Université Paris 13, Sorbonne Paris Cité

Thèse

Contributions à l'Apprentissage Collaboratif non Supervisé

Présentée par :

Mohamad Ghassany

Pour obtenir le grade de Docteur d'université

Spécialité: Informatique

Laboratoire d'Informatique de Paris Nord

UMR 7030 du CNRS

École doctorale Galilée

Soutenue le 07/11/2013 devant le jury :

Baydaa AL-AYOUBI (*Rapportrice*)	Professeur, Université Libanaise
Younès BENNANI (*Directeur de thèse*)	Professeur, Université Paris 13
Antoine CORNUÉJOLS (*Rapporteur*)	Professeur, AgroParisTech
Christophe FOUQUERÉ (*Examinateur*)	Professeur, Université Paris 13
Marc GELGON (*Examinateur*)	Professeur, Polytech'Nantes
Nistor GROZAVU (*Co-encadrant*)	Maître de conférences, Université Paris 13
Cedric WEMMERT (*Examinateur*)	Maître de conférences, HDR, Université de Strasbourg

Université Paris 13, Sorbonne Paris Cité

Doctoral Thesis

Contributions to Collaborative Clustering

Author:
Mohamad Ghassany

A thesis submitted for the degree of Doctor of Philosophy

Laboratoire d'Informatique de Paris Nord
UMR 7030 of CNRS
École doctorale Galilée

Doctoral Committee:

Baydaa AL-AYOUBI (*Reader*)	Professor, Université Libanaise
Younès BENNANI (*Supervisor*)	Professor, Université Paris 13
Antoine CORNUÉJOLS (*Reader*)	Professor, AgroParisTech
Christophe FOUQUERÉ (*Examiner*)	Professor, Université Paris 13
Marc GELGON (*Examiner*)	Professor, Polytech'Nantes
Nistor GROZAVU (*Co-advisor*)	Assistant Professor, Université Paris 13
Cedric WEMMERT (*Examiner*)	Assistant Professor, HDR, Université de Strasbourg

"If Data Had Mass, the Earth Would Be a Black Hole"

Stephen Marsland

UNIVERSITÉ PARIS 13, SORBONNE PARIS CITÉ

Résumé

Docteur d'Université

Contributions à l'Apprentissage Collaboratif non Supervisé

par Mohamad Ghassany

Le travail de recherche exposé dans cette thèse concerne le développement d'approches de clustering collaboratif à base de méthodes topologiques, telles que les cartes auto-organisatrices (SOM), les cartes topographiques génératives (GTM) et les GTM variationnelles Bayésiennes (VBGTM). Le clustering collaboratif permet de préserver la confidentialité des données en utilisant d'autres résultats de classifications sans avoir recours aux données de ces dernières. Ayant une collection de bases de données distribuées sur plusieurs sites différents, le problème consiste à partitionner chacune de ces bases en considérant les données locales et les classifications distantes des autres bases collaboratrices, sans partage de données entre les différents centres. Le principe fondamental du clustering collaboratif est d'appliquer les algorithmes de clustering localement sur les différents sites, puis collaborer les sites en partageant les résultats obtenus lors de la phase locale. Dans cette thèse nous explorons deux approches pour le clustering collaboratif. L'approche horizontale pour la collaboration des bases de données qui décrivent les mêmes individus mais avec des variables différentes. La deuxième approche collaborative est dite verticale pour la collaboration de plusieurs bases de données contenant les mêmes variables mais avec des populations différentes.

UNIVERSITÉ PARIS 13, SORBONNE PARIS CITÉ

Abstract

Doctor of Philosophy

Contributions To Collaborative Clustering

by Mohamad Ghassany

The research outlined in this thesis concerns the development of collaborative clustering approaches based on topological methods, such as self-organizing maps (SOM), generative topographic mappings (GTM) and variational Bayesian GTM (VBGTM). So far, clustering methods performs on a single data set, but recent applications require data sets distributed among several sites. So, communication between the different data sets is necessary, while respecting the privacy of every site, i.e. sharing data between sites is not allowed. The fundamental concept of collaborative clustering is that the clustering algorithms operate locally on individual data sets, but collaborate by exchanging information about their findings. The strength of collaboration, or confidence, is precised by a parameter called coefficient of collaboration. This thesis proposes to learn it automatically during the collaboration phase. Two data scenarios are treated in this thesis, referred as vertical and horizontal collaboration. The vertical collaboration occurs when data sets contain different objects and same patterns. The horizontal collaboration occurs when they have same objects and described by different patterns.

Contents

Dedicated to my Grandmother, may her soul rest in peace.

Avant-Propos

Contexte et problématique

“Qui se ressemble s’assemble”.

La classification non supervisée, ou Clustering, est une approche importante en analyse exploratoire de données non étiquetées. Sans connaissances a priori sur la structure d’une base de données, l’objectif de la classification non supervisée est de détecter automatiquement la présence de sous-groupes pertinents (ou clusters). Un cluster peut être défini comme un ensemble de données similaires entre elles et peu similaires avec les données appartenant à un autre cluster (homogénéité interne et séparation externe).

Dans cette thèse, nous nous plaçons dans une situation où nous avons une collection d’ensembles de données existantes à différents sites. Il pourrait s’agir de données décrivant les clients des institutions bancaires, magasins, ou des organisations médicales. Les données pourraient inclure des données concernant différents individus. Elles pourraient représenter les mêmes personnes, mais avec différents descripteurs (attributs) reflétant les activités de l’organisation. Le but ultime de chaque organisation est de découvrir les principales relations dans son ensemble de données. Cette découverte peut être raffinée en tenant compte des dépendances entre les différentes analyses effectuées par les différents sites, afin de produire une image fidèle de la structure globale cachée dans les différentes bases de données sans en avoir un accès direct. Dans certains cas, il pourrait y avoir aussi des problèmes techniques, la classification d’un grand ensemble de données unique ne peut pas être réalisable. Une approche collaborative permettrait de distribuer les classifications et procéder à une fusion des différents résultats. Dans cette thèse, nous nous intéressons au problème de l’apprentissage non supervisé (clustering) et spécifiquement au clustering collaboratif en préservant la confidentialité des données et en utilisant des modèles de classification à base de prototypes et permettant la visualisation des données. Ayant une collection de bases de données distribuées sur plusieurs sites différents, le problème consiste à partitionner chacune de ces bases en considérant les données locales et les classifications obtenues par les autres sites pour améliorer/enrichir la classification locale, sans toutefois avoir recours au partage de données entre les différents centres.

Nous explorons dans cette thèse deux approches pour la classification non supervisée collaborative entre plusieurs classifications issues de plusieurs jeux de données distants. L'approche horizontale pour la collaboration des bases de données qui décrivent les mêmes individus mais avec des variables différentes. Cette approche peut être vue comme une classification multi-vues où le traitement se fait sur des données multi-représentées, c'est à dire sur un même ensemble d'individus mais décrits par plusieurs représentations. La deuxième approche collaborative est dite verticale pour la collaboration de plusieurs bases de données contenant les mêmes variables mais avec des populations différentes.

Durant la phase de collaboration, nous n'avons pas besoin des bases de données mais uniquement des résultats des classifications distantes. Ainsi, sur chaque site, on utilise la base de données locale et les informations des autres classifications distantes, ce qui permettrait d'obtenir une nouvelle classification qui soit le plus proche possible de celle qu'on aurait obtenue si on avait centralisé les bases de données et faire un partitionnement ensuite.

Nous nous intéressons en particulier aux méthodes de réduction de dimensionnalité et de visualisation de données. La quantité de données enregistrées et stockées dans la société ne cesse de croître. Cependant, sans les moyens et les méthodes qui peuvent aider à l'analyse, la quantité de données devient inutile. On ne peut rien analyser quand on regarde les données brutes, par exemple, des tableaux de chiffres et de symboles ou un grand nombre d'images similaires. Nous avons donc besoin d'ordinateurs pour nous aider, non seulement dans la collecte et le stockage de données, mais aussi dans l'analyse et le traitement de celles-ci. Notamment, si l'ordinateur peut être utilisé pour synthétiser les données visuellement, les humains sont souvent capable d'interpréter ces graphiques intelligemment.

Dans cette thèse nous proposons des algorithmes de clustering collaboratif basés sur des méthodes de classification à base des prototypes. Les méthodes utilisées sont les cartes auto-organisatrices (SOM), cartes topographiques génératives (GTM), et les GTM Variationnelles Bayésiennes (VBGTM). Une caractéristique commune entre ces trois méthodes c'est la réduction de dimensionnalité d'un espace de données de grande dimension (> 3) à un espace de basse dimension, généralement de dimension 2 pour permettre la visualisation. Toutes les données dans cette thèse sont numériques.

Organisation de la thèse

Ce manuscrit est organisé en quatres chapitres principaux encadrés par une introduction et une conclusion générale.

Chapitre 1: Clustering Collaboratif Flou.

Nous commençons ce chapitre par une présentation générale du principe de l'apprentissage automatique des données, nous détaillons en particulier le cas de l'apprentissage non supervisé (clustering), où les données ne sont pas étiquetées et aucune information a priori n'est disponible. Après, nous présentons le cas où les données sont distribuées sur plusieurs sites et le besoin de formuler un algorithme qui traite ces données de manière séparée, conservant ainsi leur confidentialité. Mais les méthodes distribuées standards traitent tous les sites de données en un seul coup, sans tenir compte de l'importance d'un site à un autre. Tandis que l'apprentissage collaboratif permet une amélioration locale des résultats. Durant la phase de collaboration les sites partagent leurs paramètres entre eux et "estiment" la confiance qu'ils font aux autres sites. Selon cette confiance, nous pourrons procéder à l'amélioration des résultats locaux. L'algorithme standard du clustering collaboratif présenté dans ce chapitre est basé sur la méthode de clustering *flou*, où les données peuvent être attribuées à plusieurs groupes avec une certaine probabilité. Nous présentons les deux approches horizontale et verticale. Ce chapitre représente un point de départ pour les autres chapitres de la thèse.

Chapitre 2: Clustering Collaboratif basé sur SOM.

Dans ce chapitre, nous formulons deux algorithmes de clustering collaboratif en se basant sur les cartes auto-organisatrices (SOM) comme méthode de clustering et de visualisation. Une SOM est un algorithme neuro-inspiré (inspiré du fonctionnement des neurones biologiques) qui permet la projection non linéaire de données de grandes dimensions dans un espace à deux dimensions par l'intermédiaire d'un apprentissage non supervisé compétitif. Cet algorithme est efficace pour la réduction de dimensions et donc pour la visualisation des données sur une carte en deux dimensions. La carte est composée d'un ensemble de prototypes qui, à la fin de l'apprentissage, représentent les données et leur structure. Dans l'algorithme que nous proposons dans ce chapitre, nous ajoutons une étape à la phase de collaboration. Cette phase permet d'estimer automatiquement la meilleure valeur du coefficient de collaboration. Nous présentons les deux cas de collaboration, horizontale et verticale. Nous testons nos algorithmes sur quatre jeux de données du site UCI. Nous validons nos résultats en utilisant des critères comme l'erreur de quantification et l'indice de purité.

Chapitre 3: Clustering Collaboratif basé sur un modèle génératif.

Malgré que SOM soit une méthode très populaire, elle souffre de plusieurs limitations dont l'abscence d'un modèle probibiliste en particulier. Pour cela, nous proposons dans ce chapitre un clustering collaboratif basé sur une méthode concurrente à SOM, soit les cartes topographiques génératives (GTM). GTM est basée sur un modèle génératif non linéaire. GTM a été définie pour conserver toutes les propriétés utiles de SOM, comme le clustering et la visualisation de données multidimensionnelles, tout en évitant le plus de ses limitations grâce à une formulation entièrement probabiliste. Dans ce chapitre, nous décrivons d'abord qu'est-ce qu'un modèle génératif, l'algorithme EM en particulier. Nous décrivons le modèle original de GTM et nous le comparons à SOM. Ensuite, nous proposons un algorithme de clustering collaboratif basé sur GTM. Pour le faire, nous modifions l'étape M de l'algorithme EM en ajoutant un terme de collaboration à l'espérance de la vraisemblance, celà conduit à des modifications dans les formules de mise à jour des paramètres de GTM. Nous validons nos approches par quelques expériences en utilisant des critères internes et externes.

Chapitre 4: Clustering Collaboratif Flou des GTM Variationnelles.

L'optimisation des paramètres du modèle GTM par l'agorithme EM ne tient pas en compte la complexité du modèle et, par conséquent, le risque de sur-apprentissage des données est élevé. Une solution élégante pour éviter le sur-apprentissage des GTM est d'approximer GTM avec une vision variationnelle, la méthode est appelée GTM variationnelle Bayésienne (VBGTM). Dans ce chapitre, nous décrivons d'abord l'inférence variationnelle, puis nous décrivons la VBGTM. Ensuite, nous proposons un algorithme qui combine VBGTM et FCM pour effectuer la classification et la visualisation des données en même temps, nous l'appelons F-VBGTM. Enfin, nous proposons deux approches horizontale et verticale de clustering collaboratif basées sur le F-VBGTM. Un exemple de l'effet de la collaboration est présenté. Nous présentons également quelques méthodes de calcul automatique du coefficient de collaboration pendant la phase de collaboration.

Nous concluons cette thèse en exposant les points forts de nos contributions et les perspectives de recherche dans ce domaine.

Introduction

Clustering is the process of partitioning a set of data objects (or observations) into subsets. It can be considered the most important *unsupervised learning* problem; so, as every other problem of this kind, it deals with finding a structure in a collection of unlabeled data. A loose definition of clustering could be “the process of organizing objects into groups whose members are similar in some way”. A cluster is therefore a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters.

So far, clustering has operated on a single data set. Nowadays, computing environments and technologies are more and more evolving towards a mobile, finely distributed, interacting, dynamic environment containing massive amounts of heterogeneous, spatially and temporally *distributed data* sources. In many companies data is distributed among several sites, i.e. each site generates its own data and manages its own data repository. Analyzing these distributed sources requires distributed clustering techniques to find global patterns representing the complete information. The transmission of the entire local data set is often unacceptable because of performance considerations, privacy and security aspects, and bandwidth constraints. Traditional clustering algorithms, demanding access to complete data, are not appropriate for distributed applications. Thus, there is a need for distributed clustering algorithms in order to analyze and discover new knowledge in distributed environments.

In this thesis, we discuss a situation that arises when data is distributed over several data sets. The structure we are seeking to reveal concerns all of them, but they need to be processed separately. This leads to the fundamental concept of *Collaborative Clustering*: the clustering algorithms operate locally (namely, on individual data sets) but collaborate by exchanging information about their findings. Thus, each site uses its clustering results and the information from other clustering, which would provide a new clustering that is as close as possible to that which would be obtained if we had centralized the data sets. We formulate our algorithms in two fundamental data scenarios referred to as vertical and horizontal collaboration. The vertical collaboration occurs when data sets contain different objects and same patterns. The horizontal collaboration occurs when they have same objects and described by different patterns, this scenario is more difficult because different patterns include different feature space dimension. The strength of collaboration is precised by a parameter which we propose to learn automatically during the collaboration procedure. This parameter quantifies the confidence between data sites, precising the strength of contribution of each site in the consensus building procedure.

Beside the collaborative clustering algorithms, we are interested in this thesis in *dimensionality reduction* and *data visualization* methods. The amount of data being recorded and stored throughout society is steadily growing. However, without means and methods that can aid analysis, much data becomes useless. Human observers often find it hard spotting regularities when looking at raw data, e.g. tables of numbers and symbols or large numbers of similar images. We therefore need computers to aid us, not only in the gathering and storing of data, but also in the analysis and processing of it. In particular, if the computer can be used to summarize data visually, humans are often capable of interpreting such graphical summaries intelligently.

Scope of the thesis

This thesis is concerned in formulating collaborative clustering algorithms using computational methods for finding ‘interesting’ structures in sets of data, with little or no need of human intervention or guidance. The methods are: Self-Organizing Maps (SOM), Generative Topographic Mapping (GTM) and Variational Bayesian Generative Topographic Mapping (VBGTM). A key feature of these methods is that they involve some sort of dimensionality reduction, from the, typically high-dimensional, data space to a low-dimensional model space defined by the method used. When visualization is the ultimate aim, the model space is typically chosen to be two-dimensional. In this thesis, both the data space and the model space are taken to be subsets of $\mathbb{R}^\infty$.

Overview of the thesis

This thesis is structured into four chapters and organized as follows:

Chapter 1: Collaborative Fuzzy Clustering.

This chapter aims to introduce the basics of Machine Learning and its different types, we describe in particular the type we are interested in, which is Clustering. Then we introduce the principle of distributed data clustering, where data is distributed over different sources and the need of an algorithm to reveal a common structure of all data, taking into consideration the confidentiality of data, e.g. preserving its privacy. We introduce the original standard collaborative clustering based on the Fuzzy *c*-means algorithm (FCM), which treats different data sets separately and then collaborate them by exchanging information about their findings. As well as the horizontal and vertical collaboration scenarios, as starting point for the next chapters.

