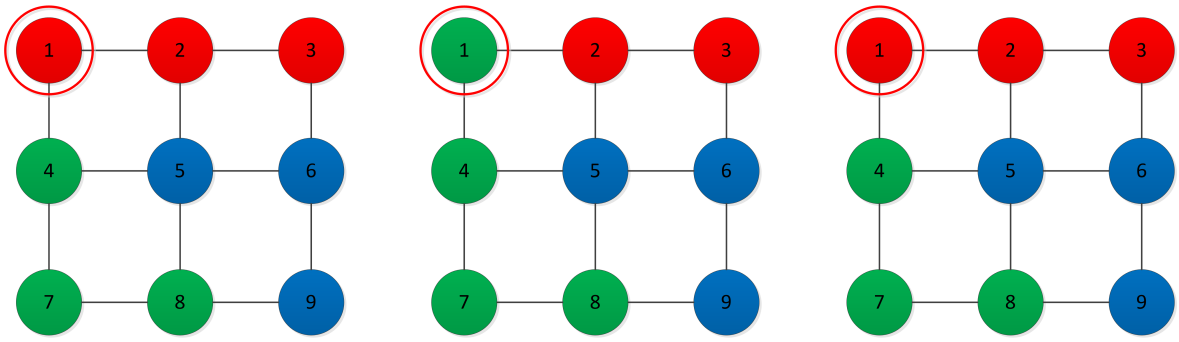


Figure 6.5: Example of 3 collaborating topographic maps. Since they had the same initialization and are used on data that are assumed to have similar distributions, the neurons are equivalent from one map to another. This simple example shows a conflict on the cluster associated to the first neuron. Using our collaborative method, the first neuron will most likely be switched to the red cluster in the second map. With bigger maps, more algorithms and more clusters, conflicts will be more difficult to resolve than in this simple example.

6.3.2 Experimental results

As criteria to validate our approach we consider the purity (accuracy) index of the map which is equal to the average purity of all the cells of the map. A good GTM map should have a high purity index.

The cells purity is the percentage of data belonging to the majority class. Assuming that the data labels set $L = \{l_1, l_2, ..., l_{|L|}\}$ and the prototypes set $C = \{c_1, c_2, ..., c_{|C|}\}$ are known, the formula that expresses the purity of a map is the following:

$$purity = \sum_{k=1}^{|C|} \frac{c_k}{N} \times \frac{max_{i=1}^{|L|} |c_{ik}|}{|c_k|} \tag{6.18}$$

where $|c_k|$ is the total number of data associated with the cell c_k, and $|c_{ik}|$ is the number of data of class l_i which are associated to the cell c_k and N - the total number of data.

In addition to the purity criterion, we also used the Adjusted Rand Index as an external index, and the Davies-Bouldin Index as a clustering index.

6.3.2.1 Raw results

The experimental protocol was the following: all data sets were randomly shuffled and split into 5 subsets with roughly equivalent data distributions in order to have the topographic maps collaborating between the different subsets.

First, we ran the local step, to obtain a GTM map for every subset. The size of all the used maps were fixed to 12×12 for the SpamBase and Waveform data sets and 4×4 for the wdbc and Madelon data sets. Then we started the collaborative step using our proposed collaborative framework with the goal of improving each local GTM by exchanging based on the maps found for the other subsets. We evaluated the maps purity, the Adjusted Rand index of the final cluster, and the Davies-Bouldin Index of the clusters, based on the new GTMs after collaboration.

The results are shown in Table 6.2. Improved results and results that have not been deteriorated during the collaborative process are shown in bold.

As one can see, the results are different depending on the considered indexes. Overall our proposed method gives good results at improving the Adjusted Rand Index with excellent performances on all data sets except for the wdbc data set. The results for the purity index are also very satisfying with a post-collaboration improvement for more than 50% (12/20) of the data sets sub-samples. The results on the Davies-Bouldin index are more contrasted with only 11 cases out of 20 when the internal index remains stable or improves. These results are similar with those of other works on collaborative learning and highlight that while the goal of a general improvement of all collaborators is usually difficult to achieve, the average results' improvements remains positive.

Furthermore, our main goal was to take into account distant information from other algorithms working on similar data distribution and to build a new map. This procedure being unsupervised, it can deteriorate the different quality indexes when collaborating with data sets the distributions of which do not exactly match between each other, or simply when the quality of their proposed maps is too low.

6.3.2.2 Comparison with other algorithms

In this section we compare our algorithm to the vertical version of the collaborative clustering using prototype-based techniques (GTM_{Col}) introduced in [51]. While the two methods may seem similar, there are some major differences: 1) In our proposed method the collaboration occurs after building the maps, while in the GTM_{Col} the collaboration occurs while building the maps. 2) In our method the collaborations is

Dataset	Map	Purity	ARI	DB index
SpamBase	GTM_1	51.1%	0.2	2.15
	GTM_2	53.3%	0.17	1.87
	GTM_3	58.4%	0.12	1.72
	GTM_4	64.89%	0.38	1.47
	GTM_5	75.97%	0.61	0.91
	GTM_{col1}	**59.8%**	**0.3**	**1.68**
	GTM_{col2}	**59.2%**	**0.27**	**1.65**
	GTM_{col3}	57.8%	**0.12**	1.77
	GTM_{col4}	**65.58%**	**0.45**	**1.23**
	GTM_{col5}	68.43%	0.52	1.09
WDBC	GTM_1	62.66%	0.32	1.37
	GTM_2	67.65%	0.37	1.29
	GTM_3	73.78%	0.48	0.94
	GTM_4	61%	0.35	1.48
	GTM_5	56.13%	0.241	1.63
	GTM_{col1}	58.66%	0.258	1.56
	GTM_{col2}	67.45%	0.36	1.34
	GTM_{col3}	71.62%	0.462	1.12
	GTM_{col4}	**63.12%**	**0.374**	**1.38**
	GTM_{col5}	**62.45%**	**0.369**	**1.44**
Madelon	GTM_1	51%	0.22	13.35
	GTM_2	56.5%	0.27	15.25
	GTM_3	52.5%	0.245	12.16
	GTM_4	50.75%	0.209	11.56
	GTM_5	50.25%	0.2	11.69
	GTM_{col1}	**51%**	**0.223**	**13.35**
	GTM_{col2}	55.5%	**0.27**	15.71
	GTM_{col3}	**52.5%**	**0.245**	**12.16**
	GTM_{col4}	**56.25%**	**0.257**	14.82
	GTM_{col5}	**51.5%**	**0.234**	14.05
Waveform	GTM_1	67.25%	0.46	1.54
	GTM_2	72.12%	0.58	1.27
	GTM_3	74.28%	0.61	1.22
	GTM_4	69.47%	0.507	1.49
	GTM_5	71.09%	0.564	1.3
	GTM_{col1}	**67.79%**	**0.472**	**1.46**
	GTM_{col2}	71.76%	**0.62**	**1.27**
	GTM_{col3}	72.59%	0.59	1.25
	GTM_{col4}	**71.52%**	**0.617**	**1.24**
	GTM_{col5}	**71.1%**	**0.603**	**1.23**

Table 6.2: Experimental results of the horizontal collaborative approach on different data sets

simultaneously enabled between all algorithms, while GTM_{Col} only enables pairwise collaborations. Given these two differences the results that we show thereafter have to be taken with caution: while the two methods have the same goals and applications, they are very different in the way they work.

In Table 6.3, we show the comparative results of the average gain of purity measured before and after collaboration.

We can clearly see that while both methods give mild performances at improving the purity of a GTM map for our algorithm and a SOM map for the GTM_{Col} method, our algorithm is always positive on average for all data sets and our global results are also slightly better.

It is obvious that our proposed V2C-GTM method outperforms other methods by increasing every time the accuracy index after the collaboration step. Even, if for Madelon dataset, the purity index after the collaboration is higher for the GTM_{Col} and SOM_{Col} methods, we have to note here that for these indexes the accuracy gain depends on the collaboration parameter β which is fixed in the algorithm (the higher this parameter is, the higher the distant collaboration will be used in the local learning process).

Another important aspect of the GTM and SOM based collaboration methods is that these approaches can attempt collaboration only between two collaborators in both direction which explain the $\pm$ in the results (without having an a priori knowledge about the quality of the collaborators the accuracy gain can be positive or negative). We note here that the proposed V2C-GTM approach can use several distant information from several collaborators without fixing any collaboration parameters and usually the accuracy gain is positive.

Dataset	Purity		
	V2C-GTM	GTM_{Col}	SOM_{Col}
SpamBase	+1.43%	-2.31%	-2.4%
WDBC	+0.416%	-2.45%	±0.32%
Madelon	+1.15%	+2.85%	+2.1%
Waveform	+0.11%	+0.07%	±2.6%

Table 6.3: Comparison of the average gain of purity before and after collaboration using our V2C-GTM algorithm, the GTM_{Col} algorithm and the SOM_{Col} algorithm

These results are quite interesting because unlike the GTM_{Col} method that was specifically thought and developed with the idea of using it with semi-organized maps or generative topographic maps, the collaborative framework that we use was thought to be as generic as possible and not particularly adapted to the GTM algorithm.

The conclusion we could draw from these results is that perhaps the probabilistic approach used by our framework is more effective than the derivative approach used in the other methods.

6.4 Discussion

In this Chapter, our goal was to propose a solution to adapt our collaborative method from Chapter 4 to the case of vertical collaboration where several algorithms work on different data sets having similar distributions and clusters.

To this end, we have proposed a strategy in which the architecture of clustering methods such as the self-organized maps (SOM) or generative topographic maps (GTM) is used in an original way to transform a vertical collaboration problem into an horizontal one. To do so, we use the data quantization properties and topological properties of both methods so that the topographic maps extracted after using several GTM (or SOM) algorithms on data that have similar distributions can be seen as a multi-view data set, and therefore be processed by our horizontal collaboration framework.

One could argue that the quantization process makes our proposed method cumbersome to use and does not offer a generic framework for vertical collaboration and transfer learning. However, our method has shown to outperform already existing algorithms designed for the same purpose.

Chapter 7

Conclusion and future works

Contents

7.1 Summary of the contributions

The aim of this thesis is to provide original solutions and new methods in order to improve collaborative clustering techniques in general, but with a particular focus on applications regarding very high resolution satellite images.

7.1.1 Contributions applied to the clustering of segmented images

The ANR project that funded this PhD thesis provided us with high resolution satellite images that were preprocessed by partner universities resulting in a data set with very specific features including: a relatively large number of heterogeneous attributes, and geographical dependencies that include highly irregular neighborhood features.

The first contribution of this PhD thesis was to create and implement a clustering algorithm that could efficiently process such a data set. In particular, we focused on developing a clustering algorithm that was able to use the geographic dependencies of data sets acquired from the segmentation of satellite images.

We did so by proposing an adaptation of the Iterated Conditional Modes (ICM) algorithm, originally developed for the clustering of pixel-based data sets. Our proposal, the Semantic-Rich ICM algorithm adapts the original algorithm to the case of region-based data sets with irregular neighborhood dependencies.

Our proposed method is a fork of the ICM algorithm which is often considered to be a deprecated algorithm. But the ICM has a lower computational complexity than more sophisticated algorithms and provides fair solutions. For example, algorithms such as the "Graph-cuts" wouldn't be able to handle the large size of our images data sets coupled with the high complexity of the neighborhood dependency graph.

Our adaptation of the ICM algorithm relies on a modification of the Markov Random Field original energy formula so that it can accommodate data that have different numbers of neighbors and these neighbors have a weighted influence depending on the proportion of the border they occupy. Our energy model proved to be competitive when compared with other energy models both in the region-based data set "VHR Strasbourg", and some pixel based data sets that we used for comparison purposes.

Furthermore, our proposed SR-ICM algorithm relies on a neighborhood compatibility matrix that adds some low level semantics regarding the relations between the different clusters of the data set, and their relative compactness on the analyzed image.

7.1.2 Contributions to collaborative clustering

The second and the main part of this PhD thesis deals with our contributions to collaborative clustering. As we see it, collaborative clustering is an emerging subject that exists under many names and forms in the literature: multi-view clustering, clustering of distributed data, ensemble clustering, and so on. All the frameworks and methods developed under these names pursue the similar goal of having several algorithms working together with a goal of mutual improvement of their solutions, eventually prior to a merging of the said solutions.

Our first contribution was to show that all these different methods follow the same principle and can be regrouped and summed up under the single framework of collaborative clustering.

Our second contribution to collaborative clustering is concerned with alleviating the weaknesses of the previously proposed framework for the case of horizontal collaboration (several algorithms working on the same data). Most of the previously proposed methods had several limitations that included:

- The impossibility of having different types of algorithms collaborating together.
- The strong requirement to have all collaborators looking for the same number of clusters or prototypes.
- The inability to account for the local model of each algorithm during the collaborative step, thus reducing the collaborative framework to a simple vote system.
- Instability of the method due to the need for user input parameters, or due to an asynchronous system to solve the conflicts between the different collaborators.
- Lack of theoretical background.

The collaborative method that we developed during the course of this PhD thesis saves all the advantages of the previously proposed methods without any of their disadvantages: our proposed frameworks allows heterogeneous algorithms to collaborate together while having no restriction on the number of cluster overlooked by each algorithm, nor its model. Our method works in a synchronous fashion ensuring the stability of the algorithm and does not heavily rely on user input parameters. Furthermore it takes into account the local model of each algorithm during the collaborative step and thus cannot be reduced to a sophisticated voting system. A comparison of our method for horizontal collaborative clustering with other collaborative methods is shown in Table 7.1.

We have studied the theoretical properties of our proposed method for horizontal collaborative clustering: stopping criterion based on the system global entropy, convergence proof, and theoretical results on the optimal weighting of the collaborators.

Our method was successfully tested on several data sets, including the very high resolution satellite image data set from the ANR project, as well as a large range of data sets from the UCI website. The results have shown the strength of our approach and its efficiency.

Beyond the method that we introduced for horizontal collaborative clustering, our third contribution is an insight on criteria that may determine the result of a collaboration. Our conclusion to that matter comes from both theoretical and empirical results and is that the parameters to set up a good collaboration are highly dependent on the criterion that one wants to optimize: the global likelihood of the solutions will be maximized when favoring the collaboration between algorithms the solutions of which have a low diversity. However, optimizing a specific internal or external criterion seems to require to collaborate only with algorithms that show good results for the said criterion and to favor high diversity between the solutions.