Chapter 2: Collaborative Clustering using Self-Organizing Maps.

This chapter starts by describing the Self-Organizing Maps (SOM), a very popular method that aims to discover some underlying structure of the data. SOM is called a topology-preserving map because there is a topological structure imposed on the nodes in the network. A topological map is simply a mapping that preserves neighborhood relations. After describing SOM, we present our contribution: a collaborative clustering algorithm based on SOM, both horizontal and vertical collaboration are presented. In addition, we propose to learn automatically the coefficient of collaboration (also called confidence parameter) during the collaboration process. We complete the chapter with experiments. We test our algorithms on four UCI data sets, using many validation criteria. Good and promising results are shown by tables and figures.

Chapter 3: Collaborative Clustering using A Generative Model.

Despite that SOM has become very popular and was applied in several domains, SOM suffers from many limitations. This is why we present a collaborative clustering scheme using a concurrent method to SOM, the Generative Topographic Mapping (GTM). GTM is a non-linear generative model. It was defined to retain all the useful properties of SOM, such as the simultaneous clustering and visualization of multivariate data, while eluding most of its limitations through a fully probabilistic formulation. In this chapter, we describe first what is a generative model, the EM algorithm in particular since GTM is based on it. We describe the original GTM model and compare it to SOM. Then we propose a collaborative clustering scheme based on GTM. To collaborate using GTM, a modification in the M-step of its EM algorithm is proposed. We validate our approaches by some experiments.

Chapter 4: Collaborative Fuzzy Clustering of Variational Bayesian GTM.

The optimization of the GTM model parameters through EM does not take into account model complexity and consequently, the risk of data overfitting is elevated. An elegant solution to avoid overfitting was proposed by applying the variational approximation framework on GTM, it is called Variational Bayesian

GTM (VBGTM). First in this chapter, we describe the Variational Bayesian Inference, then we describe the VBGTM. Next, we propose an algorithm that combines VBGTM and FCM to do data visualization and grouping at the same time, we call it F-VBGTM. Finally, we propose an horizontal and a vertical collaborative clustering schemes based on F-VBGTM. An example of the effect of the collaboration is presented. We present also some methods for calculating the collaboration coefficients during the collaboration stage.

We finish this thesis by a conclusion with a summary of its main contributions. Furthermore, a discussion on future directions, as well as of open questions of research, is summarily outlined.

Publications effectuées pendant la thèse

Main publications resulting from the thesis

International Journals:

- *Collaborative Clustering Using Prototype-Based Techniques.* International Journal of Computational Intelligence and Applications 11, 03 (2012), 1250017.
- *Collaborative Fuzzy Clustering of Variational Bayesian GTM.* International Journal of Computational Intelligence and Applications (2013), *submitted.*

International Conferences:

- *Learning Confidence Exchange in Collaborative Clustering.* In Neural Networks (IJCNN), The 2011 International Joint Conference on (2011), pp. 872879.
- *Collaborative Generative Topographic Mapping.* In Neural Information Processing, vol. 7664 of Lecture Notes in Computer Science. Proc of ICONIP'12. Springer Berlin Heidelberg, 2012, pp. 591598.
- *Collaborative Multi-View Clustering.* In Neural Networks (IJCNN), The 2013 International Joint Conference on (2013), pp. 872879.
- *Apprentissage de la confiance des échanges en classification collaborative non supervisée*, in Proc. of CAP, Conférence Francophone d'Apprentissage , Chambéry, (2011).

Chapter 1

Collaborative Fuzzy Clustering

Machine Learning

One of the most interesting features of machine learning is that it lies on the boundary of several different academic disciplines, principally computer science, statistics, mathematics, and engineering. This has been a problem as well as an asset, since these groups have traditionally not talked to each other very much. To make it even worse, the areas where machine learning methods can be applied vary even more widely, from finance and business [31] to biology [9, 71, 91] and medicine [117, 123] to physics and chemistry [38] and beyond [178].

Around the world, computers capture and store terabytes of data every day. There are computers belonging to shops, banks, hospitals, scientific laboratories, and many more that are storing data incessantly. For example, banks are building up pictures of how people spend their money, hospitals are recording what treatments patients are on for which ailments (and how they respond to them). The challenge is to do something useful with this data: if the bank's computers can learn about spending patterns, can they detect credit card fraud quickly? If hospitals share information, then can treatments that don't work as well as expected be identified quickly? These are some of the questions that machine learning methods can be used to answer.

Science has also taken advantage of the ability of computers to store massive amounts of data. Biology has led the way, with the ability to measure gene expression in DNA microarrays producing immense data sets [42], along with protein transcription data and phylogenetic trees relating species to each other. However, other sciences have not been slow to follow. Astronomy [10] now uses digital telescopes, so that each night the world's observatories are storing incredibly high-resolution images of the night sky,

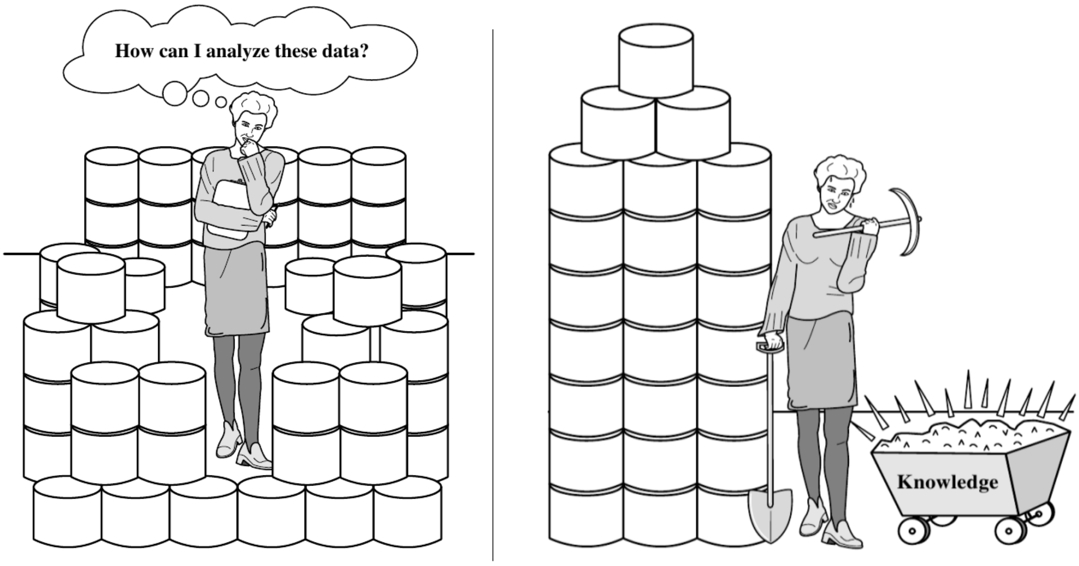


FIGURE 1.1: (left) The world is data rich but information poor. (right) Data mining: searching for knowledge (interesting patterns) in data. [79]

around a terabyte per night. Equally, medical science stores the outcomes of medical tests from measurements as diverse as Magnetic Resonance Imaging (MRI) [152] scan and simple blood tests. The explosion in stored data is well known; the challenge is to do something useful with that data.

The size and complexity of these data sets means that humans are unable to extract useful information from them. Even the way that the data is stored works against us. Given a file full of numbers, our minds generally turn away from looking at them for long. Take some of the same data and plot it in a graph and we can do something, the graph is rather easier to look at and deal with. Unfortunately, our three-dimensional world doesn't let us do much with data in higher dimensions. This is known as the **curse of dimensionality**. There are two things that we can do with this: reduce the number of dimensions (until our simple brains can deal with the problem) or use computers, which don't know that high-dimensional problems are difficult, and don't get bored with looking at a massive data files of numbers.

This is one reason why **machine learning** is becoming so popular. The problems of our human limitations go away if we can make computers do the dirty work for us.

Machine learning, then, is about making computers `modify` or `adapt` their actions so that these actions get more accurate, where accuracy is measured by how well the chosen action reflect the correct ones. It is only over the past decade or so that the inherent multi-disciplinarity of machine learning has been recognized. It merges ideas from neuroscience and biology, statistics, mathematics, and physics, to make computers learn.

Application of machine learning methods to large databases is called **data mining** [58, 79, 82, 178]. The analogy is that large volume of earth and raw material is extracted from a mine, which when processed leads to a small amount of very precious material; similarly, in data mining, a large volume of data is processed to construct a simple model with valuable use, for example, having high predictive accuracy. Its application areas are abundant: In addition to retail, in finance banks analyze their past data to build models to use in credit applications, fraud detection, and the stock market. In manufacturing, learning models are used for optimization, control, and troubleshooting. In medicine, learning programs are used for medical diagnosis. In telecommunications, call patterns are analyzed for network optimization and maximizing the quality of service. In science, large amounts of data in physics, astronomy, and biology can only be analyzed fast enough by computers. The World Wide Web is huge; it is constantly growing, and searching for relevant information cannot be done manually. Figure 1.1 illustrates the phenomena of data mining.

Example 1 A search engine (e.g., Google) receives hundreds of millions of queries every day. Each query can be viewed as a transaction where the user describes her or his information need. What novel and useful knowledge can a search engine learn from such a huge collection of queries collected from users over time? Interestingly, some patterns found in user search queries can disclose invaluable knowledge that cannot be obtained by reading individual data items alone. For example, Google's Flu Trends uses specific search terms as indicators of flu activity. It found a close relationship between the number of people who search for flu-related information and the number of people who actually have flu symptoms. A pattern emerges when all of the search queries related to flu are aggregated. Using aggregated Google search data, Flu Trends can estimate flu activity up to two weeks faster than traditional systems can [70]. This example shows how data mining can turn a large collection of data into knowledge that can help meet a current global challenge.

Example 2 A supermarket chain that has hundreds of stores all over a country selling thousands of goods to millions of customers. The point of sale terminals record the details of each transaction: date, customer identification code, goods bought and their amount, total money spent, and so forth. This typically amounts to gigabytes of data every day. What the supermarket chain wants is to be able to predict who are the likely customers for a product. Again, the algorithm for this is not evident; it changes in time and by geographic location. The stored data

> becomes useful only when it is analyzed and turned into information that we can make use of, for example, to make predictions.
>
> We do not know exactly which people are likely to buy this ice cream flavor, or the next book of this author, or see this new movie, or visit this city, or click this link. If we knew, we would not need any analysis of the data; we would just go ahead and write down the code. But because we do not, we can only collect data and hope to extract the answers to these and similar questions from data.
>
> We do believe that there is a process that explains the data we observe. Though we do not know the details of the process underlying the generation of data, for example customer behavior, we know that it is not completely random. People do not go to supermarkets and buy things at random.
>
> We may not be able to identify the process completely, but we believe we can construct a good and useful approximation. That approximation may not explain everything, but may still be able to account for some part of the data. We believe that though identifying the complete process may not be possible, we can still detect certain patterns or regularities. This is the niche of machine learning. Such patterns may help us understand the process, or we can use those patterns to make predictions: Assuming that the future, at least the near future, will not be much different from the past when the sample data was collected, the future predictions can also be expected to be right.

Based on the available information and on the desired objectives, there are different types of Machine Learning Algorithms:

- **Supervised learning (Classification)**: A training set of examples with the correct responses (targets) are provided and, based on this training set, the algorithm generalizes to respond correctly to all possible inputs. This is also called learning from exemplars. [37, 64, 118]
- **Unsupervised learning (Clustering)**: Correct responses are not provided, instead the algorithm tries to identify similarities between the inputs so that inputs that have something in common are categorized together. [12, 80, 90, 97]
- **Reinforcement learning**: This is somewhere between supervised and unsupervised learning. The algorithm gets told when the answer is wrong, but does not get told how to correct it. It has to explore and try out different possibilities until it works out how to get the answer right . Reinforcement learning is sometimes called learning with a critic because of this monitore that scores the answer, but does not suggest improvements. [101, 165, 177]

- **Evolutionary learning**: Biological evolution can be seen as a learning process: biological organisms adapt to improve their survival rates and chance of having offspring in their environment. [29, 126]

The Unsupervised Learning (Clustering) is going to be the focus of this thesis. Data Visualization equally. So, we'll have a look at what it is, and the kinds of problems that can be solved using it.

1.1 Fuzzy Clustering

Clustering (or Cluster analysis) is the process of partitioning a set of data objects (or observations) into subsets. Each subset is a **cluster**, such that objects in a cluster are similar to one another, yet dissimilar to objects in other clusters. The set of clusters resulting from a cluster analysis can be referred to as a clustering. In this context, different clustering methods may generate different clusterings on the same data set. The partitioning is not performed by humans, but by the clustering algorithm. Hence, clustering is useful in that it can lead to the discovery of previously unknown groups within the data.

Example Imagine a Director of Customer Relationships at an Electronics store, and he has five managers working for him. He would like to organize all the company's customers into five groups so that each group can be assigned to a different manager. Strategically, he would like that the customers in each group are as similar as possible. Moreover, two given customers having very different business patterns should not be placed in the same group. His intention behind this business strategy is to develop customer relationship campaigns that specifically target each group, based on common features shared by the customers per group. Unlike in classification, the class label of each customer is unknown. He needs to discover these groupings. Given a large number of customers and many attributes describing customer profiles, it can be very costly or even unfeasible to have a human study the data and manually come up with a way to partition the customers into strategic groups. He needs a *clustering* tool to help.

Clustering has been widely used in many applications such as business intelligence, image pattern recognition, Web search, biology, and security. In business intelligence, clustering can be used to organize a large number of customers into groups, where customers within a group share strong similar characteristics. In image recognition, clustering can be used to discover clusters or "subclasses" in handwritten character recognition systems. Suppose we have a data set of handwritten digits, where each digit is labeled as either 1, 2, 3, and so on. Clustering has also found many applications in Web search. 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 concise and easily accessible way. Moreover, clustering techniques have been developed to cluster documents into topics, which are commonly used in information retrieval practice.

Clustering is also called **data segmentation** in some applications because clustering partitions large data sets into groups according to their *similarity*. Clustering can also be used for **outlier detection** [81, 89, 160], where outliers (values that are "far away" from any cluster) may be more interesting than common cases. Applications of outlier 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 [78].

As a branch of statistics, clustering has been extensively studied, with the main focus on *distance-based cluster analysis*. Clustering tools were proposed like K-means, fuzzy C-means, and several other methods. In data mining, efforts have focused on finding methods for efficient and effective cluster analysis in *large databases*. Active themes of research focus on the *scalability* of clustering methods, the effectiveness of methods for clustering *complex shapes* (e.g., nonconvex) and *types of data* (e.g., text, graphs, and images), *high-dimensional* clustering techniques (e.g., clustering objects with thousands of features), and methods for clustering *mixed numerical and nominal data* in large databases.

The simplest and most fundamental version of cluster analysis is partitioning, which organizes the objects of a set into several exclusive groups or clusters. To keep the problem specification concise, we can assume that the number of clusters is given as background knowledge. This parameter is the starting point for partitioning methods.

The basic notions of data, clusters and cluster prototypes [30] are established and a broad overview of different clustering approaches is given.

The Data Set

Clustering techniques can be applied to data that are quantitative (numerical), qualitative (categorical), or a mixture of both. In this thesis, the clustering of quantitative data is considered. The data are typically a number of observations. Each observation consists of D measured variables, grouped into a D-dimensional row vector $x_n = [x_{n1}, \ldots, x_{nD}]$, $x_n \in \mathbb{R}^D$. A set of N observations is denoted by $\mathbf{X} = \{x_n | n = 1, \ldots, N\}$, and is represented as a $N \times D$ matrix:

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \ldots & x_{1D} \\ x_{21} & x_{22} & \ldots & x_{2D} \\ \vdots & \vdots & \vdots & \vdots \\ x_{N1} & x_{N2} & \ldots & x_{ND} \end{bmatrix} \tag{1.1}$$

The rows of this matrix are called *patterns* or objects, the columns are called the *features* or attributes, and $\mathbf{X}$ is called the *data matrix*.

UCI: There is a very useful resource for machine learning in the `UCI Machine Learning Repository` [7]. This website hold lots of data sets that can be downloaded and used for experimenting with different machine learning algorithms and seeing how well they work. By using these test data sets for experimenting with the algorithms, we do not have to worry about getting hold of suitable data and pre-processing it into a suitable form for learning. This is typically a large part of any real problem, but it gets in the way of learning about the algorithms.

Many clustering algorithms have been introduced in the literature. Since clusters can formally be seen as subsets of the data set, one possible classification of clustering methods can be according to whether the subsets are *fuzzy* or *crisp* (hard).

Hard clustering methods are based on classical set theory, and require that an object either does or does not belong to a cluster. Hard clustering means partitioning the data into a specified number of mutually exclusive subsets. The most common hard clustering method is k-means, which is described in section 1.1.