Finally, this Thesis' last contribution to collaborative clustering was the proposal of a method that uses the architecture of unsupervised neural networks such as the GTM algorithm or the SOM algorithm with the goal of transforming a vertical collaboration problem into an horizontal one, thus allowing us to use the framework that we developed early on.

Our proposed architecture was successfully tested on various data sets and compared with other vertical collaborative methods. Our results were positive and better on average than these of the other methods.

	Our HP2CGD [128]	Our V2C-GTM [129, 130]	SAMARAH [147, 44]	Prototype-based methods [101, 50, 55]
Collaboration Type	Horizontal	Vertical	Horizontal (+consensus)	Vertical or Horizontal
Collaborative Principle	PCM	Vector quantization (GTM) and PCM	PCM	Derivation
Conflicts resolution	Simultaneous	Simultaneous	Ordered and asynchronous	Simultaneous
Heterogeneous algorithms	Yes	Yes[1]	Yes	No
Number of clusters	No restrictions, can decrease	No restrictions, can decrease[2]	No restrictions, can increase or decrease	Needs to be and to stay identical

Table 7.1: Comparison between the different collaborative methods

7.1.3 Implementation

The different algorithms and methods proposed in this PhD thesis have been implemented and tested using the C++ language, coupled with R and Matlab when specific libraries or functions were needed.

The main program developed during this thesis was a Qt/C++ software that included the following elements:

- A complete implementation of several clustering algorithms: K-Means, Fuzzy C-Means, EM for the GMM, SOM (batch version), DBSCAN, ICM (pixel and region based version).
- Evaluation indexes both supervised and unsupervised: Silhouette index, Davies-Bouldin index, Giny index, Rand index, Adjusted Rand index, confusion entropy.
- Our proposed methods: C-ICM, SR-ICM, our collaborative framework for horizontal collaboration (handling up to 5 algorithms with the user interface), the second step of the V2C-GTM algorithm.
- Various miscellaneous methods and algorithms including: an implementation of the stability for the EM, Fuzzy C-Means, K-Means and ICM algorithms, various optimization techniques for the computation of the combination functions

[1] The GTM algorithm is mandatory during the first step, but any clustering algorithm can be used during the collaborative clustering of the obtained maps.

[2] The number of prototypes in the GTM, and their configuration however must be identical.

7.2 Short term prospects

Prospects in imaging: When it comes to our proposed clustering algorithms for preprocessed very high resolution satellite images, we can certainly regret the lack of available data sets during this PhD thesis. Among the short term perspectives that we could investigate, it would be interesting to find similar data sets and to see how our proposed algorithm performs on images from other kinds of landscapes and other sources.

Prospect in collaborative clustering: Several possibilities lie ahead for our work in collaborative clustering. Among them, since we have proposed several possible combination function for our collaborative framework, it would be interesting to conduct a further study on their properties, advantages, disadvantages and best use cases. We already have early results for such a study with some functions giving better results than others depending on the attributes the different collaborators have access to. It would be nice to get theoretical results.

Another possible perspective would be to improve our vertical method. Our proposal was to use the SOM and GTM architecture as a first step to transform a vertical collaboration problem into a horizontal one. However during our experiments, we used the original version of the GTM algorithm. It could be interesting to see if we could get better results by using the collaborative vertical version of the GTM algorithm [50] instead of the regular GTM algorithm.

7.3 Long term limitations and prospects

The most interesting long term perspectives for this thesis would be to search further on the subject of the ideal collaboration strategy.

Several earlier works have studied the influence of quality and diversity in collaborative clustering, and the notion of stability has been around for a while in regular clustering. While our work has clarified the role of quality, diversity and stability, it seems to us that we have barely scratched the surface when it comes to find ideal collaboration strategies. The question of the ideal collaboration strategy to optimize a given variable is still open, and our initial results have shown to be divergent.

For a time, we sought to use the clustering notion of stability. However, as it is defined by Shai Ben-David [11] the theory of clustering stability requires the existence of a unique optimal solution. In the case of collaborative clustering, since we have several algorithms working together there is no such thing as an optimal solution. The best we can get is a Pareto front between several sub-optimal solutions from the different algorithms. A long term prospect would therefore be to rethink the notion of stability applied to the case of collaborative clustering.

Another interesting lead would be to study the problem of the optimal strategy from a "*multi-armed bandit problem*" perspective [88, 4]. It seems to us that the ideal strategy may be impossible to guess in advance without trying several. And since the evaluation cost of a strategy is very high in collaborative clustering (one needs to compute the results of the iteration), using a multi-armed bandit problem strategy to update the collaboration weights during the collaborative process could be an original and interesting solution allowing to explore and exploit new strategies during the iterations of collaborative framework.

At last, an essential number of essential questions remain when it comes to the scientific positioning of collaborative clustering when compared with the seemingly larger context of *transfer learning*. This field also is an emerging domain within unsupervised learning, and both share a lot of common questions: how to transfer knowledge? What to transfer? With whom to share knowledge? What is the influence of diversity? and so on.

It is very possible that transfer learning, that we have implicitly named "vertical collaboration" in this thesis as well as in earlier works, actually is the larger field rather than the opposite. And in any case, it is more than likely that the theoretical findings in the field of collaborative clustering can be generalized in the field of transfer learning.

Appendix A

Résumé en français

A.1 Contexte

Cette thèse portant sur le clustering collaboratif incrémental a été co-encadrée par les Professeurs Antoine Cornuéjols (AgroParisTech) et Younès Bennani (Université Paris 13), et se situe dans le cadre du projet ANR COCLICO (COllaboration, CLassification, Incrémentalité et COnnaisances), ANR-12-MONU-0001.

Un des objectifs de ce projet était de proposer des algorithmes et des méthodes capables d'analyser des données issues d'images satellites à très haute résolution. Suivant cette idée, cette thèse est divisée en deux axes principaux :

- Le premier axe de la thèse concerne la partie clustering de données d'imagerie. En particulier, cette thèse propose une adaptation des modèles à base de champs aléatoires de Markov (MRF en anglais) applicable à des images qui ont déjà été segmentées et pour lesquelles un clustering doit être effectué sur des segments obtenus dont les voisinages sont très irréguliers ce qui rend l'application des modèles classiques difficile.

- Le deuxième axe porte sur le clustering collaboratif. Dans cette seconde partie de la thèse, l'objectif était de proposer une framework générique applicable à autant d'algorithmes que possible, ce qui incluait les algorithmes développés dans le premier axe. Les méthodes collaboratives développées lors de cette thèse peuvent traiter les deux sous-cas suivants :

  - Plusieurs algorithmes travaillant ensemble sur les mêmes objets dans des espaces de représentations différents.

  - Plusieurs algorithmes travaillant sur des données différentes mais étant dans les mêmes espaces de représentation et ayant des distributions similaires.

  Les frameworks ainsi développés au cours de cette thèse se veulent les plus génériques possibles. Ainsi, dans le cas du framework développé pour le sous-cas 1), il est applicable à n'importe quels algorithmes de clustering.

A.2 Résumé de la thèse

A.2.1 Apprentissage automatique et clustering

A.2.1.1 Apprentissage automatique

L'apprentissage automatique (ou *machine learning* en anglais) est un domaine de recherche lié à l'informatique et aux mathématiques appliquées, dont l'objectif est de permettre à une machine ou à un programme d'apprendre sans être explicitement programmé pour celà. C'est donc une science qui consiste à construire des algorithmes et des méthodes capables d'apprendre à partir de données avec divers objectifs tels que pouvoir classer les données, identifier des structures dans les données, ou encore faire des prédictions. L'apprentissage automatique est utilisé dans de nombreux domaines scientifiques tels que la biologie et la médecine [135, 85, 32, 162, 155], les mathématiques [158], la finance et le marketing [22, 153, 23], la physique [8], la chimie [133], et bien d'autres. On distingue généralement trois types d'applications pour l'apprentissage automatique :

- L'apprentissage supervisé, où la machine apprend à partir d'exemples labellisés dans le but de construire un modèle permettant de faire des prédictions ou de classer des données non-labellisées.

- L'apprentissage non-supervisé, où les données sont fournies sans étiquettes. La machine cherche alors à détecter des structures ou à faire des groupes d'éléments similaires.

- L'apprentissage par renforcement, où une machine placée dans un environnement dynamique va apprendre à effectuer une tâche donnée grâce à un système de "récompenses et pénalités" basé sur ses actions.

A.2.1.2 Le clustering

Cette thèse s'intéresse à l'apprentissage non-supervisé et plus particulièrement au clustering (parfois appelé *classification non supervisée*, mais le terme ne fait pas consensus). Le clustering est une tâche d'apprentissage consistant à chercher à faire des groupes d'objets *similaires* à partir de données sans étiquettes. Un cluster se définit donc comme un groupe d'objets similaires au sein d'un jeu de données. Ainsi, on voit immédiatement que la notion de similarité est extrêmement importante, puisqu'elle aura une très forte influence sur les clusters produits par une méthode de clustering.

Afin de définir un algorithme de clustering, la seconde étape une fois une mesure de similarité choisie est de définir une stratégie pour construire les clusters à partir de celle-ci. Le clustering étant un problème mal défini, il n'y a pas de consensus sur la mesure de similarité idéale, et il existe donc un nombre important de mesures de similarités, et problablement encore plus de stratégies de clustering. Il en résulte que de très nombreuses méthodes de clustering ont été créées, toutes avec leurs points faibles et leurs points forts selon les données et les applications.

Les applications du clustering sont très nombreuses : détections de groupes d'utilisateurs ayant des caractéristiques communes, reconnaissance d'écriture, organisation de résultats de recherches web, détection de communauté, détection de fraudes, identification de structures, segmentation de données, etc.

Les algorithmes de clustering peuvent généralement être classés en plusieurs grandes familles selon le principe sur lequel ils sont basés. Certains algorithmes peuvent appartenir à plusieurs catégories :

- Les méthodes basées sur la densité : ces méthodes suivent une idée intuitive basée sur la vision humaine, et selon laquelle les clusters doivent êtres construits en fonction des zones ou les objets sont regroupés de façon plus ou moins dense dans l'espace des données. Ces méthodes ont l'avantage de ne pas assumer de forme particulière pour les clusters, mais sont coûteuses car elles nécessitent des comparaisons pair-à-pair entre les données pour identifier les zones à forte densité. Les algorithmes DBSCAN [41] et OPTICS [2] sont des exemples d'algorithmes de clustering basés sur la notion de densité.

- Les méthodes hiérarchiques [72, 144, 134, 159, 37] : ces méthodes construisent une structure en forme d'arbre dans lequel la racine est un cluster unique contenant toutes les données et les feuilles des données isolées. Les branches (noeuds) de l'arbre sont construites par fusion des noeuds des niveaux supérieurs selon des critères de similarité. L'algorithme CURE [58] est un exemple d'algorithme appartenant à cette famille.

- Les algorithmes basés prototypes : ces algorithmes cherchent à représenter les données à partir d'éléments représentants les structures principales des données et qui sont appelés des "*prototypes*". Ce principe de représentation des données par des prototypes est appelé quantification vectorielle. Ces prototypes peuvent contenir des informations telles que les emplacements de centroïdes des clusters, la variance interne des clusters, ou encore leur forme. Quelques exemples d'algorithmes appartenant à cette famille sont les K-Moyennes (K-Means) [96] (voir Figure A.1), les K-Moyennes floues (Fuzzy C-Means) [14], l'algorithme Expectation-Maximization (EM) pour le modèle de mixture Gaussien [38], ou encore les algorithmes basés sur les réseaux neurones tels que les cartes auto-organisatrice (SOM) [84] ou les cartes topographiques (GTM) [20].

- Les méthodes spectrales : ce sont des méthodes récentes qui se basent sur les valeurs propres de la matrice d'adjacence du graphe représentant les données. L'utilisation de la matrice d'adjacence rend ces algorithmes difficiles à utiliser avec des gros jeux de données. Quelques exemples de méthodes de clustering spectrales sont l'algorithme Shi-Malik [120], ou encore l'algorithme Meila-Shi [97], et leurs dérivés.

- Les méthodes probabilistes : ces méthodes issues des mathématiques font l'hypothèse que les données suivent certaines fonctions de densité de probabilité. L'objectif de ces algorithmes est d'identifier les paramètres de ces fonctions et de définir un modèle de mixture pour représenter les différents clusters. Beaucoup d'algorithmes peuvent être ainsi décrits. On y retrouve les K-Moyennes floues, l'EM pour le modèle de mixture Gaussien, les mixtures de Bernouilli, etc.

Le clustering reste un problème difficile et pour lequel beaucoup de points sont problématiques :

La question de la définition exacte d'un cluster par exemple car aucune formalisation mathématique qui fasse consensus n'existe à ce jour. Il n'y a en effet pas de définition arrêtée sur ce qu'est un bon cluster (ou un mauvais cluster). De part sa nature non-supervisés, les résultats de clustering sont difficiltes à évaluer. Contrairement à la

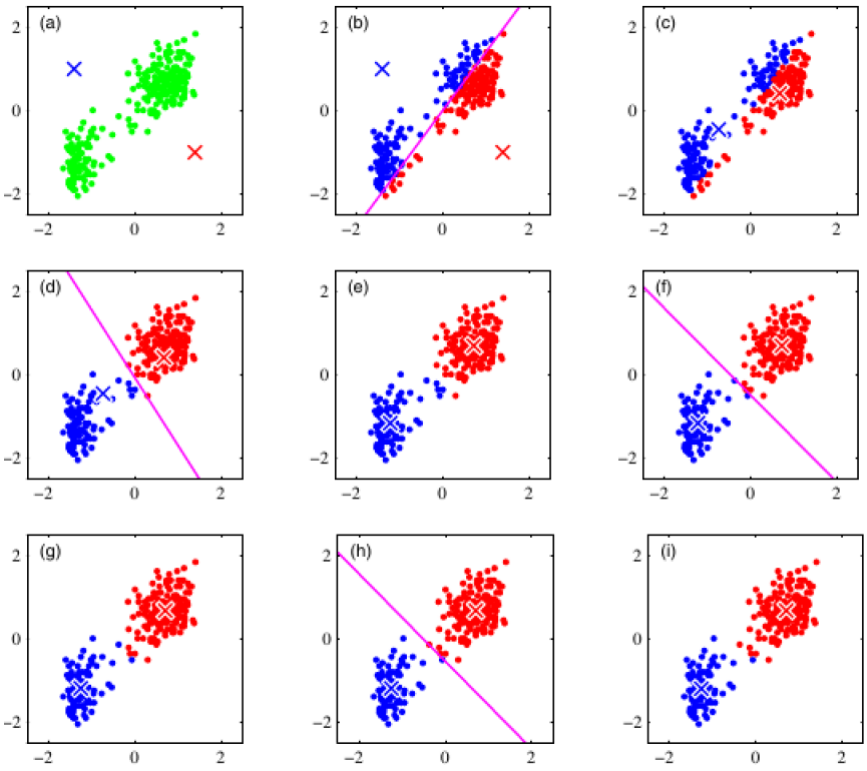


Figure A.1: Exemple de l'algorithme des K-Moyennes utilisé sur le jeu de données "Old Faithful"

classification, il n'existe en effet généralement pas de "table de vérité" permettant de vérifier l'exactitude des solutions. Lorsque de telles tables de vérité existent, il n'est pas toujours concevable qu'un algorithme puisse sans intervention humaine retrouver tout seul un critère de similarité permettant d'avoir des clusters identiques à ceux de la solution attendue. Il existe quelques critères de validation dits *"critères internes"* [98] qui mesurent généralement la séparation inter-cluster et la compacité intra-cluster. Cependant ces critères ne sont pas toujours objectifs ni pertinents pour identifier des objets réels. Il en résulte que comparer deux résultats de clustering est également un problème complexe.