Fuzzy clustering methods, however, allow the objects to belong to several clusters simultaneously, with different degrees of membership. In many situations, fuzzy clustering is more natural than hard clustering. The most know technique of fuzzy clustering is the fuzzy c-means, it is described in section 1.1.

K-Means

If you have ever watched a group of tourists with a couple of tour guides who hold umbrellas up so that everybody can see them and follow them, then you have seen a dynamic version of the K-means algorithm. K-means is even simpler, because the data (playing the part of the tourists) does not move, only the tour guides move.

Suppose that we want to divide our input data into K categories, where we know the value of K. We allocate K *cluster centres* (also called *prototypes*) to our input space, and we would like to position these centres so that there is one cluster centre in the middle of each cluster. However, we don't know where the clusters are, let alone where their 'middle' is, so we need an algorithm that will find them. Learning algorithms generally try to minimize some sort of error, so we need to think of an error criterion that describes this aim. There are two things that we need to define:

A distance measure: In order to talk about distances between points, we need some way to measure distances. It is often the normal Euclidean distance, but there are other alternatives like Manhattan distance, Correlation distance, Chessboard distance and other.

The mean average: Once we have a distance measure, we can compute the central point of a set of data points, which is the mean average. Actually, this is only true in Euclidean space, which is the one we are used to, where everything is nice and flat. Everything becomes a lot trickier if we have to think about curved spaces; when we have to worry about curvature, the Euclidean distance metric isn't the right one, and there are at least two different definitions of the mean. So we aren't going to worry about any of these things, and we'll assume that space is flat. This is what statisticians do all the time.

We can now think about a suitable way of positioning the cluster centres: we compute the mean point of each cluster, $\mathbf{v}_i$, $i = 1, \ldots, K$, and put the cluster centre there. This is equivalent to minimizing the Euclidean distance (which is the sum-of-squares error) from each data point to its cluster centre. Then we decide which points belong to which clusters by associating each point with the cluster centre that it is closest to. This changes as the algorithm iterates. We start by positioning the cluster centres randomly though the input space, since we don't know where to put them, and we update their positions according to the data. We decide which cluster each data point belongs to by computing the distance between each data point and all of the cluster centres, and assigning it to the cluster that is the closest. For all the points that are assigned to a cluster, we then compute the mean of them, and move the cluster centre to that place. We iterate the algorithm until the cluster centres stop moving.

It is convenient at this point to define some notation to describe the assignment of data points to clusters. For each data point x_k, we introduce a corresponding set of binary indicator variables $u_{ik} \in 0, 1$, where $i = 1, \ldots, K$ describing which of the K clusters the data point x_k is assigned to, so that if data point x_k is assigned to cluster i then $u_{ik} = 1$, and $u_{jk} = 0$ for $j \neq i$. This is known as the 1-of-K coding scheme. We can then define an objective function (and sometimes called a *distortion measure*), given by

$$J = \sum_{k=1}^{N} \sum_{i=1}^{K} u_{ik} \|x_k - \mathbf{v}_i\|^2 \tag{1.2}$$

which represents the sum of the squares of the distances of each data point to its assigned vector $\mathbf{v}_i$. The goal is to find values for the $\{u_{ik}\}$ and the $\{\mathbf{v}_i\}$ so as to minimize J. We can do this through an iterative procedure in which each iteration involves two successive steps corresponding to successive optimizations with respect to the u_{ik} and the $\mathbf{v}_i$. The algorithm of K-means is described in Algorithm 1.

Algorithm 1: The K-Means Algorithm

```
Data: $\mathbf{X} = \{x_{kd},\ \ k = 1, \dots, N, d = 1, \dots, D\}$ where $D$ is the dimension of the feature
      space.
Result: Cluster centres (Prototypes)
Initialization:
-Choose a value for $K$
-Choose $K$ random positions in the input space
-Assign the prototypes $\mathbf{v}_i$ to those positions.
Learning: repeat
for each data point $x_k$ do
    -compute the distance to each prototype:
        $d_{ik} = \min_i d(x_k, \mathbf{v}_i)$
    -assign the data point to the nearest prototype with distance
        $u_{ik} = \begin{cases} 1 & \text{if } i = \arg\min_j \|x_k - \mathbf{v}_j\|^2 \\ 0 & \text{otherwise} \end{cases}$   (1.3)
for each prototype do
    -move the position of the prototype to the mean of the points in that cluster:
        $\mathbf{v}_i = \frac{\sum_k u_{ik} x_k}{\sum_k u_{ik}}$   (1.4)
Until the prototypes stop moving.
```

The denominator in the expression 1.4 is equal to the number of points assigned to cluster i, and so this result has a simple interpretation, namely set $\mathbf{v}_i$ equal to the mean of all of the data points x_k assigned to cluster i. For this reason, the procedure is known as the K-means algorithm.

The two phases of re-assigning data points to clusters and re-computing the cluster means are repeated in turn until there is no further change in the assignments (or until

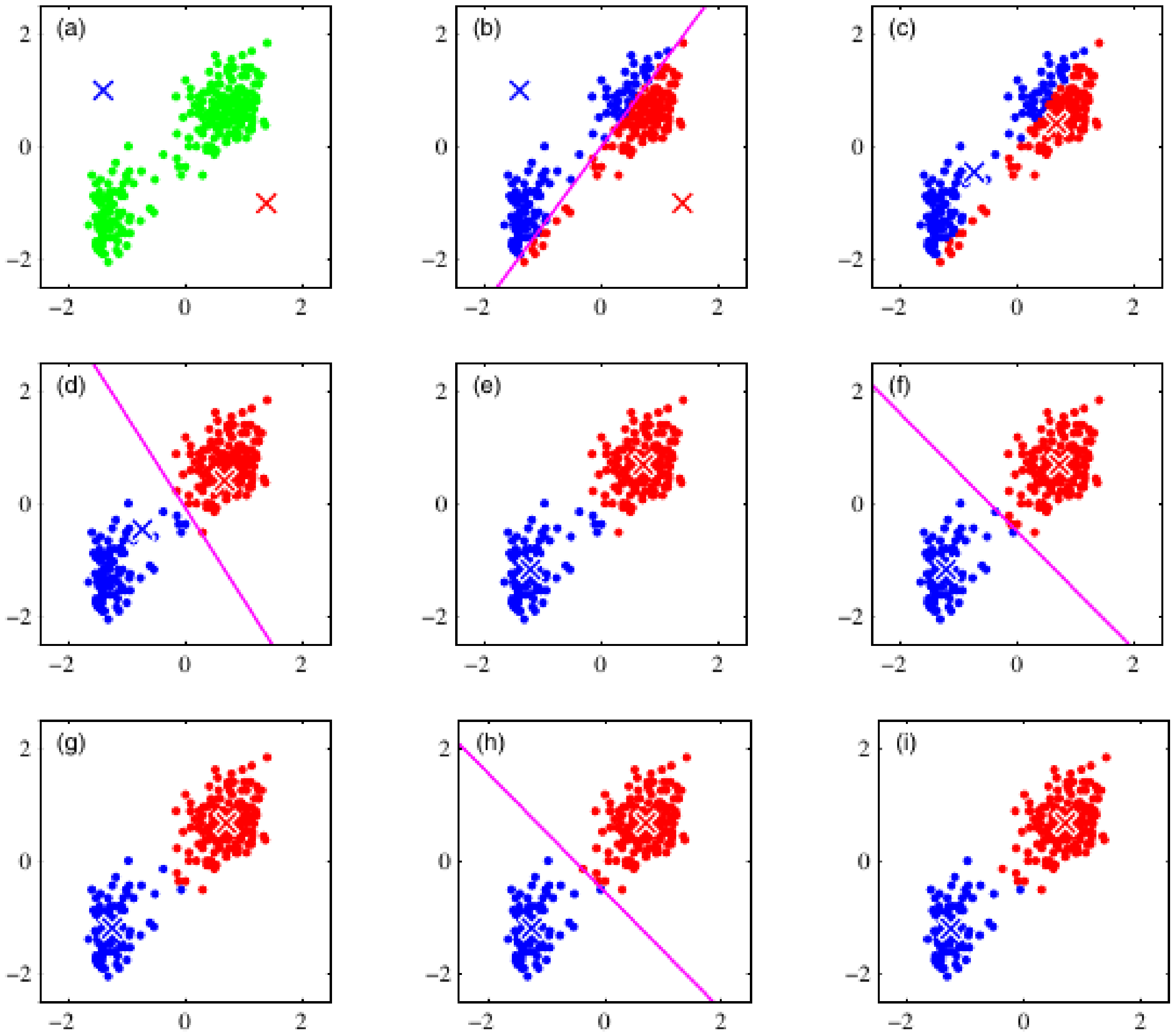


FIGURE 1.2: Illustration of the K-means algorithm using the re-scaled Old Faithful data set, where $K = 2$. [24]

some maximum number of iterations is exceeded). Because each phase reduces the value of the objective function J, convergence of the algorithm is assured. However, it may converge to a local rather than global minimum of J. The convergence properties of the K-means algorithm were studied by [129].

The K-means algorithm is illustrated using the Old Faithful data set in Figure 1.2. As we can see, the K-means algorithm converges to two clusters (red and blue). Old Faithful [24], is a hydrothermal geyser in Yellowstone National Park in the state of Wyoming, U.S.A., and is a popular tourist attraction. Its name stems from the supposed regularity of its eruptions. The data set comprises 272 observations, each of which represents a single eruption and contains two variables corresponding to the duration in minutes of the eruption, and the time until the next eruption, also in minutes.

Fuzzy C-Means (FCM)

Fuzzy clustering [20] methods allow the objects to belong to several clusters simultaneously, with different degrees of membership. In many situations, fuzzy clustering is more natural than hard clustering. Objects on the boundaries between several classes are not forced to fully belong to one of the classes, but rather are assigned membership degrees between 0 and 1 indicating their partial membership. The discrete nature of the hard partitioning also causes difficulties with algorithms based on analytic functionals (objective functions), since these functionals are not differentiable.

Generalization of the hard partition, described in the previous section, to the fuzzy case follows directly by allowing u_{ik} to attain real values in $[0, 1]$. A partition can be conveniently represented by the partition matrix $U = [u_{ik}]_{C\times N}$. The i-th row of this matrix contains values of the membership function u_i of the i-th cluster of $\mathbf{X}$. Conditions for a fuzzy partition matrix are given by [161]:

$$u_{ik} \in [0,1], \quad 1 \leq k \leq N, \quad 1 \leq i \leq C \tag{1.5}$$

$$\sum_{i=1}^{C} u_{ik} = 1, \quad 1 \leq k \leq N, \quad \text{and} \quad 0 < \sum_{k=1}^{N} u_{ik} < N, \quad 1 \leq i \leq C$$

The functional of the fuzzy C-means is formulated as follows:

$$J(\mathbf{X}; U, V) = \sum_{k=1}^{N}\sum_{i=1}^{C}(u_{ik})^m \|x_k - \mathbf{v}_i\|^2 \tag{1.6}$$

where m is the fuzzifier, which determines the fuzziness of the resulting clusters, it is generally chosen to be 2. $U = [u_{ik}]_{C\times N}$ is the fuzzy partition matrix of $\mathbf{X}$, and $V = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_C]$, $\mathbf{v}_i \in \mathbb{R}^D$, is the matrix of prototypes (cluster centres). Both U and V have to be determined in the learning process.

The minimization of the C-means objective function 1.6 represents a nonlinear optimization than can be solved by using an iterative procedure. The algorithm of fuzzy C-means looks similar to K-means, with an additional step in every iteration, which is updating the partition matrix. The algorithm of fuzzy C-means (called FCM) is presented in Algorithm 2.

Fuzziness Parameter: The weighting exponent m, called also *fuzzifier* is an important parameter. It significantly influences the fuzziness of the resulting partition. Whether the fuzzifier os adopted or not, the use of a fixed inner-product norm in the FCM

```
Algorithm 2: The Fuzzy $C$-Means Algorithm: FCM
Data: $\mathbf{X} = \{x_{kd}, \quad k = 1, \ldots, N, d = 1, \ldots, D\}$ where $D$ is the dimension of the feature
      space.
Result: Prototypes matrix $V$ and Partition matrix $U$
Initialization:
-Choose a value for $C$, $1 < C < N$
-Choose the weighting exponent $m > 1$
-Choose the termination criterion (threshold) $\epsilon > 0$
-Initialize the partition matrix $U$ randomly, such that $U$ verifies the conditions in 1.5
Learning: repeat
for $l = 1, 2, \ldots$ do
    -Step 1: Compute the cluster prototypes:
        $\mathbf{v}_i^{(l)} = \frac{\sum_{k=1}^{N} \left(u_{ik}^{(l-1)}\right)^m x_k}{\sum_{k=1}^{N} \left(u_{ik}^{(l-1))}\right)^m}, \quad 1 \leq i \leq C.$    (1.7)
    -Step 2: Compute the distances:
        $d^2(x_k, \mathbf{v}_i) = \|x_k - \mathbf{v}_i\|^2, \qquad 1 \leq i \leq C, \qquad 1 \leq k \leq N.$    (1.8)
    -Step 3: Update the partition matrix:
    for $1 \leq k \leq N$ do
        for $i = 1, 2, \ldots, C$ do
            $u_{ik}^{(l)} = \frac{1}{\sum_{j=1}^{C} \left(\frac{d(x_k, \mathbf{v}_i)}{d(x_k, \mathbf{v}_j)}\right)^{2/(m-1)}}$    (1.9)
Until $\|U^{(l)} - U^{(l-1)}\| < \epsilon$.
```

algorithm induces fuzzy clusters of a certain shape (geometry). As m approaches one from above, the partition becomes hard ($u_{ik} \in \{0, 1\}$) and $\mathbf{v}_i$ are ordinary means of the clusters. As $m \to \infty$, the partition becomes completely fuzzy ($u_{ik} = 1/C$) and the clusters means are all equal to the mean of $\mathbf{X}$. For instance, hyperspherical clusters are induced when the Euclidean norm is adopted, this means $m = 2$ is usually chosen.

Termination Criterion: The FCM algorithm stops iterating when the norm of the difference between U in two successive iterations is smaller than the termination parameter ϵ. The usual choice is $\epsilon = 10^{-3}$.

FCM Variants

An important class of FCM variants is related to algorithms that are designed to find clusters with different (possibly adaptive) geometries. This class includes the fuzzy C-varieties (FCV) algorithm [20], which is able to detect linear structures (such as lines and planes) in data [5], and the well known GustafsonKessel (GK) algorithm [74], which is able to find hyperellipsoidal fuzzy clusters with different spatial orientations. Other algorithms, such as the fuzzy maximum likelihood estimates (FMLE) proposed in [21] and the extended FCM and GK(E-FCM and E-GK)algorithms that are introduced in [109], are believed to be more suitable to handle data sets with uneven spatial distributions, namely data containing clusters with different volumes and densities [5, 62, 109].

Another important category of FCM relatives concerns algorithms that are more robust (less sensitive) to outliers and noise [48]. This category includes, e.g., L_1 norm-based FCM variants [85, 111] and possibilistic (rather than probabilistic) versions of FCM, such as the possibilistic C-Means (PCM) [121, 122], the fuzzy PCM (FPCM) [141], the possibilistic fuzzy c-means [142], and other related algorithms [13, 168, 180].

There are many other categories of FCM variants that have been proposed in the literature:

- Algorithms for handling objects with non-numerical (categorical/symbolic) attributes [54], whose main representatives are relational-data algorithms [120], such as the fuzzy analysis (FANNY) algorithm [108], the relational FCM (RFCM) [86], the non-Euclidean RFCM (NER-FCM) [83, 87], the fuzzy c-medoids (FCMdd) [120], and the fuzzy c-trimmed medoids (FCTMdd) [120].
- Algorithms for handling objects with missing value attributes (incomplete data), such as the partial distance strategy FCM (PDSFCM) and the optimal completion strategy FCM (OCSFCM), both proposed in [84].
- Algorithms conceived to scale up FCM in terms of running time and memory storage requirements, such as those based on some sort of fast numerical approximate solution or efficient exact algorithmic implementation of the FCM (see [36, 55, 92], respectively), tree-structured-data-based FCM approaches [92, 95], and subsampling-based FCM approaches [41, 140], among others, e.g., [102].
- Algorithms that are developed to incorporate partially supervised information provided by users, such as the proximity-based FCM [127] and other related knowledge-based algorithms [148, 149].
- Algorithms for handling distributed data, such as those described in [145, 151, 154]. This last category of FCM variants falls directly within the scope of this thesis, as will be discussed in Chapter 1.

The FCM algorithm is illustrated in Figure 1.3 using the Old Faithful data set we used in the previous section.