Des problèmes qui peuvent sembler assez simples comme trouver le nombre de clusters présents dans un jeu de données sont également toujours des questions ouvertes. Il n'est à vrai dire déjà pas évident de savoir si oui ou non des structures sont présentes dans un jeu de données et donc s'il y a des clusters à trouver.

A.2.2 Le clustering appliqué à la segmentation d'images

La fouille de données appliquées au image est un domaine de recherche datant des années 80 et qui a de nombreuses applications. L'objectif de ce domaine est d'apprendre à des algorithmes à imiter la vision humaine de manière à pouvoir détecter et éventuellement identifier les différents éléments présents sur une image. La vision assistée par ordinateur comprend généralement plusieurs étapes : l'acquisition des données, le pré-traitement des données (débruitage, encodage, etc.), l'analyse des données, et la validation des résultats. Dans cette section, nous nous intéresserons en partie à la partie pré-traitement des données, mais surtout à la partie analyse des données qui est le coeur d'un chapitre de cette thèse.

A.2.2.1 Segmentation d'image

Du point de vue de l'ordinateur, une image est généralement décrite sous la forme d'une grille de pixels, qui dans le cas le plus commun contiennent 3 canaux (rouge, vert et bleu) dont les valeurs peuvent varier de 0 à 255. Cependant, avec l'avènement de l'imagerie satellite moderne et des images à très haute résolution, l'analyse d'image basée sur les pixels est maintenant peu pratiquée et l'analyse basée régions est généralement préférée [21]. Une région se définit comme un groupe de pixels plus ou moins homogènes regroupés ensemble lors d'un processus appelé *segmentation d'image*. L'objectif d'une bonne segmentation d'image est de regrouper ensemble des pixels connexes afin de représenter au mieux les vrais objets présents sur l'image et que les pixels ne sauraient représenter dans leur intégralité [100, 52].

Les problèmes les plus courants lors d'une segmentation d'image sont la *sur-segmentation* qui résulte en un trop grand nombre petits segments, et la sous-segmentation qui résulte en un trop faible nombre de segments englobant plusieurs objets réels. Le cas de la sous-segmentation est plus problématique dans la mesure où il ne sera pas possible d'identifier les objets dans le cas où plusieurs seraient regroupés dans un même segment. En revanche, dans le cas d'une sur-segmentation, les objets séparés entre plusieurs clusters peuvent toujours être retrouvés en groupant ultérieurement les segments concernés.

A.2.2.2 Modèles de Markov et algorithme ICM

Les modèles de Markov sont des modèles graphiques probabilistes dont l'objectif est de tenir compte des relations géograhiques ou temporelles pouvant lier des données. Dans le cas de l'analyse d'image, on parle de *champs aléatoire de Markov* [79], un modèle de graphe initialement développé pour décrire et utiliser les dépendances entre les pixels d'une image. Des algorithmes tels que l'Iterated Conditional modes (ICM) [13] ou le Graph Cuts [24] peuvent alors être utilisés pour effectuer le clustering de données d'imagerie tout en tenant compte des dépendances entre les données décrites par le réseau de champs de markov.

Dans le cas de l'algorithme ICM auquel nous nous intéresseront particulièrement dans cette thèse, l'objectif est de maximiser une fonction d'énergie composée d'un terme local cherchant à maximiser la vraissemblance des données vis-à-vis des prototypes des clusters de la même façon que dans l'algorithme EM, et d'un terme de voisinage qui pénalise l'énergie en cas de discontinuité des clusters entre pixels voisins [78, 160].

A.2.2.3 Algorithme ICM enrichi

Notre première proposition dans cette thèse consiste en une modification de l'algorithme ICM afin qu'il soit applicable sur des données d'imagerie de type "régions". La différence principale entre les régions et les pixels du point de vue du modèle des champs aléatoires de Markov est la suivante : dans le cas des pixels, chaque pixel a exactement 4 voisins occupant chacun un quart de sa frontière. Dans le cas des régions, à cause des tailles et des formes irrégulières de celles-ci après une segmentation, chaque région peut avoir un nombre très différent de régions voisines, et l'occupation de la frontière par ces régions voisines peut-être potentiellement très déséquilibrée (Voir Figure A.2).

Notre proposition consiste donc en une modification de la fonction d'énergie de l'ICM de manière à pouvoir tenir compte de ces spécificités [126, 125]. Ainsi, nous proposons d'utiliser une matrice de compatibilité des clusters inspirées de celle des

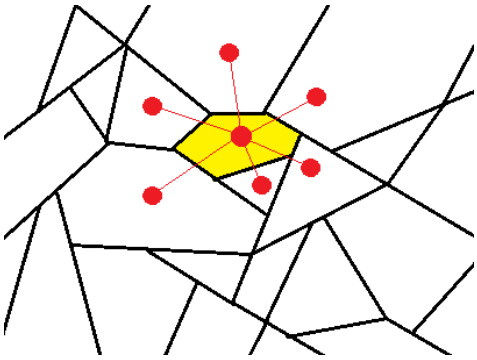


Figure A.2: Exemple de régions et de voisinages irréguliers

chaînes de Markov et qui sera placé dans le terme de voisinage afin de pénaliser plus ou moins les transitions d'un cluster à un autre entre zones adjacentes selon le pourcentage de frontière que les 2 régions partagent.

Un des bénéfices non-attendu de cette proposition est que la matrice de compatibilité entre les clusters calculée *à posteri* après chaque itération de notre algorithme contient des informations sémantiques de bas niveau sur les clusters qui ne sont habituellement pas disponibles avec des algorithmes de clustering ou de segmentation [131]. Entre autres, la matrice de compatibilité de notre algorithme SR-ICM (pour *semantic rich iterated conditional modes*) fournit des informations sur la compacité de chaque cluster sur l'image, et également sur la proximité géographique relative et la compatibilité des différentes régions. Dans le cas de notre application aux images satellites, de telles informations peuvent concrètement se traduire par : "Il y a de larges zones de forêt sur cette image", "les éléments du cluster de l'eau est très rarement adjacents à ceux du cluster des structures urbaines denses", etc.

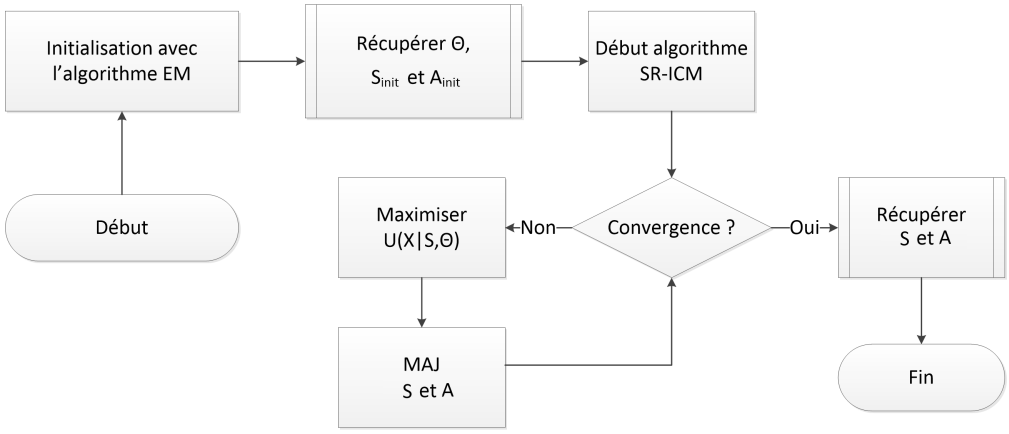


Figure A.3: Schéma de déroulement de l'algorithme SR-ICM

Cet algorithme a été testé sur des données d'imagerie à très haute résolution de la ville de Strasbourg fournies pour le projet ANR [111]. Les résultats obtenus sont très satisfaisants et montrent la qualité de la méthode proposée.

A.2.3 Le clustering collaboratif

A.2.3.1 Introduction

L'idée que collaborer permet d'obtenir de meilleurs résultats qu'en travaillant seul n'est pas nouvelle, et l'intuition que la combinaison de plusieurs sources peut aider à obtenir de meilleurs résultats a été formalisée pour la première fois à la fin du XVIIème siècle par le Marquis de Condorcet [31].

L'idée a naturellement été reprise dans de nombreux domaines, y compris en apprentissage automatique. En particulier, en apprentissage supervisé où il est facile d'évaluer la qualité d'une source d'information, les premières méthodes collaboratives apapraissent dans les années 90 sous le nom de *méthodes d'ensemble* [117, 150, 81]. En apprentissage non-supervisé, la difficulté d'analyse de la qualité des sources fait que les méthodes collaboratives sont apparues plus tardivement. C'est l'arrivée des données distribuées et des données multi-vues sur de gros jeux de données qui a poussé l'émergence des méthodes collaborative pour le clustering.

Le terme *clustering collaboratif* est mentionné pour la première fois par Pedrycz pour une version distribuée de l'algorithme des K-Moyennes floues (CoF-CMeans) [101]. Le problème consistait alors à faire travailler plusieurs algorithmes de clustering sur des bases de données de différentes entreprises décrivant les mêmes clients par des caractéristiques différentes et qui pour des raisons de respect de la vie privée ne pouvaient pas directement échanger leurs informations.

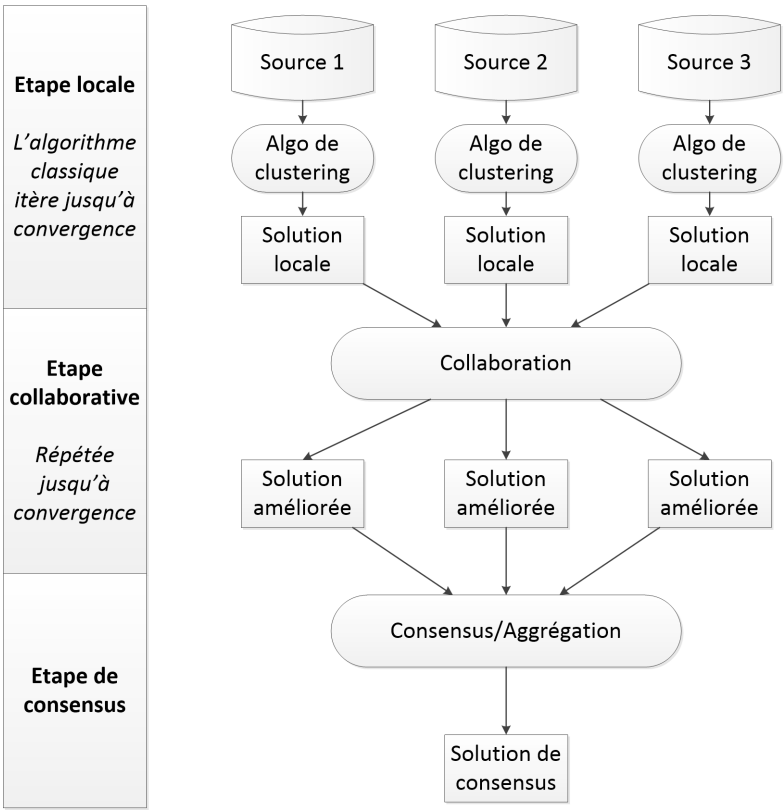


Figure A.4: Clustering Collaboratif : Schéma général

Cependant, de manière générale, le clustering collaboratif ne se limite pas au cas des données distribuées et est appliqué sous divers noms pour une large gamme de problèmes:

- Le clustering multi-vues : Dans ce cas plusieurs algorithmes s'intéressent à différentes représentations souvent redondantes des mêmes données [15, 161], parfois même avec des attributs de natures différentes.

- Le clustering de données en grande dimension : Un problème récent et d'actualité avec l'avénement du Big Data. On cherchera ici à distribuer artificiellement soit les attributs, soit les données entre plusieurs algorithmes afin de les faire travailler en parallèle sur des sous-échantillons, réduisant ainsi la complexité algorithmique.

- L'analyse multi-experts ou multi-échelles : Avec des jeux de données toujours plus complexes, il est parfois nécessaire d'essayer plusieurs algorithmes plus ou moins spécialisés afin de pouvoir espérer de bons résultats. Il est alors utile de les faire travailler ensemble afin qu'ils partagent leurs connaissances. Un autre cas d'utilisation est le cas multi-échelles. En effet, il y a parfois plusieurs niveaux hiérarchiques de structures remarquables et de clusters à trouver dans un même jeu de données [49]. Dans ce cas aussi l'analyse collaborative peut aider plusieurs algorithmes cherchant des clusters à des échelles différentes à trouver une cohésions entre les structures à différents niveaux. Ce type d'analyse est particulièrement pertinent avec des données d'imagerie comme celles utilisées dans cette thèse.

Un des apports de ce document est l'affirmation que tous ces cas d'utilisation peuvent en fait être regroupés sous le seul paradigme du clustering collaboratif selon le schéma décrit dans la figure A.4.

Les grandes problématiques du clustering collaboratif sont les suivantes :

- Comment collaborer ?

- Avec qui collaborer ?

- Faut-il collaborer ?

- Comment quantifier l'apport d'une collaboration ?

- Peut-on prédire l'apport d'une collaboration ?

A.2.3.2 Etat de l'art en clustering collaboratif

Il est d'abord important de préciser que le clustering collaboratif [103, 105, 39, 51, 42] se distingue des méthodes d'ensemble [25, 27, 7] sur les points suivants : le clustering collaboratif contrairement à l'apprentissage par ensemble ne cherche pas de consensus. L'objectif est que chaque algorithme puisse améliorer ses résultats propres en communiquant avec les autres algorithmes. On distingue néanmoins également des similitudes architecturales dans les frameworks des deux domaines. Tout comme les méthodes d'ensemble, le clustering collaboratif suit généralement un procédé en deux étapes :

1. Une étape locale pendant laquelle chaque algorithme optimise seul son propre modèle.

2. Une étape collaborative pendant laquelle les algorithmes échangent leurs informations.

L'étape collaborative peut parfois être une étape préalable pour rapprocher les solutions en vue d'une fusion ou d'une recherche de consensus, mais cette troisième étape n'est pas étudiée dans cette thèse. Outre l'argument que cette thèse n'a pas vocation à couvrir tous les aspects d'apprentissage par ensemble, nous avanceront aussi l'argument que les différences structurelles et potentielles redondances entre les différentes solutions font que la recherche d'un consensus n'est pas toujours quelque chose de désirable [157].

En résumé, le clustering collaboratif est un problème émergeant ayant des similitudes avec les méthodes d'ensemble [105] et que l'on peut définir comme suit [50] : si on considère un nombre fini de sites contentant des données, le clustering collaboratif propose des méthodes permettant de développer, concilier et renforcer les structures trouvées sur les différents sites par le biais d'un compromis entre les informations locales et les informations collectives, le tout via des échanges définis pour un niveau donné de granularité.

On distingue deux sous-types de collaboration [104, 55] :

- La collaboration horizontale (Figure A.5) : Plusieurs algorithmes travaillant sur les mêmes données avec des représentations potentiellement différentes.
- La collaboration verticale (Figure A.6) : Plusieurs algorithmes travaillant sur des données différentes mais ayant des distributions similaires.

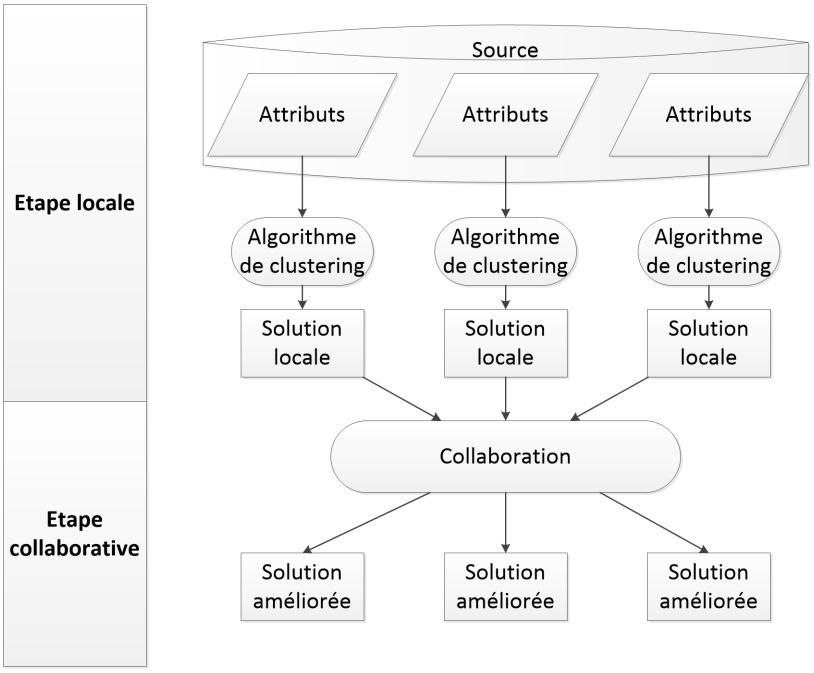


Figure A.5: Clustering collaboratif horizontal : Schéma général

Le premier algorithme de clustering collaboratif proposé par Pedrycz [101] était une version modifiée des K-moyennes floues. Cet algorithme a été proposée en deux versions, une pour la collaboration horizontale et une pour la collaboration verticale. Dans les deux cas, l'algorithme original a été modifié avec l'ajout d'un terme collaboratif dans la fonction objectif. Ce terme collaboratif issu d'une dérivation par rapport aux paramètres vise à maximiser le consensus entre les différents algorithmes.

La même idée a été plus tard reprise par Grozavu et al. afin d'être adaptée au cas des cartes de Kohonen [54, 57, 55], puis pour les cartes topographiques (GTM) par Ghassany [50, 51]. Les travaux de Pedrycz sur les K-Moyennes furent également

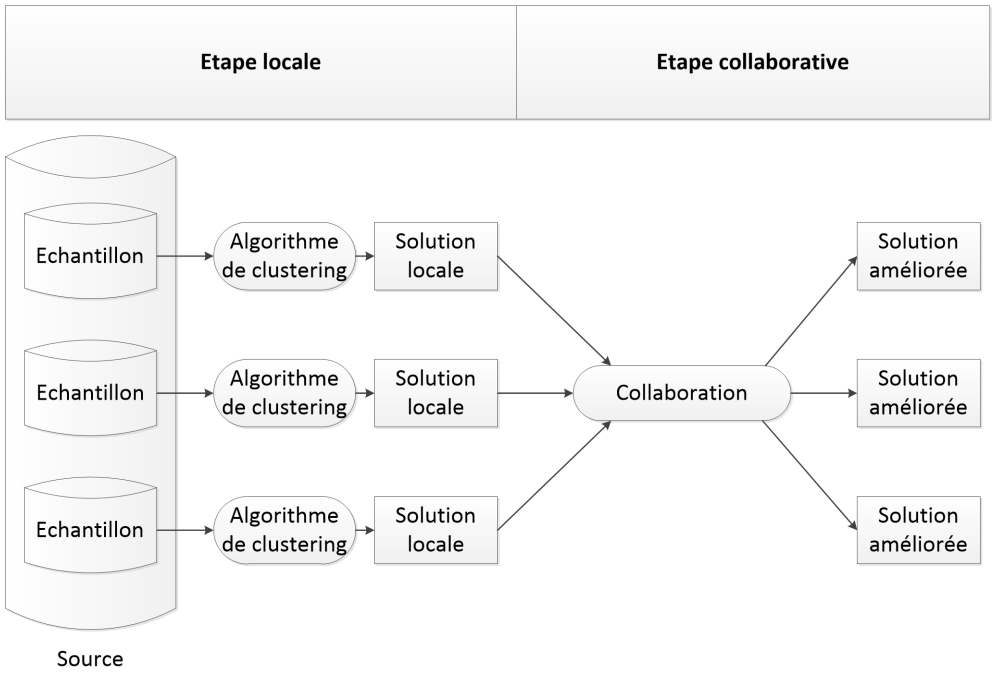


Figure A.6: Clustering collaboratif vertical : Schéma général

étendus avec l'algorithme CoEM [16] et l'algorithme CoFKM [30]. Tous ces algorithmes peuvent traiter les cas de collaborations horizontales et verticales mais souffrent tous des mêmes limites : 1) Le nombre de clusters (ou de prototypes pour SOM et GTM) doit être identique pour tous les collaborateurs. 2) Ces méthodes ne permettent pas à des algorithmes différents de collaborer. Ces deux points sont relativement limitants.

La méthode SAMARAH proposée par Wemmert [147] appartient à une autre famille d'algorithmes permettant à des algorithmes très différents de collaborer ensemble. Bien qu'étant limitée à la collaboration horizontale, cette méthode se base sur le principe solide de mapper les correspondances entre les clusters des différentes solutions à partir de matrices de confusion. L'étape collaborative de la méthode SAMARAH consiste à réduire les conflits entre les différentes partitions en modifiant des clusters, en en rajoutant ou en en supprimant. Les deux points faibles de cette méthode résident 1) dans la résolution asynchrone de ses conflits, les conflits détectés comme importants étant résolus les premiers. Cet ordre de résolution a une forte influence sur le résultat final. 2) Cette méthode bien que permettant à des algorithmes très différents de collaborer ensemble ne tient plus compte des spécificités locales de chaque algorithme une fois l'étape locale terminée.

A.2.3.3 Clustering collaboratif horizontal pour méthodes hétérogènes

Durant cette thèse, nous avons développé un framework [128, 127] ayant à la fois les avantages des méthodes basés sur les prototypes proposées depuis Pedrycz, et ceux de la méthode SAMARAH, sans en avoir les inconvénients. La méthode que nous proposons permet à des algorithmes différents de collaborer ensemble, indépendemment du type d'algorithme ou du nombre de clusters. De plus, notre méthode tient compte des caractéristiques locales des algorithmes pendant l'étape collaborative. Un comparatif des caractéristiques de notre méthodes comparées avec d'autres méthodes de la littérature est disponbible dans le Tableau A.1.

Dans notre méthode, chaque algorithme optimise localement une fonction objectif contenant à la fois un terme local basé sur les paramètres et caractéristiques de la méthode locale, et contenant également un terme collaboratif attestant du niveau de consensus entre les clusters locaux et ceux des autres algorithmes. Ce terme collaboratif utilise le mappage des clusters d'un algorithme à un autre par des matrices de confusions similaires à celles de la méthode SAMARAH. La méthode d'optimisation de ces fonctions objectif est comparable à plusieurs algorithmes de type EM fonctionnant en parallèle et s'échangeant des informations entre les itérations.

Les propriétés de notre algorithme ont été étudiées en détails. En particulier, en plus d'une étude sur la complexité, nous nous sommes intéressés au critère d'arrêt de notre système et avons choisi un critère mesurant l'évolution de l'entropie globale du système [143, 145]. Nous avons également analysé les propriétés de convergence de notre algorithme en utilisant une analyse similaire à celles des algorithmes EM et CoEM [38, 151, 16]. Nous avons ainsi démontré que notre méthode converge en moyenne vers un point stationnaire.

La méthode a été testée sur plusieurs jeux de données issus de l'UCI [47], et sur les données images de Strasbourg. Les résultats ont été validés avec plusieurs critères internes et externes et ont montré la solidité de notre approche ainsi que son efficacité.

	Notre méthode [128]	SAMARAH [147]	Méthodes basées prototypes [101, 50, 55]
Type de collaboration	Horizontale	Horizontale (+consensus)	Verticale ou Horizontale
Principe collaboratif	Matrices de confusion probabilistes	Matrices de confusion probabilistes	Dérivation
Résolution des conflits	Simultanée	ordonnée et asynchrone	Simultanée
Algorithmes hétérogènes	Oui	Oui	Non
Numbre de clusters	Pas de restrictions, peut diminuer	Pas de restrictions, peut augmenter ou diminuer	Doit être identique et rester identique

Table A.1: Comparaison de notre méthode avec les autres méthodes existantes

Enfin, nous avons proposé deux méthodes permettant d'optimiser les liens de collaboration entre les différents algorithmes : La première méthode propose une optimisation des liens de collaboration sous conditions de Karush-Kuhn-Tucker [87] afin de maximiser la vraisemblance globale du système. La seconde méthode se base sur des régressions linéaires effectuées à partir de simulations expérimentales et propose un modèle plus empirique. L'étude KKT a en particulier permis de démontrer l'importance d'avoir des solutions stables et pas trop diverses pour maximiser la fonction de vraisemblance globale de notre système.

A.2.4 Adaptation de notre framework pour la collaboration verticale

Le framework collaboratif proposé précédemment ne couvrant que le cas de la collaboration horizontale, nous avons cherché à l'adapter pour le cas de la collaboration verticale. Cependant, la grande flexibilité de notre méthode pour la collaboration horizontale constitue également une faiblesse pour l'adaptation au cas vertical. La meilleure solution que nous avons trouvée a été de tirer avantage de la structure d'algorithmes de clustering de types réseaux de neurones non-supervisés.

A.2.4.1 Réseaux neurones non-supervisés

Les réseaux de neurones non-supervisés sont des méthodes inspirées des connexions neuronales dans le cerveau. Dans le cas du clustering, les deux algorithmes de réseaux de neurones les plus connus sont les cartes auto-organisatrices (SOM) [84] et leur version probabiliste les GTM [20].

Ces deux méthodes cherchent à représenter la structure des données par une série de prototypes interconnectés, généralement disposés sur une grille en deux dimensions (d'où le nom de carte). Durant le processus d'apprentissage, les prototypes de la carte vont se spécialiser pour reconnaitre certains patterns et vont se déplacer dans leur espace en 2 dimensions afin de représenter au mieux l'espace des données. Ce sont ces propriétés qui font que les cartes SOM et GTM sont très souvent utilisées pour visualiser en deux dimensions des données multi-dimensionnelles.

Les clusters finaux sont obtenus en utilisant un algorithme de clustering classique de type K-Moyennes ou EM sur la carte obtenue à l'issue de l'algorithme SOM ou GTM. Ces méthodes proposent donc une architecture à deux étages, le premier étage étant composé de la carte des prototypes utilisée pour la visualisation, et le second étage des clusters finaux.

A.2.4.2 Adaptation à notre framework collaboratif

Ce sont ces propriétés qui nous ont intéressées. En effet, si on applique l'algorithme SOM ou GTM avec la même initialisation sur deux jeux de données différents ayant les mêmes distributions, les cartes topographiques obtenues sont en théorie très proches. Les prototypes sont donc comparables d'une carte à une autre. Ainsi, les prototypes des cartes obtenus à partir de plusieurs jeux de données ayant des distributions semblables peuvent être considérés comme plusieurs vues d'un même jeu de données. Le problème de collaboration verticale peut alors être traité comme une collaboration horizontale en appliquant notre framework collaboratif sur les algorithmes servant à générer les clusters finaux à partir des prototypes des cartes, voir Figure A.7.

C'est précisément cette méthode que nous avons proposée [129, 130] en remplaçant l'algorithme EM classique servant à obtenir le clustering final à partir de cartes GTM par une version collaborative de l'EM utilisant notre framework horizontal.

Les résultats de la méthode proposée ont été validés sur des données UCI. Les résultats obtenus étaient meilleurs que ceux des méthodes précédemment développées pour ce type d'application par Grozavu et Ghassany.