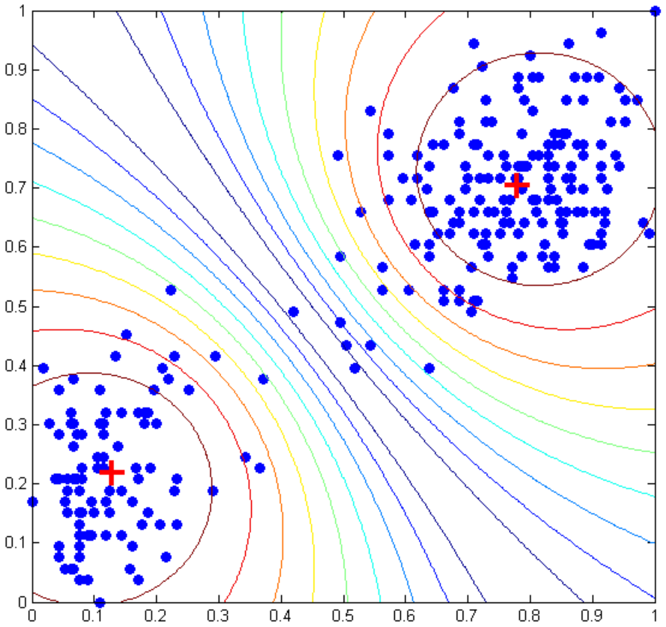


Figure 1.3: Illustration of the FCM algorithm on the Old Faithful data set, where $C = 2$. The red '+' signs are the cluster centres. Contours show the degree of membership of data to clusters.

1.2 Distributed Data Clustering

Nowadays, computing environments and technologies are more and more evolving towards a mobile, finely distributed, interacting, dynamic environment containing massive amounts of heterogeneous, spatially and temporally distributed data sources. Some typical examples of such ubiquitous computing environments are peer-to-peer systems, grid systems and wireless sensor networks, [51]. In these environments, data is distributed across several sources. So, communication between the different data sources and computing locations is necessary. In same time, data mining algorithms must be able to cope with privacy and security issues which prevent data from being gathered at a centralized repository. Traditional clustering algorithms perform its process of discovering groups over centralized databases. But recent applications require data sets distributed among several sites. A natural proposed solution is to concentrate all distributed data on a central site before applying traditional algorithms. This may not be feasible, there is a series of limitations which hinder the utilization of traditional clustering techniques on distributed databases. There are two distinct situations that demand the need for effecting cluster analysis in a distributed way. The first occurs when the volume of data to be analyzed is relatively great, which demand a considerable computational effort, which sometimes is even unfeasible, to accomplish this task. The best alternative, then, is splitting data, cluster them in a distributed way and unify the results. The second situation occurs when data is naturally distributed among several geographically distributed units and the cost associated to its centralization is very high.

As a motivating example [145], imagine a situation in which we have a collection of data sets existing at different organizations. These could be data describing customers of banking institutions, retail stores, medical organizations, etc. The data could include records of different individuals. They could also deal with the same individuals, but each data set may have different descriptors (features) reflecting the activities of the organization. The ultimate goal of each organization is to discover key relationships in its data set. These organizations also recognize that as there are other data sets, it would be advantageous to learn about the dependencies there occurring in order to reveal the overall picture of the global structure. We do not have direct access to other data, which prevents us from combining all data sets into a single database and carrying out clustering there. Access may be denied because of confidentiality requirements (e.g., medical records of patients cannot be shared and confidentiality of banking data has to be assured). There could also be some hesitation about the possibility of losing the identity of the data of the individual organization. We are more comfortable with revealing relationships in our own organization's data set. While appreciating the value of additional external sources of information, it is helpful to control how the findings there could affect the results from the data within the company. In some cases, there could be technical issues; processing (clustering) of a single huge data set may not be feasible or sufficiently informative.

Currently, a growing number of companies have strived to obtain a competitive advantage through participation in corporative organizations, as local productive arrangements, cooperatives networks and franchises. Insofar as these companies come together to overcome new challenges, their particular knowledge about the market needs to be shared among all of them. However, no company wants to share information about their customer and transact business with other companies and even competitors, because it is needed to maintain commercial confidentiality and due to local legislation matters.

So back to the traditional solution, gathering all distributed databases in a central unit and following it by algorithm application is strongly criticized, because in these cases, it is important to take into consideration some issus, namely:

- The possibility of existence of similar data with different names and formats, differences in data structures, and conflicts between one and another database [183].

- The unification of all of the registers in a single database may take to the loss of meaningful information, once that statistically interesting values in a local context may be ignored when gathered to other ones in a larger volume.
- Integration of several database in a single location is not suggested when it is composed of very large databases. If a great organization has large disperse databases and needs to gather all the data in order to apply on them data mining algorithms, this process may demand great data transference, which may be slow and costly [59].
- Any change that may occur in distributed data, for instance inclusion of new information or alteration of those already existing will have to be updated along with the central database. This requires a very complex data updating strategy, with overload of information transference in the system.
- And the most important limitation is that in some domains such as medical and business areas whereas distributed databases occurs, transferring raw data sets among parties can be insecure because confidential information can be obtained, putting in *risk privacy preserving and security requirements.*

Due to all of these problems related to database integration, research for algorithms that perform data mining in a distributed way is not recent. Several researches about algorithms to effectuate distributed data mining were presented [52], [2], [154].

Hence, a large number of studies in this research area, called privacy preserving distributed data mining (DDM), where security and confidentiality of data must be maintained throughout the process, have been prompted by the need of sharing information about a particular business segment among several companies involved in this process, respecting the privacy of its customers [46]. These studies seek to be able to process clustering securely in a way that has motivated the development of algorithms to analyze each database separately and to combine the partial results to obtain a final result, [68], [119], [98]. A very rich and updated bibliography about the matter is available in [104], [106].

Another example of the need of distributed algorithms is the NASA Earth Observing System (EOS), a data collector for a number of satellites, holds 1450 data sets that are stored, managed, and distributed by the different EOS Data and Information System (EOSDIS) sites that are geographically located all over the USA. A pair of Terra spacecraft and Landsat 7 alone produces about 350 GB of EOSDIS data per day. An online mining system for EOS data streams may not scale if we use a centralized data mining architecture. Mining the distributed EOS repositories and associating the information with other existing environmental databases may benefit from DDM, [143].

To point out the mismatch between the architecture of centralizing data mining systems and distributed data mining systems look to Figure 1.4 and Figure 1.5, [143]. Figure 1.4 presents a schematic diagram of the traditional data warehouse-based architecture for data mining. This model of data mining works by regularly uploading mission critical data in the warehouse for subsequent centralized data mining application. This centralized approach is fundamentally inappropriate for most of the distributed and ubiquitous data mining applications. As shown in Figure 1.5, the objective of DDM is to perform the data mining operation based on the type and availability of the distributed resources.

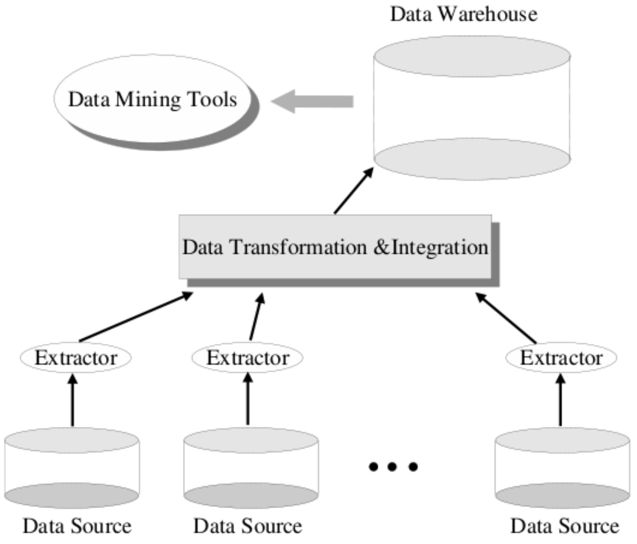


FIGURE 1.4: A data warehouse architecture

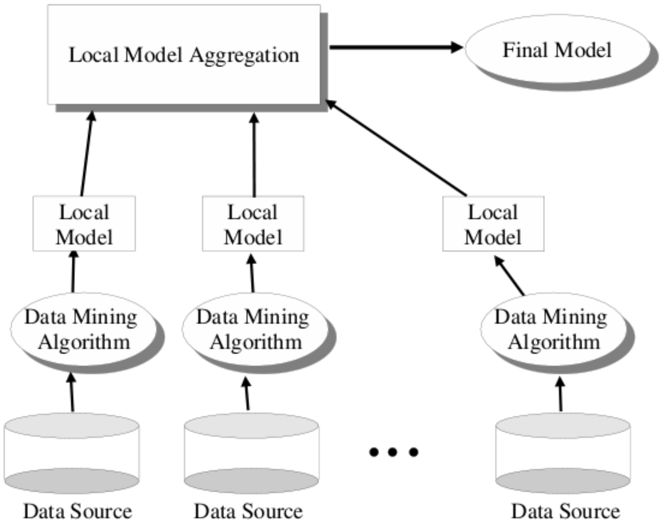


FIGURE 1.5: Distributed Data Mining Framerwork

In literature, many models of interaction in DDM were introduced [163], [112], [45], [134], [46]. But we are mostly interested in Collaborative Clustering [146]. The difference between DDM and Collaborative Clustering is in the level of interaction: the Collaborative Clustering is positioned at the more active side where the structures are revealed in a more collective manner; while DDM seeks to build a consensus clustering focused at the stage of constructing clusters when there is no activement of data. In Collaborative Clustering, we exchange information between data sets to improve clustering results before building the consensus. A more advanced use of Collaborative Clustering is to control this information exchange by estimating a coefficient of collaboration precising the confidence between data sets.

Collaborative clustering was first investigated by Pedrycz [145–147, 150], using a fuzzy c-means algorithm (FCM) [20]. The fundamental concept of collaboration is : “the clustering algorithms operate locally (namely, on individual data sets) but collaborate by exchanging information about their findings” *Pedrycz*.

In next section we present the collaborative fuzzy clustering based on FCM, introduced by [145]. We detail the algorithms in its two distinct situations, *horizontal* collaboration and *vertical* collaboration.

1.3 Collaborative Fuzzy Clustering

In 2002, Pedrycz [145] introduced a novel clustering algorithm, called Collaborative Fuzzy Clustering, which intended to reveal the overall structure of distributed data (i.e. data residing at different repositories) but, at the same time, complying with the restrictions preventing data sharing. It can be stated that this approach exhibits significant differences with other existing techniques under the umbrella of distributed clustering [151].

In brief, the problem of collaborative clustering can be defined as follows:

> Given a finite number of disjoint data sites, develop a scheme of collective development and reconciliation of a fundamental cluster structure across the sites that it is based upon exchange and communication of local findings where the communication needs to be realized at some level of information *granularity*. The development of the structures at the local level exploits the communicated findings in an *active* manner through minimization of the corresponding objective function augmented by the structural findings developed outside the individual data site. We also allow for retention of key individual (specic) findings that are essential (unique) for the corresponding data site.

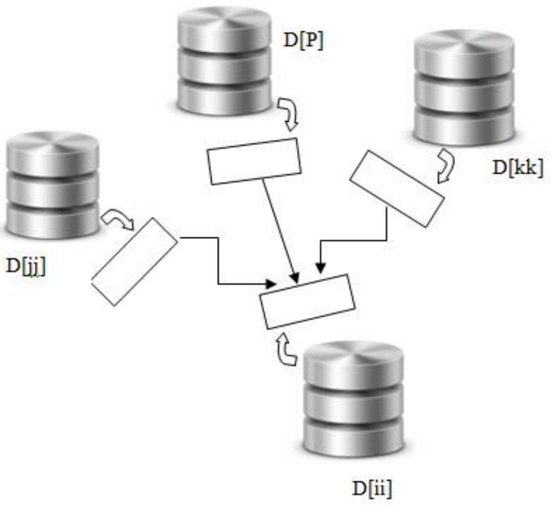


FIGURE 1.6: The essence of collaborative clustering in which we aim at striking a sound balance between local ndings (produced at the level of locally available data) and the ndings coming from other data sites building some global characterization of data. Shown are only communication links between data site $D[ii]$ and all other data sites.

Generally speaking, two types of collaborative clustering are envisioned, the horizontal mode and the vertical mode. The vertical mode assumes that each site holds information on different objects described by the same variables, i.e. in the same feature space. The horizontal mode, on the other side, assumes that each location holds information on the same set of objects but described in different feature spaces. The horizontal mode is more complicated since prototypes do not have the same dimension, so defining a distance between them is impossible.

Suppose that we have P data sets. The general collaborative clustering scheme consists of two phases:

Phase I: Generating the clusters without collaboration, using a local FCM algorithm ([53], [20]) on each data set. Although any objective-function based clustering algorithm can be used. Obviously, the number of clusters has to be the same for all data sets. FCM identifies c cluster centres and assigns each record k (customer, pattern..) with a specific membership degree u_{ik} to cluster i. The membership degrees u_{ik} for $i = 1, \dots, c$ are constrained to sum to 1. The FCM analysis minimizes the following objective function $Q[ii]$ (Eq. 1.10) where d_{ik} denotes the distance between case k and cluster center i; $[ii]$ refers to the data set where the local cluster analysis is performed.

$$Q[ii] = \sum_{k=1}^{N} \sum_{i=1}^{c} u_{ik}^2[ii] d_{ik}^2[ii] \tag{1.10}$$

$$ii = 1, 2, ..., P.$$

Phase II: After the local phase each site gets an initial set of cluster centres and $N \times c$ partition matrix containing the membership degrees of each case k to each cluster i. Here comes the phase II, a collaboration between the clusters is performed.

Each site will exchange its partition matrix or prototypes with the other sites. Because sites only exchange these findings, no private information is exchanged and consequently no privacy constraints are violated. Once the sites receive the exchanging parameters, the collaborative FCM can be applied, which minimizes a modified objective function (Eq. 1.11 and Eq. 1.28).

The collaboration between the sites depend on collaboration links $\alpha[ii, kk]$ which describe the intensity of collaboration between site $[ii]$ and site $[kk]$. Usually $\alpha[ii, kk]$ is non-negative. So, more this value $\alpha[ii, kk]$ is higher more the cooperation between sites is stronger. The set of all collaboration links is called the collaboration matrix.

Schematically, we portray the essence of the collaborative clustering as presented in Fig. 1.6, which stresses an act of balance between collaborative activities occurring between the data sites and reecting global and common characteristics of all data and the crucial ndings implied by the locally available data.

1.3.1 Horizontal Collaborative Clustering

In horizontal clustering we deal with the same patterns and different feature spaces. The communication platform is based on through the partition matrix (see Eq. 1.11). As we have the same objects, this type of collaboration makes sense. The confidentiality of data has not been breached: we do not operate on individual patterns but on the resulting information granules (fuzzy relations, that is, partition matrices). As this number is far lower than the number of data, the low granularity of these constructs moves us far from the original data. The schematic illustration of this mode of clustering is presented is Figure 1.7.

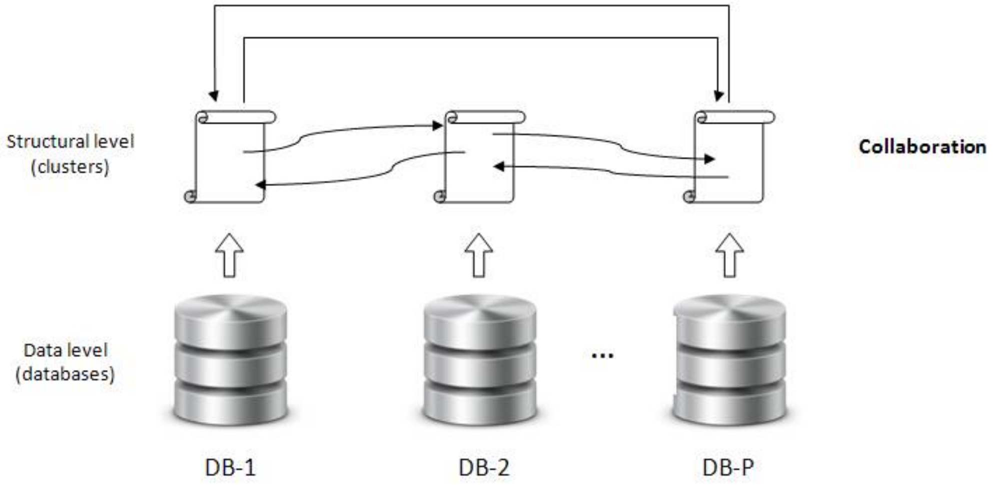


Figure 1.7: A general scheme of Horizontal Clustering.

To accommodate the collaboration mechanism in the optimization process, the objective function is expanded into the form:

$$Q^*[ii] = \sum_{k=1}^{N}\sum_{i=1}^{c} u_{ik}^2[ii]d_{ik}^2[ii] + \sum_{\substack{jj=1 \\ jj\neq ii}}^{P} \alpha[ii, jj] \sum_{k=1}^{N}\sum_{i=1}^{c}(u_{ik}[ii] - u_{ik}[jj])^2 d_{ik}^2[ii] \tag{1.11}$$

$ii = 1, 2, \ldots, P$. The role of the second term in the above expression is to have the ii-th data set become fully cognizant of what's going on in the remaining subsets. So it makes the clustering based on the ii-th subset "aware" of other partitions. It is obvious that if the structures in data sets are similar, then the differences between the partition matrices tend to be lower, and the resulting structures start becoming more similar.