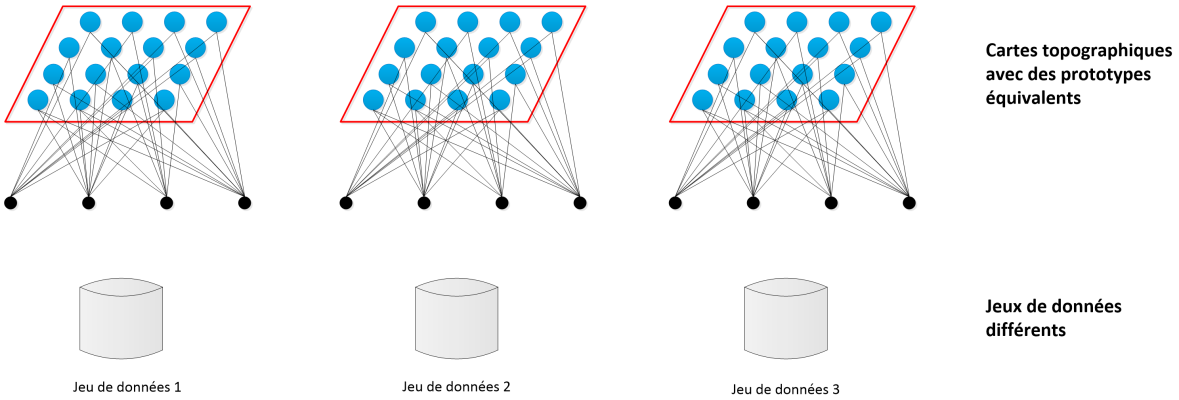


Figure A.7: De données différentes vers des prototypes similaires

A.3 Résumé des contributions scientifiques et perspectives

A.3.1 Résumé des contributions scientifiques

A.3.1.1 Contributions appliquées image

La contribution principale en terme de clustering appliqué à l'imagerie consiste en la proposition d'un nouvel algorithme de clustering pour les données d'imagerie pré-segmentées. Cet algorithme, l'algorithme SR-ICM, se base sur le modèle des champs aléatoires de Markov et est une évolution considérable par rapport aux autres algorithmes de la littérature du faire de sa capacité à traiter des données ayant des voisinages irréguliers, et aussi par la valeur sémantique de bas niveau qu'il apporte dans la descriptions des clusters ainsi identifiés.

A.3.1.2 Contributions appliquées au clustering collaboratif

Les apports au clustering collaboratif constituent le bloc central de cette thèse.

Notre première contribution dans ce domaine est la formalisation générale et le regroupement de plusieurs méthodes collaboratives et ensemblistes sous un seul schéma contenant trois blocs potentiellement indépendants : l'étape locale, l'étape collaborative et l'étape de consensus.

Nous avons ensuite proposé une méthode collaborative générique permettant de faire travailler ensemble des algorithmes de clustering de natures très différentes, et sans contrainte sur le nombre de clusters recherchés par chaque algrithme. La méthode que nous avons originalement proposée est applicable au cas de la collaboration horizontale, c'est à dire au cas de plusieurs algorithmes travaillant sur les mêmes données dans des espaces potentiellement différents. Nous avons étudié la complexité, le critère d'arrêt et les propriété de convergence de notre méthode et avons pu démontrer la convergence du système que nous proposons.

Nous avons également réalisé une étude à la fois théorique et pratique sur l'influence d'éléments tels que la qualité des solutions, leur diversité, ou encore la stabilité de celles-ci. Nous avons pu en tirer quelques conclusions intéressantes, en particulier sur le rôle de la diversité et l'importance de la stabilité. Cette étude qui dépasse le cadre de notre algorithme vient s'ajouter à d'autres travaux encore peu nombreux sur ce sujet.

Enfin, nous avons proposé une méthode astucieuse utilisant l'architecture des réseaux neurones non-supervisés de type "carte de kohonen" et qui permet de transformer n'importe quel problème de collaboration verticale en un problème de collaboration horizontale.

A.3.2 Perspectives

Les persepectives d'utilisation des méthodes proposées dans cette concernent des tâches variées et pourraient être développées d'avantage. On peut envisager des perspectives en deux temps :

A.3.2.1 Perspectives à court terme

Travaux d'imagerie : Les perspectives à court terme de ce travail concernent en particulier les travaux réalisés en imagerie. En effet, l'algorithme proposé gagnerait à être testé sur des jeux de données ayant des charactéristiques similaires (images à très hautes résolutions pré-traitées), mais concernant des paysages différents des paysages urbains étudiés avec les données de la ville de Strasbourg.

Travaux sur le collaboratif : Concernant le clustering collaboratif, il existe également différentes possibilités de travaux à court terme. Les fonctions de combinaisons proposées pour notre algorithme mériteraient de faire l'objet d'une étude afin de connaitre plus en détails, les propriétés, avantages, inconvénients et cas d'utilisation potentiels pour chacune d'elle. Certaines de ces fonctions posent également des problèmes algorithmiques intéressants pour optimiser la complexité de notre méthode.

La méthode proposée pour passer d'un problème de collaboration verticale à un problème de collaboration horizontale pourrait sans doûte être améliorée en utilisant des cartes de Kohonen (SOM ou GTM) collaborative dès le début de l'algorithme, plutôt que d'utilsier les modèles non-collaboratifs de ces algorithmes.

A.3.2.2 Perspectives à long terme

Parmi les perspectives intéressantes ouvertes par les travaux de cette thèse, on notera la question de la stratégie de collaboration.

Si nos travaux ont contribués à clarifier le rôle de la qualité des solutions, de leur diversité et de leur stabilité, la problématique de la stratégie de collaboration optimale vis-à-vis de telles variables reste ouverte. En effet, étant données les divergences de conclusion entre les approches empiriques et les approches théoriques basées sur les mathématiques, ces questions ne sont pas résolues.

Un angle d'approche original et nouveau serait de voir si des méthodologies de type "théorie de jeux" ou "problèmes de bandits" ne pourraient pas se révéler être des solutions intéressantes pour définir des stratégies de collaboration de façon plus objective et plus efficace.

A plus long terme, il nous semble évident que va se poser la question de la place du clustering collaboratif dans le contexte visiblement plus large de *l'apprentissage par transfert*. Dans le cadre de l'apprentissage non-supervisé, ce domaine est également émergeant et a des problématiques très similaires à celles du collaboratif : Comment transférer la connaissance ? A qui ou depuis qui transférer la connaissance ? Quelle est l'influence de la diversité ? Etc. Ainsi, une partie des résultats issus de l'apprentissage

par ensemble et de l'apprentissage collaborative est très certainement réutilisable dans ce cadre plus général.

Appendix B

External evaluation criteria

In this appendix we give a list of external criteria that can be used to evaluate clustering results whenever the real objects classes are available, or when a ground truth is given with the data. These criteria consists in comparing two partitions $\boldsymbol{C}^1$ and $\boldsymbol{C}^2$.

The first thing to know when comparing two partitions is that in most cases it is not possible to make a direct link between the clusters of the two partitions, or in our case between the clusters and the real classes. Because of this issue, external indexes are not directly based on the classes and clusters of the objects. Instead they compare pairs of objects to see if they are put together or separated in both partitions. For simplicity purposes, we will only consider the case of hard clustering where each object is linked to a single cluster.

Let $\mathcal{P}_2(X)$ be the set of all possibles pairs of objects from the data set X. Let M be the size of this set. For a data set that contains N elements, we have $M = |\mathcal{P}_2(X)| = \frac{1}{2}N \cdot (N-1)$.

Let aa be the number of pairs of objects that are in the same class in both $\boldsymbol{C}^1$ and $\boldsymbol{C}^2$, bb the number of pairs of objects that are in different classes in both $\boldsymbol{C}^1$ and $\boldsymbol{C}^2$, ab the number of pairs of objects that are in the same class in $\boldsymbol{C}^1$ but not in $\boldsymbol{C}^2$, and ba the number of pairs of objects that are in different classes in $\boldsymbol{C}^1$ but in the same class in $\boldsymbol{C}^2$. We have: $M = aa + ab + ba + bb$.

The higher aa and bb are, the most similar the two partitions are.

We now introduce several criteria based on these numbers of pairs.

Precision: The Precision assesses the probability that 2 objects will be in the same class in the partition $\boldsymbol{C}^2$ knowing that they are in the same class in the partition $\boldsymbol{C}^1$. The precision takes its values between 0 and 1.

$$Precision = \frac{aa}{aa + ab} \tag{B.1}$$

Jaccard Index: The Jaccard Index takes its values between 0 and 1 and is equal to 1 if and only if the two partitions $\boldsymbol{C}^1$ and $\boldsymbol{C}^2$ are identical.

$$Jaccard = \frac{aa}{aa + ab + ba} \tag{B.2}$$

The Rand Index (Rand): The Rand Index [108] takes its values between 0 and 1 and is equal to 1 if the two partitions $\boldsymbol{C}^1$ and $\boldsymbol{C}^2$ are identical.

$$Rand = \frac{aa + bb}{M} \tag{B.3}$$

The Adjusted Rand Index (ARI): The adjusted Rand index [69] is the corrected-for-chance version of the Rand index. The adjusted Rand Index can be formulated as follows shown in Equation (B.4) bellow:

$$ARI = \frac{aa - (aa + ab) \cdot (aa + ba)/M}{\frac{1}{2}(ab + ba + 2 \cdot aa) - (aa + ab) \cdot (aa + ba)/M} \tag{B.4}$$

Unlike the original Rand index, the Adjusted Rand Index can have negative values. However, the adjusted Rand Index still is equal to 1 if and only if the two partitions $\boldsymbol{C}^1$ and $\boldsymbol{C}^2$ are identical.

The Probabilistic Rand Index (PR): The Probabilistic Rand index [137] is another variant of the Rand index specifically designed to compare a proposed image segmentation with a manually defined ground-truth. It is computed as follows: let $\{S_1, ..., S_K\}$ be a set of manually segmented ground truth images corresponding to an image $X = \{x_1, ..., x_N\}$ containing N pixels. We denote l_i the label of point x_i and l_i^k the label of x_i in the segmentation S_k. For convenience of notation, we assume the existence of a set of "true labels", which we denote by $\hat{l}_i$ for the pixel x_i. Although there is arguably not one but many correct sets of labels, this measure only considers the distribution of pairwise relationships between pixels and not the values defined by one dataset. The goal is to compare a candidate segmentation S to this set and obtain a suitable measure of consistency $d(S, S_{1\cdots K})$. Given the manually labeled images, we can compute the empirical probability of the label relationship of a pixel pair x_i and x_j simply as:

$$\hat{P}(\hat{l}_i = \hat{l}_j) = \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{I}\left(l_i^k = l_j^k\right) \tag{B.5}$$

and

$$\hat{P}(\hat{l}_i \neq \hat{l}_j) = \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{I}\left(l_i^k \neq l_j^k\right) = 1 - \hat{P}(\hat{l}_i = \hat{l}_j) \tag{B.6}$$

Then, the probabilistic Rand index which takes its values between 0 and 1 is computed as follows:

$$PR(S, S_{1\cdots K}) = \frac{1}{\binom{N}{2}} \sum_{\substack{i,j \\ i \neq j}} \left[\boldsymbol{I}(l_i = l_j)(\hat{l}_i = \hat{l}_j) + \boldsymbol{I}(l_i \neq l_j)(\hat{l}_i \neq \hat{l}_j)\right] \tag{B.7}$$

The Fowlkes-Mallows index (FM): The Fowlkes-Mallows index [45] is another external index originally designed to compare two hierarchical clustering results. It is computed as follows:

$$FM = \sqrt{\frac{aa}{aa + ab} \cdot \frac{aa}{aa + ba}} \tag{B.8}$$

This index takes its values between 0 and 1 and is equal to 1 if the two partitions $\boldsymbol{C}^1$ and $\boldsymbol{C}^2$ are identical.

Outside of the context of evaluating clustering results, the Fowlkes-Mallows index is the geometric mean of the precision multiplied by the recall.

F1 Score: The F1 score, also known as the F-Mesure, or balanced F Score, is another external index based on the precision and the recall. The F1 score is the harmonic mean between the precision and the recall and is computed as follows:

$$F1 = 2\frac{\frac{aa}{aa+ab} \cdot \frac{aa}{aa+ba}}{\frac{aa}{aa+ab} + \frac{aa}{aa+ba}} \tag{B.9}$$

The F1 score also takes its values in $[0, 1]$, with 1 being the best value.

Cohen's Kappa coefficient ($\mathcal{K}$): The Kappa coefficient is defined as follows:

$$\mathcal{K} = 1 - \frac{1 - p_0}{1 - p_e} \tag{B.10}$$

with $p_0 = \frac{aa+bb}{M}$, and $p_e = \frac{1}{M^2}(aa + ab) \cdot (aa + ba) + (ab + bb) \cdot (ba + bb)$

The Kappa coefficient ($\mathcal{K}$) takes its values between -1 and 1, 1 meaning that the partitions $\boldsymbol{C}^1$ and $\boldsymbol{C}^2$ are identical.

Appendix C

UCI Data sets

In this appendix, we describe a few data sets taken from the UCI repository [47] and that have been used in the experimental parts of this Thesis.

Escherichia coli data set (EColi): This data set contains 336 instances describing cells measures. The original data set contains 7 numerical attributes (we removed the first attribute containing the sequence name). The goal of this data set is to predict the localization site of proteins by employing some measures about the cells. There are 4 main site locations that can be divided into 8 hierarchical classes.

Image Segmentation data set (ImgSeg): The 2,310 instances of this data set were drawn randomly from a database of 7 outdoor images. The images were hand segmented to create a classification for every pixel. Each instance is a 3x3 region represented by 19 attributes and there are 7 classes to be found.

Iris data set (Iris): This data set has 150 instances of iris flowers described by 4 integer attributes. The flowers can be classified in 3 categories: Iris Setosa, Iris Versicolour and Iris Virginica. Class structures are "well behaved" and the class instances are balanced (50/50/50).

Madelon data set (Madelon): Madelon is an artificial data set, 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 data set 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).

Spam base data set (Spam Base): The Spam Base data set contains 4601 observations described by 57 attributes and a label column: Spam or not Spam (1 or 0).

Waveform data set (Waveform): 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.

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.

Wine data set (Wine): This data set contains 178 instances of Italian wines from three different cultivars. All wines are described by 13 numerical attributes and the classes to be found are the 3 cultivars of origin. Class structures are "well behaved" in this data set, but the class instances are unbalanced (59/71/48).