In the optimization task, we determine the partition matrix $U[ii]$ and the prototypes $\mathbf{v}_1[ii], \ldots, \mathbf{v}_c[ii]$, separately for each of the collaborating subsets of patterns. The partition matrix is required to satisfy standard requirements of membership grades summing to 1 for each pattern and the membership grades contained in the unit interval. So, collaborative clustering converts into the following family of P optimization problems with the membership constraints

$$\min_{U\in U,\mathbf{v}_1,\ldots,\mathbf{v}_c} Q \quad \text{subject to} \quad U \in \mathbf{U}$$

where $\mathbf{U}$ is a family of all fuzzy partition matrices.

Calculation of the partition matrix

To determine the partition matrix, a technique of Lagrange multipliers is exploited so that the constraint occurring in the problem becomes merged as a part of constraint-free optimization. This lead to the new objective function $V[ii]$:

$$\begin{aligned} V[ii] = & \sum_{i=1}^{c} u_{ik}^2[ii] d_{ik}^2[ii] + \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \alpha[ii,jj] \sum_{i=1}^{c} (u_{ik}[ii] - u_{ik}[jj])^2 d_{ik}^2[ii] \\ & - \lambda\Bigg(\sum_{i=1}^{c} u_{ik}[ii] - 1 \Bigg) \end{aligned} \tag{1.12}$$

for $k = 1, \ldots, N$, where λ denotes a Lagrange multiplier. The necessary conditions leading to the local minimum of $V[ii]$ are

$$\frac{\partial V[ii]}{\partial u_{st}[ii]} = 0 \quad \text{and} \quad \frac{\partial V[ii]}{\partial \lambda} = 0 \tag{1.13}$$

$s = 1, 2, \ldots, c$, $t = 1, 2, \ldots, N$. The derivative computed with respect to the partition matrix is

$$\frac{\partial V}{\partial u_{st}} = 2u_{st}[ii] d_{st}^2[ii] + 2 \sum_{jj \neq ii} \alpha[ii,jj] (u_{st}[ii] - u_{st}[jj]) d_{st}^2[ii] - \lambda = 0 \tag{1.14}$$

Therefore,

$$u_{st}[ii] = \frac{\lambda + 2d_{st}^2[ii] \overbrace{\sum_{jj \neq ii} \alpha[ii,jj] u_{st}[jj]}^{\varphi_{st}[ii]}}{2\Bigg(d_{st}^2[ii] + d_{st}^2[ii] \underbrace{\sum_{jj \neq ii} \alpha[ii,jj]}_{\psi[ii]} \Bigg)} \tag{1.15}$$

Let's introduce the following notation:

$$\varphi_{st}[ii] = \sum_{jj \neq ii} \alpha[ii,jj] u_{st}[jj] \tag{1.16}$$

$$\psi[ii] = \sum_{jj \neq ii} \alpha[ii,jj] \tag{1.17}$$

In light of the constraint imposed on the membership values $\sum_{j=1}^{c} u_{jk}[ii] = 1$, the use of the above expression yields the result

$$\sum_{j=1}^{c} \frac{\lambda + 2d_{jk}^2[ii]\varphi_{jk}[ii]}{2d_{st}^2[ii](1+\psi[ii])} = 1 \tag{1.18}$$

Next, the Lagrange multiplier is computed in the form

$$\lambda = 2\frac{1 - \frac{1}{1+\psi[ii]}\sum_{j=1}^{c}\varphi_{jk}[ii]}{\sum_{j=1}^{c}\frac{1}{d_{jk}^2[ii]}}\big(1+\psi[ii]\big) \tag{1.19}$$

Plugging this multiplier into the formula for the partition matrix produces the final expression:

$$u_{st}[ii] = \frac{\varphi_{st}[ii]}{1+\psi[ii]} + \frac{1 - \frac{1}{1+\psi[ii]}\sum_{j=1}^{c}\varphi_{jt}[ii]}{\sum_{j=1}^{c}\frac{d_{st}^2[ii]}{d_{jt}^2[ii]}} \tag{1.20}$$

Calculation of the prototypes

In the calculations of the prototypes, we confine ourselves to the Euclidean distance between the patterns and the prototypes. The necessary condition for the minimum of the objective function is of the form $\bigtriangledown_{\mathbf{v}[ii]}Q[ii] = 0$. Rewriting $Q[ii]$ in an explicit manner to emphasize the character of the distance function gives:

$$\begin{aligned} Q[ii] =& \sum_{k=1}^{N}\sum_{i=1}^{c} u_{ik}^2[ii]\sum_{j=1}^{N}(x_{kj} - \mathbf{v}_{ij}[ii])^2 + \sum_{\substack{jj=1 \\ jj\neq ii}}^{P} \alpha[ii,jj] \\ &\times \sum_{k=1}^{N}\sum_{i=1}^{c}(u_{ik}[ii] - u_{ik}[jj])^2 \sum_{j=1}^{N}(x_{kj} - \mathbf{v}_{ij}[ii])^2 \end{aligned} \tag{1.21}$$

The patterns in this expression come from the iith data set. Computing the derivative of $Q[ii]$ with respect to $\mathbf{v}_{st}[ii]$ ($s = 1, 2, \ldots, c$, $t = 1, 2, \ldots, N$) and setting it to 0, we obtain

$$\frac{\partial Q[ii]}{\partial \mathbf{v}_{st}[ii]} = -2\sum_{k=1}^{N} u_{st}^2[ii](x_{kt} - \mathbf{v}_{st}[ii]) \\ - 2\sum_{k=1}^{N}\sum_{jj \neq ii} \alpha[ii,jj](u_{sk}[ii] - u_{sk}[jj])^2(x_{kt} - \mathbf{v}_{st}[ii]) = 0 \tag{1.22}$$

which leads to expression of calculation of the prototypes:

$$\mathbf{v}_{st}[ii] = \frac{A_{st}[ii] + C_{st}[ii]}{B_s[ii] + D_s[ii]} \tag{1.23}$$

$$s = 1, 2, \ldots, c,\ \ t = 1, 2, \ldots, N,\ \ ii = 1, 2, \ldots, P$$

where

$$A_{st}[ii] = \sum_{k=1}^{N} u_{sk}^2[ii]x_{kt} \tag{1.24}$$

$$B_s[ii] = \sum_{k=1}^{N} u_{sk}^2[ii] \tag{1.25}$$

$$C_{st}[ii] = \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \alpha[ii,jj] \sum_{k=1}^{N} (u_{sk}[ii] - u_{sk}[jj])^2 x_{kt} \tag{1.26}$$

$$D_s[ii] = \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \alpha[ii,jj] \sum_{k=1}^{N} (u_{sk}[ii] - u_{sk}[jj])^2 \tag{1.27}$$

1.3.2 Vertical Collaborative Clustering

The concept of vertical collaborative clustering is the situation when we deal with different data sets where all the patterns are described in the same feature space [145]. In this case we cannot establish communication at the level of the partition matrices, but we can use the prototypes of this data sets since they are defined in the same feature space. Figure 1.8 shows the general scheme of this case. This type of collaboration is interesting in business applications. Lets take an example of two (or many) supermarkets trying to study their customer's behavior. They have the same variables but they don't have the same customers, they want to see if rich customers of the first behave the same of rich customers of the second supermarket.

Another interesting application of vertical collaborative clustering occurs when dealing with huge data sets. Instead of clustering them in a single pass, we split them into individual data sets, cluster each of them separately, and reconcile the results through the collaborative exchange of prototypes.

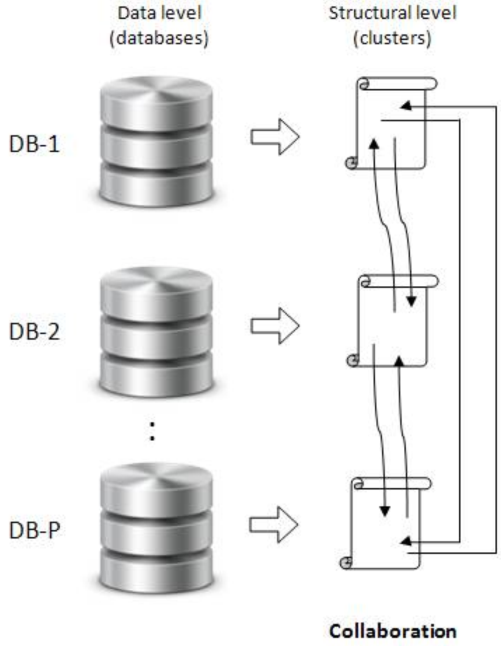


FIGURE 1.8: A general scheme of Vertical Clustering.

The proposed objective function governing a search for structure in the ii-th data set is the following:

$$Q[ii] = \sum_{k=1}^{N[ii]} \sum_{i=1}^{c} u_{ik}^2[ii] d_{ik}^2[ii] + \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \beta[ii,jj] \sum_{i=1}^{c} \sum_{k=1}^{N[ii]} u_{ik}^2[ii] \|\mathbf{v}_i[ii] - \mathbf{v}_i[jj]\|^2 \tag{1.28}$$

where $\beta[ii,jj]$ is a collaboration coefficient supporting an impact of jj-th data set and affecting the structure to be determined in the ii-th data set. The number of patterns in the ii-th data set is denoted by $N[ii]$ (since it is not the same for all data sets). Equation 1.28 is interpreted as follows: the first term is the objective function used to search for the structure of the ii-th data set, and the second term articulates the differences between the prototypes which have to be made smaller through the refinement of the partition matrix.

The optimization of $Q[ii]$ involves the determination of the partition matrix $U[ii]$ and the prototypes $\mathbf{v}_i[ii]$. As before, the problem is solved for each data set separately and allow the results to interact, forming a collaboration between the sets.

Calculation of the partition matrix

The minimization of $Q[ii]$ with respect to the partition matrix requires the use of Lagrange multipliers because of the existence of the standard constraints imposed on the partition matrix ($\sum_{i=1}^{c} u_{ik}[ii] = 1$). We obtain the following

$$\begin{aligned} V[ii] = & \sum_{i=1}^{c} u_{it}^2[ii] d_{it}^2[ii] + \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \beta[ii,jj] \sum_{i=1}^{c} u_{it}^2[ii] \|\mathbf{v}_i[ii] - \mathbf{v}_i[jj]\|^2 \\ & - \lambda \Bigg(\sum_{i=1}^{c} u_{it}[ii] - 1 \Bigg) \end{aligned} \tag{1.29}$$

where $t = 1, 2, \ldots, N[ii]$. Taking the derivative of $V[ii]$ with respect to $u_{st}[ii]$ and making it 0, we obtain

$$\frac{\partial V}{\partial u_{st}} = 2u_{st}[ii]d_{st}^2[ii] + 2 \sum_{jj \neq ii} \beta[ii,jj] u_{st}[ii] \|\mathbf{v}_s[ii] - \mathbf{v}_s[jj]\|^2 - \lambda = 0 \tag{1.30}$$

Introducing the following notation:

$$D_{ii,jj,s} = \|\mathbf{v}_s[ii] - \mathbf{v}_s[jj]\|^2 \tag{1.31}$$

$$\psi_s[ii] = \sum_{jj \neq ii}^{P} \beta[ii,jj] D_{ii,jj,s} \tag{1.32}$$

The partition matrix is

$$u_{st}[ii] = \left(\sum_{j=1}^{c} \frac{d_{st}^2[ii] + \psi_s[ii]}{d_{jt}^2[ii] + \psi_j[ii]} \right)^{-1} \tag{1.33}$$

Calculation of the prototypes

For the prototypes, we complete calculations of the gradient of Q with respect to the coordinates of the prototypes $\mathbf{v}[ii]$ and then solve the following equations:

$$\frac{\partial Q[ii]}{\partial \mathbf{v}_{st}[ii]} = 0, \quad s = 1, 2, \ldots, c, \; t = 1, 2, \ldots, N[ii]. \tag{1.34}$$

We obtain

$$\begin{aligned}\frac{\partial Q[ii]}{\partial \mathbf{v}_{st}[ii]} &= 2\sum_{k=1}^{N[ii]} u_{st}^2[ii](x_{kt} - \mathbf{v}_{st}[ii]) \\ &\quad +2\sum_{jj \neq ii} \beta[ii,jj] \sum_{k=1}^{N[ii]} u_{st}^2[ii](\mathbf{v}_{st}[ii] - \mathbf{v}_{st}[jj]) = 0\end{aligned} \tag{1.35}$$

This leads to the calculation of the prototypes

$$\mathbf{v}_{st}[ii] = \frac{\sum_{jj \neq ii} \beta[ii,jj] \sum_{k=1}^{N[ii]} u_{sk}^2[ii]\mathbf{v}_{st}[jj] - 2\sum_{k=1}^{N[ii]} u_{sk}^2[ii]x_{kt}}{\sum_{jj \neq ii} \beta[ii,jj] \sum_{k=1}^{N[ii]} u_{sk}^2[ii] - \sum_{k=1}^{N[ii]} u_{sk}^2[ii]} \tag{1.36}$$

We present the algorithm of collaborative clustering in Algorithm 3, for both horizontal and vertical approaches.

```
Algorithm 3: The Collaborative Clustering scheme: Co-FCM
Data: subsets of patterns $\mathbf{X}[1], \mathbf{X}[2], \ldots, \mathbf{X}[P]$.
Result: Prototypes and Partition matrix.
Initialization: Select the distance function, number of clusters $c$, termination
criterion, and collaboration matrix $\alpha[ii,jj]$.
Phase I
for each data set $[ii]$ do
    compute prototypes $\{\mathbf{v}_i[ii]\}$, $i = 1, 2, \ldots, c$ and partition matrix $U[ii]$ for all subsets
    of patterns.
    until a termination criterion has been satisfied
Phase II
for each data set $[ii]$ do
    For given matrix of collaborative link $\alpha[ii,jj]$, compute prototypes and partition
    matrices. Using (1.20) and (1.23) for horizontal clustering. Using (1.33) and (1.36)
    for vertical clustering.
    until a termination criterion has been satisfied
```

1.3.3 Hybrid Collaborative Clustering

There could be also, situations when both collaborative clustering approaches: vertical and horizontal are used in the same time. For example, when patterns from various sources give rise to common subsets of data as well as being positioned in the same feature space. This leads to a mode called 'hybrid collaborative clustering'. This leads to the grid mode of clustering, with examples of collaboration shown in Figure 1.9.

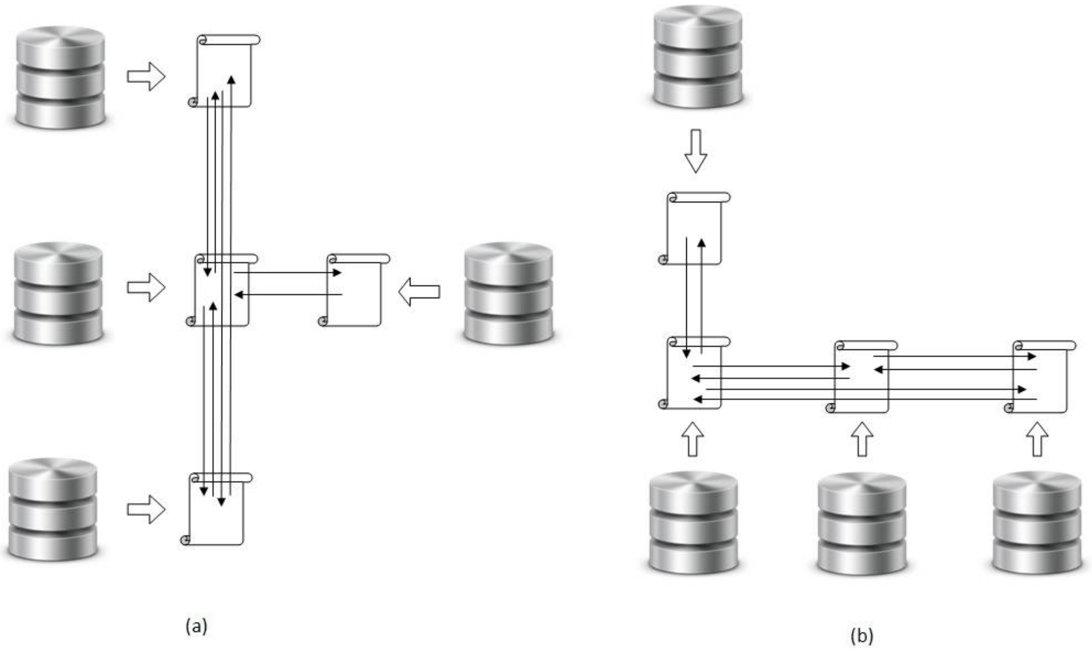


Figure 1.9: Illustrative examples of grid-based clustering: (a) data structure with a dominant component of vertical clustering; (b) data structure with a dominant component of horizontal clustering and some linkages of vertical clustering.