Data Set	Instances	Attributes	Classes (Clusters)
EColi	336	7	4 - 8
ImgSeg	2,310	19	7
Iris	150	4	3
Madelon	4.400	500	32
Spam Base	4,601	57	2
Waveform	5,000	40	3
WDBC	569	30	2
Wine	178	13	3

Table C.1: Data sets characteristics

Appendix D

KKT calculus

In this appendix, we show the calculus for the Karush-Kuhn-Tucker optimization problem from section 5.1. Given the $\beta_{j,i} \geq 0$, we are trying to optimize the matrix $T = \{\tau_{j,i}\}_{J\times J}$ as shown in the system bellow:

$$\begin{cases} T = \text{argmax}_T \sum_{i=1}^{J} \sum_{j\neq i} \tau_{j,i} \cdot \beta_{j,i} \\ \forall i \quad \sum_{j\neq i}^{J} (\tau_{j,i})^p = 1, \qquad p \in \mathbb{N}^* \\ \forall (i,j) \quad \tau_{j,i} \geq 0 \end{cases} \tag{D.1}$$

From the previous system, by using Lagrange multipliers, we get the following KKT conditions:

$$\forall (i,j), i \neq j \begin{cases} (1) & \tau_{j,i} \geq 0 \\ (2) & \sum_{j\neq i}^{J} (\tau_{j,i})^p = 1 \\ (3) & \lambda_{j,i} \geq 0 \\ (4) & \tau_{j,i} \cdot \lambda_{j,i} = 0 \\ (5) & -\beta_{j,i} - \lambda_{j,i} + \nu_i \cdot (p \cdot (\tau_{j,i})^{p-1}) = 0 \end{cases} \tag{D.2}$$

For now, we will ignore the case $p = 1$ to which we will come back later. Let's begin by considering the case where $\lambda_{j,i} \neq 0$ in (4). Then, we would have $\tau_{j,i} = 0$ and with (5): $\beta_{j,i} = -\lambda_{j,i} \leq 0$. Since the $\beta_{j,i}$ have been defined as non-negative, this case is not possible, therefore we will only consider the case $\tau_{j,i} \neq 0$ and $\lambda_{j,i} = 0$. Then, with (5), we have:

$$\tau_{j,i} = \left(\frac{\beta_{j,i}}{p \cdot \nu_i} \right)^{\frac{1}{p-1}} \tag{D.3}$$

From Equation (D.3) and (2), we have:

$$1 = (p \cdot \nu_i)^{\frac{-p}{p-1}} \sum_{j\neq i} (\beta_{j,i})^{\frac{p}{p-1}} = (\nu_i)^{\frac{-p}{p-1}} \sum_{j\neq i} \left(\frac{\beta_{j,i}}{p} \right)^{\frac{p}{p-1}} \tag{D.4}$$

Then we can write:

$$\nu_i = \left(\frac{1}{\sum_{j\neq i} \left(\frac{\beta_{j,i}}{p} \right)^{\frac{p}{p-1}}} \right)^{-\frac{p-1}{p}} = \frac{1}{p} \left(\sum_{j\neq i} (\beta_{j,i})^{\frac{p}{p-1}} \right)^{\frac{p-1}{p}} \tag{D.5}$$

Then by injecting the expression of ν_i into Equation (D.3), $\forall (i,j), i \neq j, p > 1$ we have:

$$\tau_{j,i} = \left(\frac{\beta_{j,i}}{\left(\sum_{k \neq i} (\beta_{k,i})^{\frac{p}{p-1}} \right)^{\frac{p-1}{p}}} \right)^{\frac{1}{p-1}} \tag{D.6}$$

$$= \frac{(\beta_{j,i})^{\frac{1}{p-1}}}{(\sum_{k \neq i}^{J} (\beta_{k,i})^{\frac{p}{p-1}})^{\frac{1}{p}}} \tag{D.7}$$

And thus we have proved the result of Equation (5.6).

We will now come back to the case where $p = 1$. In this case, the KKT system is a bit different:

$$\forall (i,j), i \neq j \begin{cases} (1) & \tau_{j,i} \geq 0 \\ (2) & \sum_{j \neq i}^{J} (\tau_{j,i}) = 1 \\ (3) & \lambda_{j,i} \geq 0 \\ (4) & \tau_{j,i} \cdot \lambda_{j,i} = 0 \\ (5) & -\beta_{j,i} - \lambda_{j,i} + \nu_i = 0 \end{cases} \tag{D.8}$$

With (3) and (5), we have:

$$\lambda_{j,i} = \nu_i - \beta_{j,i} \geq 0 \tag{D.9}$$

Let's suppose that $\exists k$ so that $\tau_{k,i} > 0$. Then with (4), we have: $\lambda_{k,i} = 0$. And with Equation (D.9), we have: $\nu_i = \beta_{k,i}$. Then, using this information we can say that:

$$\forall j \neq k \begin{cases} \tau_{j,i} \neq 0 \implies \beta_{j,i} = \beta_{k,i} \implies \lambda_{j,i} = 0 \\ \tau_{j,i} = 0 \implies \lambda_{j,i} = \beta_{k,i} - \beta_{j,i} \geq 0 \end{cases} \tag{D.10}$$

From the second line of Equation (D.10), we can conclude the following:

$$\tau_{j,i} \neq 0 \implies \beta_{j,i} = \max_k \beta_{k,i} \tag{D.11}$$

Then, if we use (4) and (5), we have:

$$\tau_{j,i}(\nu_i - \beta_{j,i}) = 0 \tag{D.12}$$

If we sum Equation (D.12) over j and use (2), we have:

$$\nu_i = \sum_{j \neq i}^{J} \tau_{j,i} \cdot \beta_{j,i} \tag{D.13}$$

For Equation (D.13) to be correct while respecting the constraints given in Equations (D.10) and (D.11), the only solution is:

$$\forall j \neq i, \tau_{j,i} = \begin{cases} \frac{1}{Card(\beta_{j,i} = max_k(\beta_{k,i}))} & \text{if } \beta_{j,i} = max_k(\beta_{k,i}) \\ 0 & \text{otherwise} \end{cases} \tag{D.14}$$

We have therefore proved the result shown in Equation (5.5).

Appendix E

Results of the Linear Regression experiment to predict the stability after collaboration

In this appendix, we show the results of an experiment using the same protocol than in Section 5.2. Given point clouds of collaboration results for several data sets, we tried to build a linear model to predict the gain in stability during the collaborative process of our proposed horizontal collaboration framework.

We consider the following variables :

- D, the diversity with the collaborator before collaboration . We use our oriented entropy $H_{i,j}$ to get the potential information gain that the collaborator can provide.

- S_{loc}, the stability of the local result before collaboration. For each result, the stability was evaluated using 50 subsets containing 20% of randomly picked data from the original data set. We use the Adjusted Rand Index as a similarity measure between the subsets.

- S_{col}, the stability of the collaborator before collaboration.

- $\Delta S = S_{col} - S_{loc}$, the difference in stability between the receiving algorithm and the collaborating algorithm.

- GS, the stability improvement during the collaboration.

Due to the high computational cost of the stability variables on large data sets, we were unable to compute enough points for the VHR Strasbourg data set (see Section 3.3.3). Even when reducing the sample sizes and number of subsets to test, the process was still to slow and not reliable enough to generate a significant number of collaboration statistics for this data set.

The linear regression factors found for the different data sets are shown in Table E.1. As one can see, the correlation coefficients are not that good and the absolute mean error is high for most data sets.

A second linear regression on the grouped data set (excluding Spam Base which seemed problematic) gave the regression factors shown in Equation (E.1) with a correlation coefficient of 73.74% and a mean error of 0.1231. The detail of the correlation

Data Set	D	ΔS	Constant c	Correlation Coeff.	Abs. Mean Error
Iris	0.4097	0.4414	-0.0371	0.7129	0.1210
Wine	0.3533	0.4284	0.0075	0.5645	0.1351
WDBC	0.4175	0.4344	0.0892	0.9137	0.0801
EColi	0.0882	0.3800	0.0767	0.5211	0.1229
ImgSeg	0.2817	0.4392	0.0340	0.7652	0.0587
Spam Base	-0.0645	0.1545	-0.204	0.2510	0.0518
Average	0.36555	0.4385	0.0234	0.7390	0.0987

Table E.1: Regression factors for the prediction of GS

coefficients and absolute mean error on each data set with these coefficients is shown in Table E.2.

$$GS = 0.4097 \times D + 0.4297 \times \Delta S - 0.011 \tag{E.1}$$

Data Set	GS Mean Error	GS Correlation
Iris	0.1234	0.7136
Wine	0.1375	0.5644
WDBC	0.1234	0.9138
EColi	0.1608	0.4627
ImgSeg	0.0676	0.7517

Table E.2: Linear Regression Absolute Mean Errors and Correlation Coefficients

Following these relatively poor results and another experiment which failed to prove that stability could be used to predict potential improvements or to help refining the prediction for quality indexes (even with other factors such as difference in quality or diversity), we conclude that the stability of a global solution had little could be used neither to predict the quality of a final solution, nor to predict its stability after collaboration.

Bibliography

[1] Akaike, H.: Information theory and an extension of the maximum likelihood principle. In: B.N. Petrov, F. Csaki (eds.) Second International Symposium on Information Theory, pp. 267–281. Akadémiai Kiado, Budapest (1973) 33

[2] Ankerst, M., Breunig, M.M., peter Kriegel, H., Sander, J.: Optics: Ordering points to identify the clustering structure. In: In ACM SIGMOD international conference on Management of data, pp. 49–60. ACM Press (1999) 19, 135

[3] Ardila, J., Tolpekin, V., Bijker, W.: Markov random field based super-resolution mapping for identification of urban trees in VHR images. In: IEEE International Geoscience & Remote Sensing Symposium, IGARSS 2010, July 25-30, 2010, Honolulu, Hawaii, USA, Proceedings, pp. 1402–1405 (2010) 42

[4] Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Mach. Learn. **47**(2-3), 235–256 (2002) 131

[5] Avriel, M.: Nonlinear programming: analysis and methods. Prentice-Hall series in automatic computation. Prentice-Hall (1976) 114

[6] Azimi, J., Fern, X.: Adaptive cluster ensemble selection. In: C. Boutilier (ed.) IJCAI, pp. 992–997 (2009) 78

[7] Bachman, P., Alsharif, O., Precup, D.: Learning with pseudo-ensembles. In: Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, K. Weinberger (eds.) Advances in Neural Information Processing Systems 27, pp. 3365–3373. Curran Associates, Inc. (2014) 59, 140

[8] BALL, N.M., BRUNNER, R.J.: Data mining and machine learning in astronomy. International Journal of Modern Physics D **19**(07), 1049–1106 (2010) 16, 134

[9] Bartkowiak, A.: Intelligent Information Processing and Web Mining: Proceedings of the International IIS: IIPWM'04 Conference held in Zakopane, Poland, May 17–20, 2004. Springer Berlin Heidelberg, Berlin, Heidelberg (2004) 119

[10] Bechtel, A., Puttmann, W., Carlson, T.N., Ripley, D.A.: On the Relation between NDVI, Fractional Vegetation Cover, and Leaf Area Index. Remote Sensing of Environment **62**(3), 241–252 (1997) 39

[11] Ben-David, S., von Luxburg, U., Pál, D.: A sober look at clustering stability. In: G. Lugosi, H. Simon (eds.) Learning Theory, Lecture Notes in Computer Science, vol. 4005, pp. 5–19. Springer Berlin Heidelberg (2006) 34, 95, 131

[12] Berkhin, P.: Survey of clustering data mining techniques. Tech. rep., Accrue Software (2002) 18

[13] Besag, J.: On the statistical analysis of dirty pictures. Journal of the Royal Statistical Society. Series B **48**(3), 259–302 (1986) 39, 43, 44, 137

[14] Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Kluwer Academic Publishers, Norwell, MA, USA (1981) 18, 24, 67, 135

[15] Bickel, S., Scheffer, T.: Multi-view clustering. In: Proceedings of the 4th IEEE International Conference on Data Mining (ICDM 2004), 1-4 November 2004, Brighton, UK, pp. 19–26. IEEE Computer Society (2004) 60, 140

[16] Bickel, S., Scheffer, T.: Estimation of mixture models using co-em. In: J. Gama, R. Camacho, P. Brazdil, A. Jorge, L. Torgo (eds.) Machine Learning: ECML 2005, 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005, Proceedings, Lecture Notes in Computer Science, vol. 3720, pp. 35–46. Springer (2005) 68, 71, 84, 142, 143

[17] Bickel, S., Scheffer, T.: Estimation of mixture models using co-em. In: Proceedings of the ICML workshop on learning with multiple views (2005) 71

[18] Bishop, C., Svensén, M., Williams, C.: GTM: A principled alternative to the Self-Organizing Map. In: Artificial Neural Networks - ICANN 96, LNCS, pp. 165–170. Springer (1996) 112, 115

[19] Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press, Inc., New York, NY, USA (1995) 116

[20] Bishop, C.M., Svensén, M., I.Williams, C.K.: GTM: The generative topographic mapping. Neural Comput **10**(1), 215–234 (1998) 8, 69, 115, 116, 117, 119, 135, 144

[21] Blaschke, T.: Object based image analysis for remote sensing. ISPRS Journal of Photogrammetry and Remote Sensing **65**(1), 2–16 (2010) 39, 137

[22] Bose, I., Mahapatra, R.K.: Business data mining - a machine learning perspective. Information & Management **39**(3), 211–225 (2001) 16, 134

[23] Boyarshinov, V.: Machine learning in computational finance. Ph.D. thesis, Rensselaer Polytechnic Institute, Troy, NY, USA (2005). AAI3173253 16, 134

[24] Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. **23**(11), 1222–1239 (2001) 43, 137

[25] Breiman, L.: Bagging predictors. Machine Learning **24**(2), 123–140 (1996) 59, 140

[26] Burnham, K.P., Anderson, D.R.: Multimodel Inference: Understanding AIC and BIC in Model Selection. Sociological Methods & Research **33**(2), 261–304 (2004) 33

[27] Chawla, N.V., Eschrich, S., Hall, L.O.: Creating ensembles of classifiers. In: N. Cercone, T.Y. Lin, X. Wu (eds.) ICDM, pp. 580–581. IEEE Computer Society (2001) 59, 140

[28] Chen, Y., Qin, B., Liu, T., Liu, Y., Li, S.: The comparison of SOM and k-means for text clustering. Computer and Information Science **3**(2), 268–274 (2010) 115

[29] Chu, S., Roddick, J.F., Pan, J.: Improved search strategies and extensions to k-medoids based clustering algorithms. Int. J. Bus. Intell. Data Min. **3**(2), 212–231 (2008) 23

[30] Cleuziou, G., Exbrayat, M., Martin, L., Sublemontier, J.: Cofkm: A centralized method for multiple-view clustering. In: W. Wang, H. Kargupta, S. Ranka, P.S. Yu, X. Wu (eds.) ICDM 2009, The Ninth IEEE International Conference on Data Mining, Miami, Florida, USA, 6-9 December 2009, pp. 752–757. IEEE Computer Society (2009) 68, 142

[31] Condorcet, M.N.D.: Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix. De l'imprimerie royale, Paris (1785) 58, 139

[32] Cordier, M., Fromont, É., Quiniou, R.: Learning rules from multisource data for cardiac monitoring. CoRR **abs/0902.3373** (2009) 16, 134

[33] Cornuéjols, A., Martin, C.: Unsupervised object ranking using not even weak experts. In: B.L. Lu, L. Zhang, J.T. Kwok (eds.) ICONIP (1), Lecture Notes in Computer Science, vol. 7062, pp. 608–616. Springer (2011) 63

[34] Cuesta-Albertos, J.A., Fraiman, R.: Impartial trimmed k-means for functional data. Comput. Stat. Data Anal. **51**(10), 4864–4877 (2007) 23

[35] Das, S., Mozer, M.: A Unified Gradient-Descent/Clustering Architecture for Finite State Machine Induction. In: J.D. Cowan, G. Tesauro, J. Alspector (eds.) NIPS, pp. 19–26. Morgan Kaufmann (1993) 27

[36] Davies, D.L., Bouldin, D.W.: A cluster separation measure. IEEE Transactions on Pattern Analysis and Machine Intelligence **1**(2), 224–227 (1979) 23, 31, 84, 96

[37] Day, W.H., Edelsbrunner, H.: Efficient algorithms for agglomerative hierarchical clustering methods. Journal of Classification **1**(1), 7–24 (1984) 21, 135

[38] Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B **39**(1), 1–38 (1977) 27, 45, 72, 77, 87, 117, 135, 143

[39] Depaire, B., Falcon, R., Vanhoof, K., Wets, G.: PSO Driven Collaborative Clustering: a Clustering Algorithm for Ubiquitous Environments. Intelligent Data Analysis **15**, 49–68 (2011) 59, 140

[40] Dunn, J.C.: A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. Journal of Cybernetics **3**(3), 32–57 (1973) 30

[41] Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: In International Conference on Knowledge Discovery and Information Retrieval, pp. 226–231 (1996) 19, 23, 135

[42] Falcon, R., Jeon, G., Bello, R., Jeong, J.: Learning collaboration links in a collaborative fuzzy clustering environment. In: Proceedings of the artificial intelligence 6th Mexican international conference on Advances in artificial intelligence, MICAI'07, pp. 483–495. Springer-Verlag, Berlin, Heidelberg (2007) 59, 140

[43] Figueiredo, M.A.T., Jain, A.K.: Unsupervised learning of finite mixture models. IEEE transactions on pattern analysis and machine intelligence **24**, 381–396 (2000) 34

[44] Forestier, G., Wemmert, C., Gancarski, P.: Collaborative multi-strategical classification for object-oriented image analysis. In: Workshop on Supervised and Unsupervised Ensemble Methods and Their Applications in conjunction with IbPRIA, pp. 80–90 (2007) 69, 130

[45] Fowlkes, E.B., Mallows, C.L.: A Method for Comparing Two Hierarchical Clusterings. Journal of the American Statistical Association **78**(383), 553–569 (1983) 150

[46] Frahling, G., Sohler, C.: A fast k-means implementation using coresets. In: SCG '06: Proceedings of the twenty-second annual symposium on Computational geometry, pp. 135–143. ACM, New York, NY, USA (2006) 23

[47] Frank, A., Asuncion, A.: UCI machine learning repository (2010) 84, 143, 153

[48] Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science **315** (2007) 25

[49] Friedman, J.H., Meulman, J.J.: Clustering objects on subsets of attributes (with discussion). Journal Of The Royal Statistical Society Series B **66**(4), 815–849 (2004) 62, 140

[50] Ghassany, M.: Contributions to collaborative clustering. Ph.D. thesis, The Paris 13 University (2012) 65, 66, 69, 94, 119, 130, 131, 141, 143

[51] Ghassany, M., Grozavu, N., Bennani, Y.: Collaborative clustering using prototype-based techniques. International Journal of Computational Intelligence and Applications **11**(3) (2012) 59, 69, 122, 140, 141

[52] Gonzalez, R.C., Woods, R.E.: Digital Image Processing (3rd Edition). Prentice-Hall, Inc., Upper Saddle River, NJ, USA (2006) 40, 137

[53] Green, P.J.: On Use of the EM Algorithm for Penalized Likelihood Estimation. Journal of the Royal Statistical Society. Series B (Methodological) **52**(3), 443–452 (1990) 69

[54] Grozavu, N.: Collaborative unsupervised learning and cluster characterization. Ph.D. thesis, The Paris 13 University (2009) 68, 141

[55] Grozavu, N., Bennani, Y.: Topological collaborative clustering. Australian Journal of Intelligent Information Processing Systems **12**(3) (2010) 63, 66, 68, 94, 130, 141, 143

[56] Grozavu, N., Cabanes, G., Bennani, Y.: Diversity analysis in collaborative clustering. In: 2014 International Joint Conference on Neural Networks, IJCNN 2014, Beijing, China, July 6-11, 2014, pp. 1754–1761 (2014) 78, 100, 108

[57] Grozavu N., B.Y.: Topological collaborative clustering. in LNCS Springer of ICONIP'10 : 17th International Conference on Neural Information Processing (2010) 68, 69, 141

[58] Guha, S., Rastogi, R., Shim, K.: Cure: An efficient clustering algorithm for large databases. SIGMOD Rec. **27**(2), 73–84 (1998) 22, 135

[59] Halkidi, M., Batistakis, Y., Vazirgiannis, M.: Clustering validity checking methods: Part ii (2001) 29

[60] Hamdan, H., Hajjar, C.: Kohonen neural networks for interval-valued data clustering. International Journal of Advanced Computer Science **2**(11), 412–419 (2012) 8, 114

[61] Hamdi, F., Bennani, Y.: Learning random subspace novelty detection filters. In: The 2011 International Joint Conference on Neural Networks, IJCNN 2011, San Jose, California, USA, July 31 - August 5, 2011, pp. 2273–2280 (2011) 17

[62] Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques, 3rd edn. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2011) 19

[63] Hansen, M.H., Yu, B.: Model selection and the principle of minimum description length. Journal of the American Statistical Association **96**, 746–774 (1998) 34

[64] Hartigan, J., Wong, M.: Algorithm AS 136: A K-means clustering algorithm. Applied Statistics pp. 100–108 (1979) 23

[65] Hernández-Gracidas, C.A., Sucar, L.E.: Markov random fields and spatial information to improve automatic image annotation. In: D. Mery, L. Rueda (eds.) PSIVT, Lecture Notes in Computer Science, vol. 4872, pp. 879–892. Springer (2007) 42

[66] Hodge, V., Austin, J.: A survey of outlier detection methodologies. Artif. Intell. Rev. **22**(2), 85–126 (2004) 17

[67] Hu, T., Yu, Y., Xiong, J., Sung, S.Y.: Maximum likelihood combination of multiple clusterings. Pattern Recogn. Lett. **27**(13), 1457–1464 (2006) 68

[68] Huang, D., Lai, J., Wang, C.: Combining multiple clusterings via crowd agreement estimation and multi-granularity link analysis. Neurocomputing **170**, 240–250 (2015) 63

[69] Hubert, L., Arabie, P.: Comparing Partitions. Journal of the Classification **2**, 193–218 (1985) 84, 150

[70] Iam-on, N., Boongoen, T.: Comparative study of matrix refinement approaches for ensemble clustering. Machine Learning **98**(1-2), 269–300 (2015) 79

[71] Jain, A.K., Dubes, R.C.: Algorithms for clustering data. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1988) 29, 34

[72] Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: A review. ACM Computing Surveys **31**(3), 264–323 (1999) 18, 21, 22, 135

[73] Jasani, B., Pesaresi, M., Schneiderbauer, S., Zeug, G.: Remote Sensing from Space: Supporting International Peace and Security, 1st edn. Springer Publishing Company, Incorporated (2009) 38

[74] Jenssen, R., Erdogmus, D., II, K.E.H., Principe, J.C., Eltoft, T.: Information cut for clustering using a gradient descent approach. Pattern Recognition **40**(3), 796 – 806 (2007) 27

[75] Jähne, B.: Digital Image Processing 6th Edition. Springer, Berlin [u.a.] (2005) 38

[76] Jong, K., Mary, J., Cornuéjols, A., Marchiori, E., Sebag, M.: Ensemble feature ranking. In: J.F. Boulicaut, F. Esposito, F. Giannotti, D. Pedreschi (eds.) PKDD, Lecture Notes in Computer Science, vol. 3202, pp. 267–278. Springer (2004) 63

[77] Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., Wu, A.Y.: An efficient k-means clustering algorithm: Analysis and implementation. IEEE Trans. Pattern Anal. Mach. Intell. **24**(7), 881–892 (2002) 23

[78] Kato, Z., Berthod, M., Zerubia, J.: A hierarchical Markov random field model and multi-temperature annealing for parallel image classification. Research Report RR-1938, INRIA (1993) 45, 137

[79] Kindermann, R., Snell, J.L.: Markov Random Fields and Their Applications. AMS (1980) 42, 137

[80] Kittler, J., Föglein, J.: Contextual classification of multispectral pixel data. Image Vision Comput. **2**(1), 13–29 (1984) 39

[81] Kittler, J., Hatef, M., Duin, R.P.W., Matas, J.: On combining classifiers. IEEE Trans. Pattern Anal. Mach. Intell. **20**(3), 226–239 (1998) 59, 139

[82] Kohonen, T.: Self-Organization and Associative Memory. Springer-Verlag Berlin (1984) 112

[83] Kohonen, T.: Self-organizing Maps. Springer-Verlag Berlin, Berlin (1995) 115

[84] Kohonen, T.: Self-organizing Maps. Springer Berlin (2001) 69, 112, 119, 135, 144

[85] Kononenko, I.: Machine learning for medical diagnosis: History, state of the art and perspective. Artif. Intell. Med. **23**(1), 89–109 (2001) 16, 134

[86] Krieger, M.A., Green, E.P.: A generalized rand-index method for consensus clustering of separate partitions of the same data base. Journal of Classification **16**(1), 63–89 (1999) 61

[87] Kuhn, H.W., Tucker, A.W.: Nonlinear programming. In: B.U. of California Press (ed.) Proceedings of 2nd Berkeley Symposium, pp. 481–492 (1951) 93, 143

[88] Kuleshov, V., Precup, D.: Algorithms for multi-armed bandit problems. CoRR **abs/1402.6028** (2014) 131

[89] Kuncheva, L.I.: Classifier ensembles: Facts, fiction, faults and future (2008). (slides, plenary talk) 58

[90] Leordeanu, M., Hebert, M., Sukthankar, R.: An integer projected fixed point method for graph matching and map inference. In: Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, A. Culotta (eds.) NIPS, pp. 1114–1122. Curran Associates, Inc. (2009) 43

[91] Li, C.X., Yu, J.: A Novel Fuzzy C-Means Clustering Algorithm. In: RSKT, Lecture Notes in Computer Science, vol. 4062, pp. 510–515. Springer (2006) 24

[92] Liu, Z.Y., Qiao, H., Su, J.H.: Neural Information Processing: 21st International Conference, ICONIP 2014, Kuching, Malaysia, November 3-6, 2014. Proceedings, Part II, chap. MAP Inference with MRF by Graduated Non-Convexity and Concavity Procedure, pp. 404–412. Springer International Publishing, Cham (2014) 43

[93] Lourenço, A., Rota Bulò, S., Rebagliati, N., Fred, A., Figueiredo, M., Pelillo, M.: Probabilistic consensus clustering using evidence accumulation. Machine Learning **98**(1-2), 331–357 (2015) 63

[94] von Luxburg, U.: Clustering stability: An overview. Foundations and Trends in Machine Learning **2**(3), 235–274 (2010) 34, 95, 108

[95] Ma, D., Zhang, A.: An Adaptive Density-Based Clustering Algorithm for Spatial Database with Noise. Data Mining, IEEE International Conference on **0**, 467–470 (2004) 19

[96] MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967) 22, 23, 135

[97] Meila, M., Shi, J.: Learning segmentation by random walks. In: In Advances in Neural Information Processing Systems, pp. 873–879. MIT Press (2001) 26, 135

[98] Pakhira, M.K., Bandyopadhyay, S., Maulik, U.: Validity index for crisp and fuzzy clusters. Pattern Recognition **37**(3), 487–501 (2004) 29, 136

[99] Pal, N.R., Pal, S.K.: A review on image segmentation techniques. Pattern Recognition **26**(9), 1277–1294 (1993) 41

[100] Pavlidis, T.: Structural Pattern Recognition. Springer (1977) 40, 137

[101] Pedrycz, W.: Collaborative fuzzy clustering. Pattern Recognition Letters **23**(14), 1675–1686 (2002) 60, 63, 64, 67, 94, 130, 139, 141, 143

[102] Pedrycz, W.: Fuzzy clustering with a knowledge-based guidance. Pattern Recogn. Lett. **25**(4), 469–480 (2004) 63, 67

[103] Pedrycz, W.: Interpretation of clusters in the framework of shadowed sets. Pattern Recogn. Lett. **26**(15), 2439–2449 (2005) 59, 140

[104] Pedrycz, W.: Knowledge-Based Clustering. John Wiley & Sons, Inc. (2005) 66, 141

[105] Pedrycz, W., Hirota, K.: A consensus-driven fuzzy clustering. Pattern Recognition Letters **29**(9), 1333–1343 (2008) 59, 65, 140, 141

[106] Pelleg, D., Moore, A.: X-means: Extending K-means with efficient estimation of the number of clusters. In: In Proceedings of the 17th International Conf. on Machine Learning, pp. 727–734 (2000) 23