In this case, the objective function formulated for the *ii*-th pattern as a subject of minimization is an aggregation (sum) of the components used in the previous modes of collaborative clustering. In general, we have

$$
\begin{aligned}
Q[ii] =& \sum_{k=1}^{N}\sum_{i=1}^{c} u_{ik}^2[ii]d_{ik}^2[ii] + \sum_{\substack{jj=1\\ jj\neq ii\\ \mathbf{D}_1}}^{P} \alpha[ii,jj]\sum_{k=1}^{N}\sum_{i=1}^{c}(u_{ik}[ii]-u_{ik}[jj])^2 d_{ik}^2[ii] \\
&+ \sum_{\substack{jj=1\\ jj\neq ii\\ \mathbf{D}_2}}^{P} \beta[ii,jj]\sum_{i=1}^{c}\sum_{k=1}^{N[ii]} u_{ik}^2[ii]\|\mathbf{v}_i[ii]-\mathbf{v}_i[jj]\|^2
\end{aligned}
\tag{1.37}
$$

using the same notation used earlier. Note that the summation points at the corresponding data sets (operating in either mode of collaboration), that is, $\mathbf{D}_1$ (in Eq. 1.37) involves all data sets that operate in the horizontal mode of clustering, whereas $\mathbf{D}_2$ concerns those using vertical collaboration.

We are not going to discuss it in this thesis, it may be a perspective for the future work.

1.4 Quantifying the Collaboration Effect

There are two levels of assessing a collaboration effect occurring between the clusters, namely *the level of data* and *the level of information granules*.

The level of data

The level of data involves a comparison carried out at the level of numeric representatives of the clustering, that is the prototypes. The impact of the collaboration is then expressed in the changes of the prototypes occurring as a result of the collaboration.

The level of information granules

At this level (partition and fuzzy sets), the effect of collaboration is expressed in two ways as show schematically in Figure 1.10 where the collaboration involves two data sets (viz. $P = 2$) indicated by ***ii*** and ***kk***. Similarly, by ***ii*-ref** and ***kk*-ref** we denote the results resulting from the clustering carried out without any collaboration.

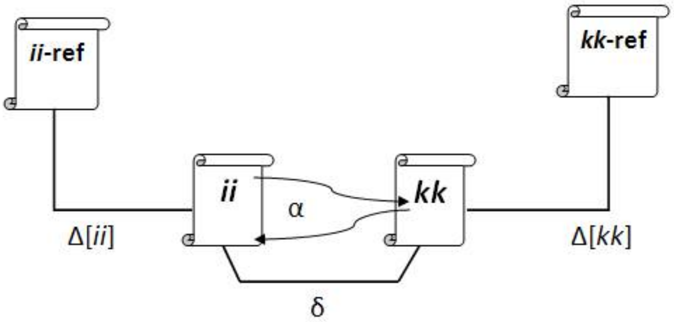


FIGURE 1.10: Quantification of the collaboration at the level of information granules (results).

The first measure compares the membership degrees of each data pattern k to each clusters i before ($u_{ik}[ii\text{-ref}]$) and after ($u_{ik}[ii]$) the collaboration. The overall impact on the partition matrices in a specific data site ii is expressed as

$$\Delta[ii] = \frac{1}{N}\sum_{k=1}^{N} u_{ik}[ii] - u_{ik}[ii\text{-ref}] \tag{1.38}$$

The average collaboration effect on the membership degrees for all data sites can then be computed as

$$\Delta = \frac{1}{P}\sum_{ii=1}^{P} \Delta[ii] \tag{1.39}$$

A significant variation in the membership degrees for each data site before and after the collaboration (high value of Δ) translates into a stronger collaborative impact.

The second measure expresses how close two partition matrices are as a result of the collaboration. The pertinent measure reads as an average distance between the partition matrices $U[ii] = \{u_{ik}[ii]\}$ and $U[kk] = \{u_{ik}[kk]\}$, that is

$$\delta[ii, kk] = \frac{1}{N \times c}\sum_{k=1}^{N}\sum_{i=1}^{c} u_{ik}[ii] - u_{ik}[kk] \tag{1.40}$$

Evidently, the stronger the collaboration (higher values of the corresponding α), the lower the values of δ. In this sense, this index helps us translate the collaboration parameters α into the effective changes in the membership grades (that are the apparent final result of such interaction).

1.5 Estimation of the Collaboration Coefficients

The collaborative clustering is aimed at forming a consensus and each external source of information should be used to refine the already developed structure within the given data set. In the previous section, two measures of the collaboration effect are presented. Note that once the level of collaboration (the coefficients α) increases, the structures within data sets start to exhibit smaller differences. The level of collaboration can be adjusted, allowing for a certain maximal value of changes of the membership grades. This is in case we consider (as presented in this chapter) that the level of collaboration is fixed by the user.

The good news would be how the collaboration can improve ALL the results, i.e., how every clustering helps to improve the overall clustering, accepting good clustering and rejecting bad clustering. So, an interesting task would be to estimate the level of collaboration, or to learn it automatically during the collaboration phase. By doing this, each coefficient will tell how much a site trust another site, this task forces a sort of discussion between the data sets before building the consensus, each site estimates the confidence it gives for all other data sets. After doing this, we can move forward to the construction of the consensus.

Some papers discussed this subject and proposed different methods to do the task [73], [65], [56], [51].

In our paper [73], we presented a collaborative clustering scheme based on SOM as a local phase of clustering, we estimated the coefficients of collaboration by using a gradient [4] algorithm. More details are presented in Chapter 2.

In [56], the coefficients of collaboration are estimated using a Particle Swarm Optimization (PSO [110]) driven algorithm. Later in his work, [51] used a multi-PSO to do the task.

In [181], a method for calculating the coefficients of collaboration automatically is presented, basing on a similarity measure of partition matrices. This method, however, works only for horizontal collaboration.

1.6 Summary

A cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters. The process of grouping a set of physical or abstract objects into classes of similar objects is called clustering. Cluster analysis has extensive applications, including business intelligence, image pattern recognition, Web search, biology, and security. Cluster analysis can be used as a standalone data mining tool to gain insight into the data distribution, or as a preprocessing step for other data mining algorithms operating on the detected clusters. Clustering is a dynamic field of research in data mining. It is related to unsupervised learning in machine learning.

Many clustering algorithms have been developed. These can be categorized from several orthogonal aspects such as those regarding partitioning criteria, separation of clusters, similarity measures used, and clustering space. This chapter discusses two fundamental clustering partitioning methods.

So far, clustering methods performs on a single data set, but recent applications require data sets distributed among several sites. So, communication between the different data sets is necessary, while respecting the privacy of every site, i.e. sharing data between sites is not allowed.

Collaborative clustering is useful to achieve interaction between different sources of information for the purpose of revealing underlying structures and regularities within data sets. It can be treated as a process of consensus building where we attempt to reveal a structure that is common across all sets of data. The introduced models of horizontal and vertical clustering achieve an active form of collaboration. Vertical, horizontal, and hybrid clustering are essential mechanisms of communication between the clusters. The level of granularity at which the communication takes place is a useful and practical way of retaining the features of data security and confidentiality.

An interesting task to do is to estimate the level of collaboration (coefficients of collaboration) to evaluate the trust between the different data sets. It is an important task to do before moving forward to the built of consensus.

An inconvenience of the algorithm proposed in this chapter is that FCM does not provide any type of visualization once the dimension of the feature space is higher than 3. In this thesis, we present a formalism of collaborative clustering using methods of topological clustering, giving advantage of visualization. We started by applying the collaborative clustering using Self-Organizing Maps (SOM) [115] as local step of clustering, the proposed formalism is presented in next chapter. In chapter 3, we applied the collaborative clustering scheme using a generative model, which is the Generative Topographic Mapping (GTM) [25], instead of SOM. GTM was proposed as a probabilistic counterpart of SOM. But GTM suffers from some limitations, especially the risk of over-fitting. An elegant solution to this limitation is to apply a Variational Bayesian technique to GTM, it was presented in [139] and called VBGTM. In chapter 4, we present a quick explanation of the Variational Bayesian techniques and show a collaborative clustering scheme using VBGTM. In addition, we propose a method to estimate the collaboration level during the learning process.

Chapter 2

Collaborative Clustering using Self-Organizing Maps

2.1 Introduction

A large variety of clustering methods has been developed. Several of these methods are based on very simple fundamentals, yet very effective idea, namely describing the data under consideration by a set of prototypes, which capture characteristics of the data distribution (like location, size, and shape), and to classify or divide the data set based on the similarity of the data points to these prototypes. The approaches relying on this idea differ mainly in the way in which prototypes are described and how they are updated during the model construction step. One of the most known methods is the Self-Organizing Maps (SOM), introduced by Kohonen [115]. It has been widely used for unsupervised classification and visualization of multidimensional data sets. In this chapter, we present our algorithm of collaborative clustering for the horizontal and vertical cases of collaboration, using SOMs as local step for clustering. The chapter is organized as follows: we present an introduction of SOM algorithm, then the principle of collaborative clustering using SOMs, we enrich our algorithm with a step in the collaboration phase, which is estimating the coefficients of collaboration during the process of collaboration. Finally we present our experimental results.

2.2 Self-Organizing Maps (SOM)

The self-organizing maps introduced by Kohonen ([114], [115], [116]) is a popular nonlinear technique for unsupervised classification and has been widely used for dimensionality reduction and data visualization, with a very low computational cost. There is a wide variety of algorithms for topological maps derived from the original model proposed firstly by Kohonen ([25], [182], [172], [103]). These models are different from each other, but share the same idea to present the large data in a simple geometric relationship on a reduced topology.

The model can be seen as a K-means algorithm with *topological* constraints, usually with a better overall clustering performance [40], it consists in the attempting of clustering a learning set $A = \{x^{(i)} \in \mathbb{R}^n, i = 1, ..., N\}$ where $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, ..., x_j^{(i)}, ..., x_n^{(i)})$. This classical model consists in a discrete set C of cells (neurons) called map. This map has a discrete topology defined by undirected graph; usually it is a regular grid in two dimensions. The influence notion of a cell k on a cell l, which depends on their proximity, is presented by a kernel function $\mathcal{K}$ ($\mathcal{K} \geq 0 \;\; and \;\; lim_{|x| \to \infty} \mathcal{K}(x) = 0$). The mutual influence between two units k and l is defined by the function $\mathcal{K}_{k,l}(.)$:

$$\mathcal{K}_{ij} = \frac{1}{\lambda(t)} \exp\left(-\frac{d_1^2(i,j)}{\lambda^2(t)}\right) \tag{2.1}$$

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

$$\lambda(t) = \lambda_i (\frac{\lambda_f}{\lambda_i})^{\overline{t_{max}}^{\,t}} \tag{2.2}$$

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

$$d_1(r, s) = |i - k| + |j - m| \tag{2.3}$$

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

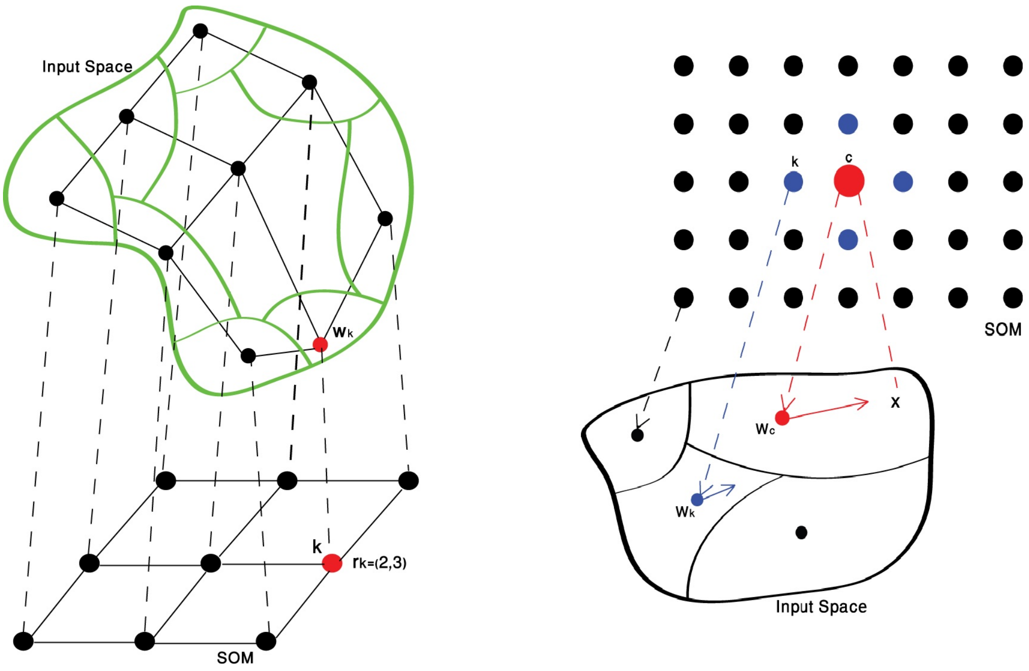


FIGURE 2.1: Architecture of SOM: Mapping the data space to the self-organizing map (left). SOM prototypes update (right). [75]

and prototypes of the map, weighted by the neighborhood. A gradient algorithm [4] can be used for this purpose. The criterion to minimize in this case is:

$$R(\chi, \mathcal{W}) = \sum_{i=1}^{N} \sum_{j=1}^{C} \mathcal{K}_{j,\chi(\mathbf{x}^{(i)})} \|\mathbf{x}^{(i)} - \mathbf{w}^{(j)}\|^2 \tag{2.4}$$

where χ assigns each pattern (observation) $x^{(i)}$ to a single cell of the SOM.

At the end of the learning, the SOM determines a data partition in C groups associated with each cell k of the map. Each group or cell is associated with a reference vector $w^{(k)} \in \mathbb{R}^n$, which will be the representative, the "local mean" or the prototype of the observation's set associated with this cell. It was proven in [34] that the generated SOM is optimally topology-preserving in a very general sense.

2.2.1 Understanding SOM visualization

A SOM may be the most compact way to represent a data distribution. Because SOMs represent complex data in an intuitive two-dimensional perceptional space, data dependencies can be understood easily if one is familiar with the map visualization. The following example provides an intuitive explanation of the basics of SOM visualization.

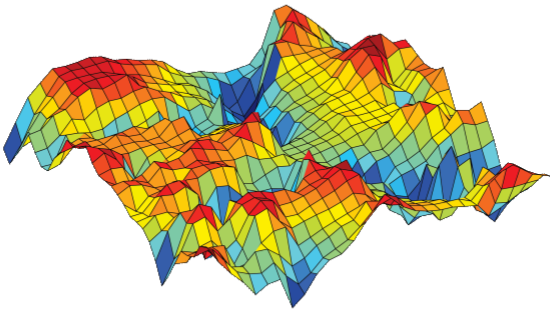


Figure 2.2: Example of SOM visualization

Imagine 1000 people on a football field. We define a number of attributes (e.g. gender, age, family status, income) and ask the people on the field to move closer to other people who are most similar to them according to all these attributes. After a while, everyone on the field is surrounded by those people that share similar attribute values. This configuration is an example of a two-dimensional representation of multi-dimensional data points.

Now imagine that, looking over the crowd, you ask everyone to raise a colored flag according to their age (blue for <20, green for 20 to 29, yellow for 30 to 39, orange for 40 to 49, and red for 50 and over). The pattern of color that you see corresponds to the distribution of the attribute Age in the football field. Next you ask the crowd to remain in place and raise a colored flag according to their income, and so on for other attributes. For each attribute, you take a photo of the color distribution in the field. This color pattern corresponds to the color-coded maps visualized by SOM.

Finally, you can put all the photos side by side and inspect the dependencies. For example, you might see clusters of younger people (blue/green) as well as clusters of older people (orange/red). Further, you could detect some correlation between age clusters and income clusters: e.g., higher incomes occur in older groups. Continuing in this manner, you will discover further relationships among the defined attributes.

To better understand the vizualisation of SOM, a Matlab [130] toolbox for SOM [175] is available with some demos showing different examples and applications.

2.2.2 Analytical applications based on SOMs

The unique SOM representation and visualization are powerful instruments for data modeling and exploration. However, the above mentioned visualization is just the starting point for much more extensive and in-depth data mining and predictive modeling. The following have been chosen from a multitude of analytics capabilities to provide an overview of some prominent fields of application.

- **Clustering**: SOMs simplify clustering and allow the user to identify homogenous data groups visually, [174][61] [169] [40].

- **Prediction**: A combination of the non-linear data representation of the SOM with linear statistical prediction methods for each homogeneous sub-group improves prediction accuracy, [14] [176].

- **Data representation**: Data are highly compressed using statistical methods, allowing a single map that uses only a few megabytes of space to represent databases that are orders of magnitude larger, [107] [173].