[107] Puissant, A.: Information géographique et image à très haute résolution. Ph.D. thesis, University Strasbourg 1 (2006) 41

[108] Rand, W.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association. pp. 846–850 (1971) 79, 150

[109] Richards, J.A., Jia, X.: Remote Sensing Digital Image Analysis: An Introduction, 3rd edn. Springer-Verlag New York, Inc., Secaucus, NJ, USA (1999) 38

[110] Roth, S., Black, M.J.: Fields of experts. In: A. Blake, P. Kohli, C. Rother (eds.) Markov Random Fields for Vision and Image Processing, pp. 297–310. MIT Press (2011) 42

[111] Rougier, S., Puissant, A.: Improvements of urban vegetation segmentation and classification using multi-temporal pleiades images. 5th International Conference on Geographic Object-Based Image Analysis p. 6 (2014) 48, 84, 138

[112] Rousseeuw, P.J., Leroy, A.M.: Robust Regression and Outlier Detection. John Wiley & Sons, Inc., New York, NY, USA (1987) 17, 31

[113] Rousseeuw, R.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics. **20**, 53–65 (1987) 23, 84, 96

[114] Ryali, S., Chen, T., Supekar, K., Menon, V.: A parcellation scheme based on von mises-fisher distributions and markov random fields for segmenting brain regions using resting-state fmri. NeuroImage **65**, 83–96 (2013) 42

[115] Sander, J., Ester, M., Kriegel, H.P., Xu, X.: Density-based clustering in spatial databases: The algorithm gdbscan and its applications. Data Min. Knowl. Discov. **2**(2), 169–194 (1998) 19

[116] Sander, J., Ester, M., Kriegel, H.P., Xu, X.: Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and Its Applications. Data Min. Knowl. Discov. **2**(2), 169–194 (1998) 19

[117] Schapire, R.E.: The strength of weak learnability. Mach. Learn. **5**(2), 197–227 (1990) 59, 139

[118] Schwarz, G.: Estimating the Dimension of a Model. The Annals of Statistics **2**(6), 461–464 (1978) 33

[119] Shekhar, S., Member, S., Schrater, P.R., Vatsavai, R.R., Wu, W., Chawla, S.: Spatial contextual classification and prediction models for mining geospatial data. IEEE Transactions on Multimedia **4**, 174–188 (2002) 39

[120] Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 888–905 (2000) 26, 135

[121] Silva, A.D., Lechevallier, Y., de A. T. de Carvalho, F., Trousse, B.: Mining web usage data for discovering navigation clusters. In: P. Bellavista, C.M. Chen, A. Corradi, M. Daneshmand (eds.) ISCC, pp. 910–915. IEEE Computer Society (2006) 63

[122] Steinhaus, H.: Sur la division des corps matériels en parties. Bull. Acad. Polon. Sci. Cl. III. 4 pp. 801–804 (1956) 22

[123] Strehl, A., Ghosh, J., Cardie, C.: Cluster ensembles - a knowledge reuse framework for combining multiple partitions. Journal of Machine Learning Research **3**, 583–617 (2002) 63, 79

[124] Sublemontier, J.: Unsupervised collaborative boosting of clustering: An unifying framework for multi-view clustering, multiple consensus clusterings and alternative clustering. In: The 2013 International Joint Conference on Neural Networks, IJCNN 2013, Dallas, TX, USA, August 4-9, 2013, pp. 1–8. IEEE (2013) 63

[125] Sublime, J., Bennani, Y., Cornuéjols, A.: A compactness-based icm algorithm for vhr satellite image segmentation. In: Extraction et Gestion des Connaissances EGC2015, Luxembourg (2015) 46, 137

[126] Sublime, J., Cornuéjols, A., Bennani, Y.: A new energy model for the hidden markov random fields. In: ICONIP 2014, Part II, Lecture Notes in Computer Science, vol. 8835, pp. 60–67 (2014) 46, 137

[127] Sublime, J., Cornuéjols, A., Bennani, Y.: Collaborative-based multi-scale clustering in very high resolution satellite images. In: The 23rd International Conference on Neural Information Processing (ICONIP 2016), Lecture Notes in Computer Science (2016) 71, 86, 142

[128] Sublime, J., Grozavu, N., Bennani, Y., Cornuéjols, A.: Collaborative clustering with heterogeneous algorithms. In: 2015 International Joint Conference on Neural Networks, IJCNN 2015, Killarney, Ireland, July 12-18, 2015 (2015) 71, 130, 142, 143

[129] Sublime, J., Grozavu, N., Bennani, Y., Cornuéjols, A.: Vertical collaborative clustering using generative topographic maps. In: 7th International Conference on Soft Computing and Pattern Recognition, Fukuoka, Japan (2015) 112, 130, 144

[130] Sublime, J., Grozavu, N., Cabanes, G., Bennani, Y., Cornuéjols, A.: From horizontal to vertical collaborative clustering using generative topographic maps. International Journal of Hybrid Intelligent Systems **12**(4) (2016) 112, 130, 144

[131] Sublime, J., Troya-Galvis, A., Bennani, Y., Gancarski, P., Cornuéjols, A.: Semantic rich icm algorithm for vhr satellite image segmentation. In: IAPR International Conference on Machine Vision Applications, Tokyo (2015) 46, 87, 138

[132] Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining, (First Edition). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (2005) 29

[133] Tanabe, K., Lučić, B., Amić, D., Kurita, T., Kaihara, M., Onodera, N., Suzuki, T.: Prediction of carcinogenicity for diverse chemicals based on substructure grouping and svm modeling. Molecular Diversity **14**(4), 789–802 (2010) 16, 134

[134] Tang, Y., Wu, X., Bu, W.: Saliency detection based on graph-structural agglomerative clustering. In: Proceedings of the 23rd ACM International Conference on Multimedia, MM '15, pp. 1083–1086. ACM, New York, NY, USA (2015) 21, 135

[135] Tarca, A.L., Carey, V.J., Chen, X.W., Romero, R., Drăghici, S.: Machine learning and its applications to biology. PLoS Comput Biol **3**(6), e116+ (2007) 16, 134

[136] Tsai, C.Y., Chiu, C.C.: Developing a feature weight self-adjustment mechanism for a K-means clustering algorithm. Comput. Stat. Data Anal. **52**(10), 4658–4672 (2008) 23

[137] Unnikrishnan, R., Hebert, M.: Measures of similarity. In: WACV/MOTION, p. 394. IEEE Computer Society (2005) 52, 150

[138] Unnikrishnan, R., Pantofaru, C., Hebert, M.: A measure for objective evaluation of image segmentation algorithms. In: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops - Volume 03, CVPR '05, pp. 34–. IEEE Computer Society, Washington, DC, USA (2005) 52

[139] Vatanen, T., Osmala, M., Raiko, T., Lagus, K., Sysi-Aho, M., OreŽic, M., Honkela, T., Lähdesmäki, H.: Self-organization and missing values in som and gtm. Neurocomputing **147**(Complete), 60–70 (2015) 119

[140] Vesanto, J., Alhoniemi, E.: Clustering of the Self-Organizing Map. Neural Networks, IEEE Transactions on **11**(3), 586–600 (2000) 119

[141] Viswanath, P., Pinkesh, R.: l-DBSCAN: A Fast Hybrid Density Based Clustering Method. In: ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition, pp. 912–915. IEEE Computer Society, Washington, DC, USA (2006) 19

[142] Vrieze, S.I.: Model selection and psychological theory: a discussion of the differences between the Akaike information criterion (AIC) and the Bayesian information criterion (BIC). Psychological methods **17**(2), 228–243 (2012) 33

[143] Wang, X.N., Wei, J.M., Jin, H., Yu, G., Zhang, H.W.: Probabilistic confusion entropy for evaluating classifiers. Entropy **15**(11), 4969–4992 (2013) 79, 143

[144] Ward, J.H.: Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association **58**(301), 236–244 (1963) 21, 22, 135

[145] Wei, J.M., Yuan, X.J., Hu, Q.H., Wang, S.Q.: A novel measure for evaluating classifiers. Expert System Applications **37**(5), 3799–3809 (2010) 79, 143

[146] Weinman, J.: A brief introduction to ICM (2008) 44

[147] Wemmert, C.: Classification hybride distribuée par collaboration de méthodes non supervisées. Ph.D. thesis, The University of Strasbourg (2000) 32, 69, 70, 130, 142, 143

[148] Wemmert, C., Gancarski, P.: A multi-view voting method to combine unsupervised classifications. Artificial Intelligence and Applications, Malaga, Spain, pp. 447 – 452 (2002) 70

[149] Weng, Q.: Remote sensing of impervious surfaces in the urban areas: Requirements, methods, and trends. Remote Sensing of Environment **117**(0), 34–49 (2012) 39

[150] Wolpert, D.H.: Stacked generalization. Neural Networks **5**, 241–259 (1992) 59, 139

[151] Wu, C.F.J.: On the Convergence Properties of the EM Algorithm. The annals of statistics **11**(1), 95–103 (1983) 28, 143

[152] Xu, R., Wunsch II, D.: Survey of clustering algorithms. Trans. Neur. Netw. **16**(3), 645–678 (2005) 18, 34

[153] Yu, Q., Sorjamaa, A., Miche, Y., Severin, E.: A methodology for time series prediction in finance. In: A. Lendasse (ed.) ESTSP, European Symposium on Time Series Prediction, pp. 285–293. Multiprint Oy / Otamedia , Espoo, Finland, Porvoo, Finland (2008) 16, 134

[154] Zarinbal, M., Zarandi, M.F., Turksen, I.: Relative entropy collaborative fuzzy clustering method. Pattern Recognition **48**(3), 933 – 940 (2015) 79

[155] Zhang, D., Wang, Y., Zhou, L., Yuan, H., Shen, D.: Multimodal classification of alzheimer's disease and mild cognitive impairment. NeuroImage **55**(3), 856–867 (2011) 16, 134

[156] Zhang, L., Ji, Q.: Image segmentation with a unified graphical model. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(8), 1406–1425 (2010) 43

[157] Zhang, S., Zhang, C., Wu, X.: Knowledge Discovery in Multiple Databases. Advanced Information and Knowledge Processing. Springer (2004) 65, 141

[158] Zhang, W., Zhang, L., Hu, Y., Jin, R., Cai, D., He, X.: Sparse learning for stochastic composite optimization. In: Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI), pp. 893–899 (2014) 16, 134

[159] Zhang, W., Zhao, D., Wang, X.: Agglomerative clustering via maximum incremental path integral. Pattern Recognition **46**(11), 3056–3065 (2013) 21, 135

[160] Zhang, Y., Brady, M., Smith, S.M.: Segmentation of brain mr images through a hidden markov random field model and the expectation maximization algorithm. IEEE Trans. Med. Imaging **20**(1), 45–57 (2001) 45, 137

[161] Zimek, A., Vreeken, J.: The blind men and the elephant: on meeting the problem of multiple truths in data from clustering and pattern mining perspectives. Machine Learning **98**(1-2), 121–155 (2015) 59, 61, 140

[162] Zucker, S.W.: Region growing: Childhood and Adolescence. Computer Graphics and Image Processing **5**(3), 382–399 (1976) 16, 40, 134

université PARIS-SACLAY
ÉCOLE DOCTORALE
Agriculture, alimentation, biologie, environnement, santé (ABIES)

Titre : Contributions au clustering collaboratif et à ses potentielles applications en imagerie à très haute résolution

Mots clés : Apprentissage non-supervisé, clustering collaboratif, segmentation d'image

Résumé : Cette thèse présente plusieurs algorithmes développés dans le cadre du projet ANR COCLICO et contient deux axes principaux :

Le premier axe concerne l'introduction d'un algorithme applicable aux images satellite à très haute résolution, qui est basé sur les champs aléatoires de Markov et qui apporte des notions sémantiques sur les clusters découverts. Cet algorithme est inspiré de l'algorithme *Iterated conditional modes* (ICM) et permet de faire un clustering sur des segments d'images pré-traitées. La méthode que nous proposons permet de gérer des voisinages irréguliers entre segments et d'obtenir des informations sémantiques de bas niveau sur les clusters de l'image traitée.

Le second axe porte sur le développement de méthodes de clustering collaboratif applicables à autant d'algorithmes que possible, ce qui inclut les algorithmes du premier axe. La caractéristique principale des méthodes proposées dans cette thèse est leur applicabilité aux deux cas suivants : 1) plusieurs algorithmes travaillant sur les mêmes objets dans des espaces de représentation différents, 2) plusieurs algorithmes travaillant sur des données différentes ayant des distributions similaires. Les méthodes que nous proposons peuvent s'appliquer à de nombreux algorithmes comme l'ICM, les K-Moyennes, l'algorithme EM, ou les cartes topographiques (SOM et GTM). Contrairement aux méthodes précédemment proposées, notre modèle permet à des algorithmes très différents de collaborer ensemble, n'impose pas de contrainte sur le nombre de clusters recherchés et a une base mathématique solide.

Title : Contributions to collaborative clustering and its potential applications on very high resolution satellite images

Keywords : Unsupervised learning, collaborative clustering, image segmentation

Abstract: This thesis presents several algorithms developed in the context of the ANR COCLICO project and contains two main axis:

The first axis is concerned with introducing Markov Random Fields (MRF) based models to provide a semantic rich and suited algorithm applicable to images that are already segmented. This method is based on the Iterated Conditional Modes Algorithm (ICM algorithm) and can be applied to the segments of very high resolution (VHR) satellite pictures. Our proposed method can cope with highly irregular neighborhood dependencies and provides some low level semantic information on the clusters and their relationship within the image.

The second axis deals with collaborative clustering methods developed with the goal of being applicable to as many clustering algorithms as possible, including the algorithms used in the first axis of this work. A key feature of the methods proposed in this thesis is that they can deal with either of the following two cases: 1) several clustering algorithms working together on the same data represented in different feature spaces, 2) several clustering algorithms looking for similar clusters in different data sets having similar distributions. Clustering algorithms to which these methods are applicable include the ICM algorithm, the K-Means algorithm, density based algorithms such as DB-scan, all Expectation-Maximization (EM) based algorithms such as the Self-Organizing Maps (SOM) and the Generative Topographic Mapping (GTM) algorithms. Unlike previously introduced methods, our models have no restrictions in term of types of algorithms that can collaborate together, do not require that all methods be looking for the same number of clusters, and are provided with solid mathematical foundations.