- **Real-time classification**: New data can be located in the map extremely quickly up to 100,000 previously unseen data records can be classified per second allowing real-time assessment of new data, [47].

2.3 Topological Collaborative Clustering based on SOM

While collaboration can include a variety of detailed schemes, as we saw in the previous chapter, two of them are the most essential. We refer to them as horizontal and vertical modes of collaboration or simply horizontal and vertical clustering. More descriptively, given data sets $X[1], X[2], \ldots, X[P]$ where P denotes their number and $X[ii]$ stands for the ii-th data set (we adhere to the practice of using square brackets to identify a certain data set).

In *horizontal* clustering we have the same objects that are described in different feature spaces. In other words, these could be the same collection of patients whose records are developed within each medical institution. In horizontal clustering we deal with the same patterns and different feature spaces. The communication platform is based on through the partition matrix (Kernels in case of SOM). As we have the same objects, this type of collaboration makes sense. The confidentiality of data has not been breached: we do not operate on individual patterns but on the resulting information granules (fuzzy relations, that is, partition matrices). As this number is far lower than the number of data, the low granularity of these constructs moves us far from the original data.

Vertical clustering is complementary to horizontal clustering. Here the data sets are described in the same feature space but deal with different patterns. In other words, we consider that $X[1], X[2], \ldots, X[P]$ are defined in the same feature space, while each of them consists of different patterns, $dim(X[1]) = dim(X[2]) = \ldots = dim(X[P])$, while $X[ii] \neq X[jj]$. In vertical clustering we are concerned with different patterns but the same feature space. Hence communication at the level of the prototypes (which are high-level representatives of the data) becomes feasible. Again, because of the aggregate nature of the prototypes, the confidentiality requirement has been satisfied.

2.3.1 Topological Horizontal Collaboration

Here we formulate the underlying optimization problem implied by objective function-based clustering, and derive the detailed algorithm. There are P sets of data located in different spaces (viz., the patterns there are described by different features). As each subset deals with the same patterns, the number of elements in each subset is the same and is equal to N. What we are going to propose is that the collaboration between two subsets is established through an interaction coefficient which describes the intensity of the interaction and able to be estimated. Let $\alpha_{[ii]}^{[jj]}$ and $\beta_{[ii]}^{[jj]}$ non-negative values. The higher the value of the interaction (collaboration) coefficients, the stronger the collaboration between the corresponding data sets. In this paper, we will estimate these coefficients during the collaboration phase of the algorithm. The main idea of the horizontal collaboration between different SOM is that if an observation from the ii-th data set is projected on the j-th neuron in the ii-map, then that same observation in the jj-th data set will be projected on the same j neuron of the jj-th map or one of its neighboring neurons. In other words, *neurons that correspond to different maps should capture the same observations.* To accommodate the collaboration mechanism in the optimization process, the objective function of the SOM is expanded into the form

$$\begin{aligned} R_H^{[ii]}(\chi, w) &= \alpha_{[ii]}^{[jj]} \sum_{i=1}^{N} \sum_{j=1}^{|w|} \mathcal{K}_{\sigma(j,\chi(x_i))}^{[ii]} \|x_i^{[ii]} - w_j^{[ii]}\|^2 \\ &+ \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \beta_{[ii]}^{[jj]} \sum_{i=1}^{N} \sum_{j=1}^{|w|} \left(\mathcal{K}_{\sigma(j,\chi(x_i))}^{[ii]} - \mathcal{K}_{\sigma(j,\chi(x_i))}^{[jj]} \right)^2 \|x_i^{[ii]} - w_j^{[ii]}\|^2 \end{aligned} \tag{2.5}$$

where P represents the number of data sets (or the classifications), N the number of observations, $|w|$ is the number of prototype vectors from the ii-th SOM map (the number of neurons). $\chi(x_i)$ is the assignment function which allows to find the Best Matching Unit (BMU), it selects the neuron with the closest prototype from the data x_i using the Euclidean distance.

$$\chi(x_i) = argmin \left(\|x_i - w_j\|^2 \right) \tag{2.6}$$

$\sigma(i,j)$ represents the distance between two neurons i and j from the map, and it is defined as the length of the shortest path linking cells i and j on the SOM map.

$\mathcal{K}^{[cc]}_{\sigma(i,j)}$ is the neighborhood function on the $SOM[cc]$ map between two cells i and j. The nature of the neighborhood function $\mathcal{K}^{[cc]}_{\sigma(i,j)}$ is identical for all the maps, but its value varies from one map to another: it depends on the closest prototype to the observation that is not necessarily the same for all the SOM maps.

Algorithm 4: The horizontal SOM collaboration algorithm: *HCo-SOM*

```
Random the collaboration matrix $\alpha^{[jj]}_{[ii]}$
1. Local step:
for $t = 1$ to $N_{iter}$ do
For each $DB[ii]$, $ii = 1$ to $P$ :
Find the prototypes minimizing the classical SOM
    $w^* = \arg\min_{w} \left[ \sum_{i=1}^{N} \sum_{j=1}^{|w|} \mathcal{K}^{[ii]}_{\sigma(j,\chi(x_i))} \|x_i^{[ii]} - w_j^{[ii]}\|^2 \right]$
2. Collaboration step:
For the horizontal collaboration of the $[ii]$ map with the $[jj]$ map:
Collaboration Phase 1:
Update the prototypes of the $[ii]$-th map minimizing the objective function of the
horizontal collaboration using the expression 2.9
Collaboration Phase 2:
The confidence exchange parameter is adapted using the following expression:
    $\alpha^{[jj]}_{[ii]}(t+1) = \alpha^{[jj]}_{[ii]}(t) + \frac{\sum_{i=1}^{N} \sum_{j=1}^{|w|} \mathcal{K}^{[ii]}_{\sigma(j,\chi(x_i))}}{2 \sum_{i=1}^{N} \sum_{j=1}^{|w|} \left( \mathcal{K}^{[ii]}_{\sigma(j,\chi(x_i))} - \mathcal{K}^{[jj]}_{\sigma(j,\chi(x_i))} \right)^2}$    (2.7)
    $\textit{with} \quad K_{ij} = \left( K^{[ii]}_{\sigma(j,\chi(x_i))} - K^{[jj]}_{\sigma(j,\chi(x_i))} \right)^2$
    $\textit{and} \quad \beta \leftarrow \alpha^2$
```

The value of the collaboration parameter α is determined during the first phase of the collaboration step, and $\beta = \alpha^2$. This parameter allows to determine the importance of the collaboration between each two data sets, i.e. to learn the collaboration confidence between all data sets and maps. Its value belongs to [1-10], it is 1 for the neutral link, when no importance to collaboration is given, and 10 for the maximal collaboration within a map. Its value varies after each iteration during the collaboration step. In the case of the horizontal collaborative learning, as is shown in the Algorithm 4, the value of the collaboration confidence parameter depends on topological similarity between both collaboration maps. To compute the collaborated prototypes matrix, we use gradient optimization technique, we obtain the following expression:

$$w^{*[ii]} = \arg\min_{w} \left[R_H^{[ii]}(\chi, w) \right] \tag{2.8}$$

$$w_{jk}^{*[ii]}(t+1) = w_{jk}^{*[ii]}(t) + \frac{\sum\limits_{i=1}^{N} K_{\sigma(j,\chi(x_i))}^{[ii]} x_{ik}^{[ii]} + \sum\limits_{\substack{jj=1 \\ jj \neq ii}}^{P} \sum\limits_{i=1}^{N} \alpha_{[ii]}^{[jj]} L_{ij} x_{ik}^{[ii]}}{\sum\limits_{i=1}^{N} K_{\sigma(j,\chi(x_i))}^{[ii]} + \sum\limits_{\substack{jj=1 \\ jj \neq ii}}^{P} \sum\limits_{i=1}^{N} \alpha_{[ii]}^{[jj]} L_{ij}} \tag{2.9}$$

where

$$L_{ij} = \left(K_{\sigma(j,\chi(x_i))}^{[ii]} - K_{\sigma(j,\chi(x_i))}^{[jj]} \right)^2$$

Indeed, during the collaboration with a SOM map, the algorithm takes into account the prototypes of the map and its topology (the neighborhood function). The horizontal collaboration algorithm is presented in Algorithm 4.

2.3.2 Topological Vertical Collaboration

In the case of vertical collaborative clustering, contrarily to the horizontal case, we deal with different data sets where all patterns are described in the same feature space. We establish communication at the level of prototypes of the data sets, they are defined in the same feature space. The basic idea of collaboration in this case is the following: a neuron j of ii-th SOM map and the same neuron j of the jj-th map should be very similar using the Euclidean distance. In other words, *neurons that correspond to the different maps should represent groups of similar observations.* The proposed objective function governing a search for structure in the ii-th data set is

$$\begin{aligned} R_V^{[ii]}(\chi, w) &= \alpha_{[ii]}^{[jj]} \sum_{i=1}^{N} \sum_{j=1}^{|w|} \mathcal{K}_{\sigma(j,\chi(x_i))}^{[ii]} \|x_i^{[ii]} - w_j^{[ii]}\|^2 \\ &+ \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \beta_{[ii]}^{[jj]} \sum_{i=1}^{N^{[ii]}} \sum_{j=1}^{|w|} \left(\mathcal{K}_{\sigma(j,\chi(x_i))}^{[ii]} - \mathcal{K}_{\sigma(j,\chi(x_i))}^{[jj]} \right)^2 \|w_j^{[ii]} - w_j^{[jj]}\|^2 \end{aligned} \tag{2.10}$$

where P represents the number of data sets, N - the number of observations of the ii-th data set, $|w|$ is the number of prototype vectors from the ii-SOM map and which is the same for all the maps.

We will estimate the coefficients of collaboration during the collaboration phase, as same as we did in the horizontal case.

Using the gradient optimization procedure, we obtain the following formulas to compute the prototypes matrix:

$$w^{*[ii]} = \arg\min_{w} \left[R_V^{[ii]}(\chi, w) \right] \tag{2.11}$$

$$w_{jk}^{*[ii]}(t+1) = w^{*[ii]}(t) + \frac{\sum\limits_{i=1}^{N} K^{[ii]}_{\sigma(j,\chi(x_i))} x^{[ii]}_{ik} + \sum\limits_{\substack{jj=1 \\ jj \neq ii}}^{P} \sum\limits_{i=1}^{N^{[ii]}} \alpha^{[jj]}_{[ii]} L_{ij} w^{[jj]}_{ik}}{\sum\limits_{i=1}^{N} K^{[ii]}_{\sigma(j,\chi(x_i))} + \sum\limits_{\substack{jj=1 \\ jj \neq ii}}^{P} \sum\limits_{i=1}^{N} \alpha^{[jj]}_{[ii]} L_{ij}} \tag{2.12}$$

where

$$L_{ij} = \left(K^{[ii]}_{\sigma(j,\chi(x_i))} - K^{[jj]}_{\sigma(j,\chi(x_i))} \right)^2$$

The learning algorithm in this case is presented by Algorithm 5.

Algorithm 5: Vertical Collaboration algorithm of SOM: *VCo-SOM*

```
Choose randomly the collaboration matrix $\alpha^{[jj]}_{[ii]}$
1. Local step:
for $t = 1$ to $N_{iter}$ do
For each $DB[ii]$, $ii = 1$ to $P$ :
Find the prototypes minimizing the classical SOM objective function:
    $w^{*} = \arg\min_{w} \left[ \sum_{i=1}^{N} \sum_{j=1}^{|w|} \mathcal{K}^{[ii]}_{\sigma(j,\chi(x_i))} \|x_i^{[ii]} - w_j^{[ii]}\|^2 \right]$
2. Collaboration step:
For the vertical collaboration of the $[ii]$-th map with the map $[jj]$:
Collaboration phase 1:
Update the prototypes of the $[ii]$ map minimizing the objective function of the vertical collaboration using the expression 2.12.
Collaboration phase 2:
The collaboration confidence parameter is adapted using the following expression:
    $\alpha^{[jj]}_{[ii]}(t+1) = \alpha^{[jj]}_{[ii]}(t) + \frac{\sum_{i=1}^{N} \sum_{j=1}^{|w|} K^{[ii]}_{\sigma(j,\chi(x_i))} \|x_i^{[ii]} - w_j^{[ii]}\|^2}{2 \sum_{i=1}^{N} \sum_{j=1}^{|w|} \left( K^{[ii]}_{\sigma(j,\chi(x_i))} - K^{[jj]}_{\sigma(j,\chi(x_i))} \right)^2 \|w_j^{[ii]} - w_j^{[jj]}\|^2}$    (2.13)
    $and \quad \beta \leftarrow \alpha^2$
```

2.4 Experimental Results

To evaluate the proposed collaborative approaches on SOM we applied the algorithms on several data sets of different size and complexity. The used data sets are the following: *waveform, Wisconsin Diagnostic Breast Cancer (wdbc), Isolet, Madelon and Spambase.* We will give more details on the results obtained on the waveform data set to illustrate the principle of the proposed approaches, especially in the validation since the *waveform* data set contains 21 relevant variables and 19 noisy variables, so it is useful to show the effect of the collaboration in the horizontal approach.

2.4.1 Validation criteria

As criteria to validate the approaches we use the quantization error (distortion) on many maps of different sizes and the purity index for each SOM.

Quantization error

The quantization error [164] is the most used criteria to evaluate the quality of a Kohonen's topological map. This error measures the average distance between each data vector and its winning neuron, e.g. Best Matching Unit (BMU). It is calculated using the following expression:

$$qe = \frac{1}{N} \sum \|x^{(i)} - w_{x_i}\|^2 \tag{2.14}$$

where N represents the number of data vectors and $w_{x^{(i)}}$ is the nearest prototype to the vector x_i. The values of the quantization error depends on the size of data sets and on the sizes of built maps, so these values can alter according to the data set.

Purity index

The purity index of a map is equal to the average purity of all the clusters of the map. Larger purity values indicate better clustering.

Assuming we have K clusters c_r, $r = 1, \ldots, K$. First, we calculate the purity of each cluster, which is given by:

$$Pu(c_r) = \frac{1}{|c_r|} \, max_i(|c_r^i|)$$

where $|c_k|$ is the total number of data associated to the cluster c_k, $|c_r^i|$ is the number of objects in c_r with class label i.

In other words, $Pu(c_r)$ is a fraction of the overall cluster size that the largest class of objects assigned to that cluster represents. Therefore, the overall purity of the clustering solution is obtained as a weighted sum of the individual cluster purities and given as:

$$Purity = \sum_{r=1}^{K} \frac{|c_r|}{N} Pu(c_r) \tag{2.15}$$

where K is the number of clusters and N is the total number of objects.

2.4.2 Data sets

All data sets are available on UCI Machine Learning Repository [7].

- *Waveform data set*: This data set consists of 5000 instances divided into 3 classes. The original base included 40 variables, 19 are all noise attributes with mean 0 and variance 1. Each class is generated from a combination of 2 of 3 "base" waves.
- *Wisconsin Diagnostic Breast Cancer (WDBC)*: This data has 569 instances with 32 variables (ID, diagnosis, 30 real-valued input variables). Each data observation is labeled as benign (357) or malignant (212). Variables are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.
- *Isolet*: This data set was generated as follows. 150 subjects spoke the name of each letter of the alphabet twice. Hence, we have 52 training examples from each speaker. The speakers are grouped into sets of 30 speakers each, and are referred to as isolet1, isolet2, isolet3, isolet4, and isolet5. The data consists of 1559 instances and 617 variables. All variables are continuous, real-valued variables scaled into the range -1.0 to 1.0.
- *Madelon*: MADELON is an artificial data set, which was part of the NIPS 2003 [1] 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). The order of the features

and patterns were randomized. The original data set was splitting in three parts (learning, validation and test), but we used only 2600 observations from learning set and from validation for which the classes were known.

- *SpamBase*: The SpamBase data set is composed from 4601 observations described by 57 variables. Every variable described an e-mail and its category: spam or not-spam. Most of the attributes indicate whether a particular word or character was frequently occurring in the e-mail. The run-length attributes (55-57) measure the length of sequences of consecutive capital letters.

2.4.3 Data partitioning

The data sets mentioned above are unified and need to be divided in subsets in order to have distributed data "scenarios", we will use the vertical and horizontal partitioning (Figure 2.3). In the horizontal approach we divide the data sets into subsets so that each algorithm operates on different features considering, however, the same set of individuals. In the case of vertical approach, each algorithm operates on the same features, dealing, however, with different set of individuals.

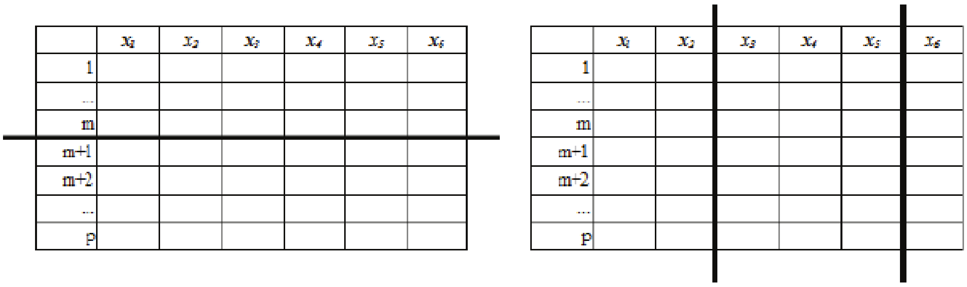


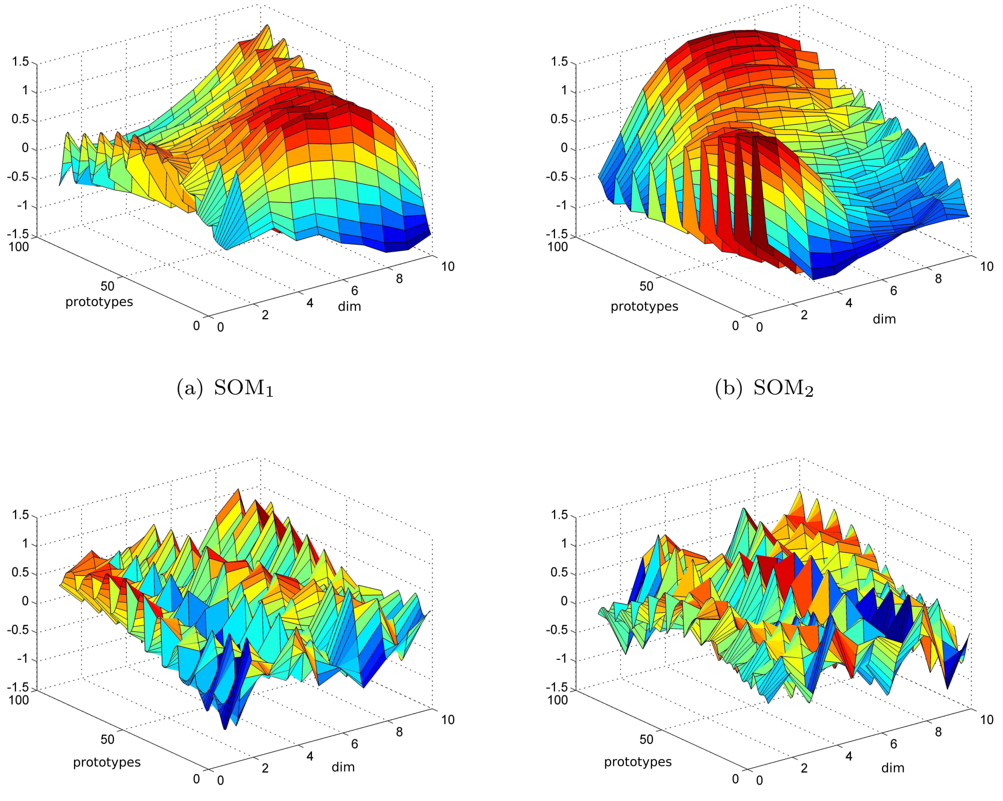
FIGURE 2.3: Vertical (left) and horizontal (right) partitioning.

2.4.4 Interpretation of the approach on the Waveform data set

We divided the Waveform data set, of size 5000 × 40, into four subsets to assume a scenario of a horizontal collaboration between four sites. The first and the second part of the data set $2 \times (5000 \times 10)$ correspond to all the relevant variables and the third and fourth part $2 \times (5000 \times 10)$ contain noisy variables.

As the first an second data sets are relevant, we expect that the collaboration confidence within these data sets is bigger than the 3rd and 4th data sets.

We selected maps of size 10 × 10. Then we achieved the local step of the proposed approach on all four data sets which is to learn a SOM for all observations of these data



(a) SOM$_1$

(b) SOM$_2$

(c) SOM$_3$

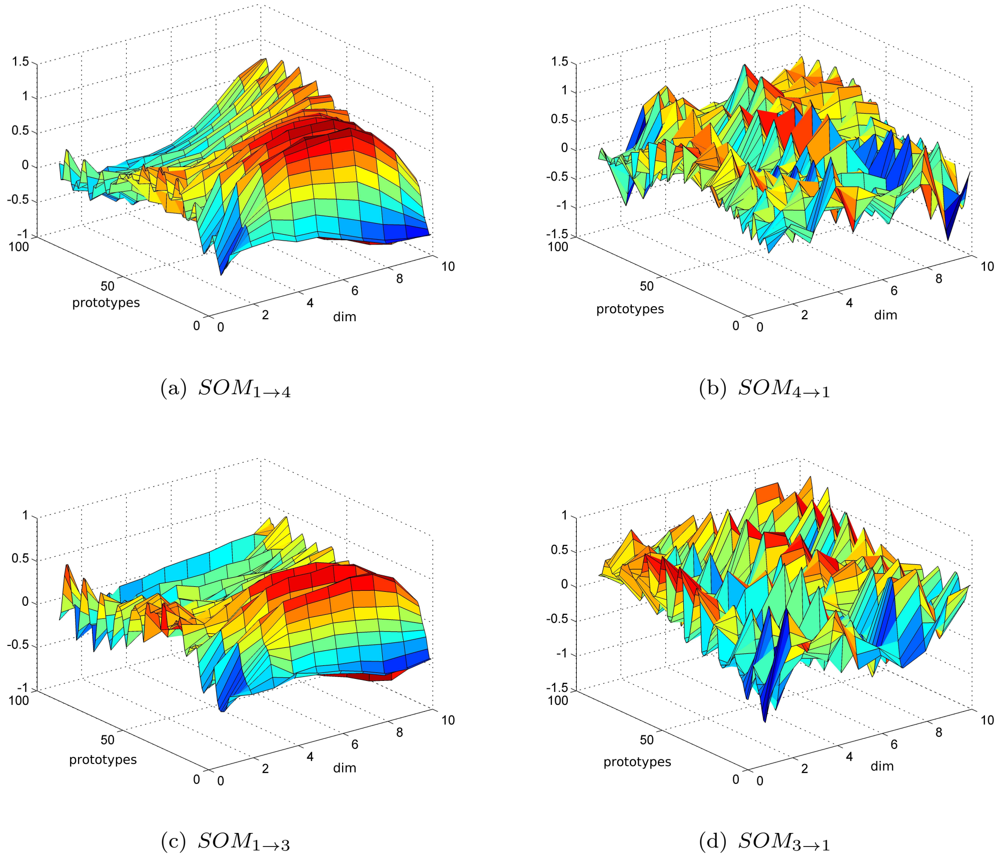
(d) SOM$_4$

FIGURE 2.4: Visualization of the prototypes after the first local step (classical SOM)

sets. Figure 2.4 represent the prototypes vectors obtained on all the four data sets after the local step of the new learning approach. X-axis and Y-axis represent respectively the indices of variables and prototypes for these maps. Figures 2.4(a) and 2.4(b) correspond to the maps which contain the relevant variables from the waveform data set (1-20) which are represented by the red (darker) color and have an index of purity of 81.64% and 81.5% respectively. Knowing that the purity of the map presenting the waveform data set before partitioning is 85.84% and the quantization error is 6.12.

We applied the second step of our algorithm to exchange the clustering information between all the maps without using the original data. Figures 2.5(a) and 2.5(b) illustrate the collaboration between 1st and 4th data sets. After the collaboration, the purity index decreased to 78.93% because the SOM$_1$ map (81.64%) has used the information from a noisy map (SOM$_4$) which has very low purity index (40.21%). Contrarily, by applying the collaboration in the opposite direction, the purity index of the $SOM_{4\to1}$ map increased to 42.45% due to the collaboration with the relevant SOM$_1$ map (75.71% of purity). The learned collaboration confidence parameter are for the SOM$_1$, $\alpha = 6.03$, and for SOM$_4$, $\alpha = 1.34$ which means that the algorithm gives more importance to the collaboration with SOM$_1$ and less importance to SOM$_4$ map which contains noisy features.

After the collaboration of the "relevant" second data set with the irrelevant SOM$_3$ map, the purity index decreased to 78.18% because the SOM$_2$ map (81.5%) has used the information from a noisy map (SOM$_3$) with a very low purity index (39.37%). Contrarily, by applying the collaboration step in the opposite direction, the purity index of the SOM$_{2\to3}$ map increased to 41.67% due to the collaboration with the relevant SOM$_2$ map with a collaboration confidence parameter equals to 5.9 higher than the confidence parameter with the noisy SOM$_3$ map which value is 1.2.



(a) $SOM_{1\to4}$

(b) $SOM_{4\to1}$

(c) $SOM_{1\to3}$

(d) $SOM_{3\to1}$

FIGURE 2.5: Horizontal collaboration between the data sets 1 and 4 and between the 1st and 3rd data set.

The collaboration of a noisy map with a relevant map leads to an improvement of its quality (the purity index). The task of the horizontal collaboration is a complex problem because in an unsupervised learning process it is difficult to identify relevant maps and we are forced to make the collaboration in both directions, and here comes the importance of learning the collaboration confidence parameters, in order to give more importance to some links.

Table 2.1 summarizes the purities of the maps and the quantization errors before and after collaboration. As for the indices of purity, the quantization errors are improving (decreasing) after a collaboration with a more relevant map. We improve these indices due to the collaboration process and the learning of the collaboration confidence parameters, the value of each collaboration parameter is given in Table 2.1.

Table 2.1: Experimental results of the horizontal collaboration approach on the waveform data set

Horizontal Collaboration			
Map	Purity	qe	α
SOM_1	81.64	1.98	
SOM_2	79.61	1.87	
SOM_3	47.19	2.64	
SOM_4	40.21	2.41	
$SOM_{1\to4}$	62.47	2.14	1.2
$SOM_{4\to1}$	54.63	2.27	5.9
$SOM_{2\to3}$	78.93	2.05	1.34
$SOM_{3\to2}$	41.45	2.35	6.03

Vertical collaboration process: waveform data set

To apply vertical collaboration on waveform data set, we divided the database into 4 subsets, the division was made randomly on the observations. We got 4 databases of size 1250 × 40 and we chose 1 as the value of collaboration parameter (for the both directions). The obtained results are summarized in Table 2.2. We note that in most cases the purity index increases, as is the case for $SOM_{2\to1}$, $SOM_{3\to4}$, $SOM_{1\to4}$ and $SOM_{4\to1}$ and the collaboration confidence parameters are similar because all the maps are similar. As all four data sets are described in the same feature space, the purity of the maps before and after the collaboration is higher compared to the horizontal collaboration. The quantization error is also improved for the maps obtained after the collaboration with the maps having a lower quantization error.

2.4.5 Validation on other data sets

We applied the same experimental protocol on other databases and all computed indices are presented in Tables 2.3 and 2.4, for horizontal and vertical collaboration respectively.

The size of all the used maps were fixed to 10 × 10. From the Tables 2.3 and 2.4, we note that the purity index of the SOM maps after the horizontal collaboration increases for each data set and the quantization error decreases. This is due to the use of the

Table 2.2: Experimental results of the vertical collaboration approach on the waveform data set

Vertical Collaboration			
Map	Purity	qe	α
SOM_1	88.33	5.64	
SOM_2	87.75	5.83	
SOM_3	90.04	5.24	
SOM_4	88.76	5.57	
$SOM_{1\to2}$	88.06	5.62	2.2
$SOM_{2\to1}$	87.93	5.79	2.47
$SOM_{3\to4}$	90.12	5.07	2.36
$SOM_{4\to3}$	89.57	5.16	2.27
$SOM_{1\to4}$	88.46	5.59	2.41
$SOM_{4\to1}$	88.57	5.51	2.36

Table 2.3: Experimental results of the horizontal collaborative approach on different data sets

Data set	Map	Horizontal Collaboration		
		Purity	qe	α
Wdbc	SOM_1	94.95	1.99	
	SOM_2	97.27	2.07	
	$SOM_{1\to2}$	95.77	1.84	1.74
	$SOM_{2\to1}$	97.32	1.94	2.12
Isolet	SOM_1	81.20	12.61	
	SOM_2	95.12	14.45	
	$SOM_{1\to2}$	81.39	12.21	2.05
	$SOM_{2\to1}$	96.06	14.18	1.86
Madelon	SOM_1	60.88	15.58	
	SOM_2	62.64	15.50	
	$SOM_{1\to2}$	61.01	15.48	1.65
	$SOM_{2\to1}$	63.57	15.40	1.79
SpamBase	SOM_1	83.86	3.45	
	SOM_2	85.72	2.55	
	$SOM_{1\to2}$	84.17	3.23	1.92
	$SOM_{2\to1}$	83.59	2.41	1.59

information from the maps related to the collaborative data sets. Also, we can note that the values of the collaboration confidence parameters are computed using the topological structure of the distant maps (distant classifications) and learning these parameters allows the system to detect the important collaboration links and directions and to avoid a collaboration with are irrelevant classification.

Table 2.4: Experimental results of the vertical collaborative approach on different data sets

Data set	Map	Vertical Collaboration		
		Purity	qe	α
Wdbc	SOM_1	96.71	90.54	
	SOM_2	97.87	67.60	
	$\text{SOM}_{1\to2}$	96.99	71.49	1.42
	$\text{SOM}_{2\to1}$	97.49	61.47	4.16
Isolet	SOM_1	98.85	8.19	
	SOM_2	98.46	8.76	
	$\text{SOM}_{1\to2}$	79.54	8.34	1.93
	$\text{SOM}_{2\to1}$	98.30	8.78	2.04
Madelon	SOM_1	69.71	61.23	
	SOM_2	69.87	61.15	
	$\text{SOM}_{1\to2}$	74.57	59.59	2.26
	$\text{SOM}_{2\to1}$	70.71	59.55	2.39
SpamBase	SOM_1	76.26	61.83	
	SOM_2	70.43	48.27	
	$\text{SOM}_{1\to2}$	72.28	45.98	1.47
	$\text{SOM}_{2\to1}$	69.78	36.74	4.25

For the vertical collaboration experiments (Table 2.4), the size of all the maps is set to 10 × 10, except for the Isolet data set which map size is 5 × 5.

For the Wdbc data set, we note that the purity index of the first SOM map after the collaboration has improved. Contrarily, the purity of the second SOM map after the collaboration decreased. We also note that the quantization error of the first and second map has improved after the collaboration. For the Isolet data set, we do not observe any improvement on the maps obtained after the collaboration compared with that before. The purity of the maps and the quantization errors after the collaboration are improved for the Madelon dataset. For the Spam data set, the quantization error has improved. For the vertical collaboration approach, these results show that the purity of maps and the quantization error is not always improved after collaborating the maps, and depends strongly on the relevance of the collaborative map (the quality of the collaborative classification) and on the confidence on this map (the collaboration parameter). This conclusion corresponds to the intuitive understanding of the principle and to the consequences of such cooperation.

2.5 Conclusion

In this chapter, we proposed a topological collaborative clustering based on SOM, for both cases: horizontal and vertical. Plus a methodology to learn the collaboration confidence parameters.

Collaborative clustering allows the interaction between the different sources of information for the purpose to reveal (to detect) the underlying structures and the regularities from the data sets. It can be treated as a process of consensus building where we search for a structure that is common to all the data sets. The impact of the collaboration matrix (the collaboration confidence values) over the overall effect of the collaboration is very important since in an unsupervised learning model there is no information about the data structure. The proposed horizontal learning approach is adapted for collaboration between data sets that describe the same observations but with different variables, and in this case choosing of the value of the collaboration confidence becomes very important as the data sets are in different feature spaces. Contrarily, the vertical collaborative learning approach is adapted to the problem of collaboration of several data sets containing the same variables but with different observations.

Since collaborative clustering is based on an specific algorithm in its local phase, we believe that switching between algorithms, e.g. choosing a better clustering algorithm, in the local phase affects the final results obtained by the collaboration phase, hence leads to a better construction of the consensus.

Despite that SOM has become very popular and was applied in several domains, SOM suffers from some limitations. That is why we decided to present a collaborative clustering scheme using a concurrent method to SOM, which is the Generative Topographic Mapping (GTM) [25]. In next chapter, we present the difference between SOM and GTM, then we present the collaborative clustering algorithm based on GTM, [66, 67].

Chapter 3

Collaborative Clustering using A Generative Model

3.1 Introduction

In the previous chapter, we presented an algorithm of collaborative clustering based on SOM as an algorithm of clustering in the local phase. Although the SOM has been subject of a considerable amount of research and applied to a wide range of tasks, there are still a number of problems that remain unresolved [115]. Some of these problems are:

- The SOM does not define a density model in the data space. Attempts has been made to formalize the relationship between the distribution of reference vectors and the distribution of the data, but has only succeeded under very restricted conditions [158, 159].
- There is no general guarantee the training algorithm will converge.
- There is no theoretical framework based on which appropriate values for the model parameters can be chosen.
- For SOM the choice of how the neighborhood function should shrink over time during training is arbitrary, and so this must be optimized empirically [26].
- It is not obvious how SOM models should be compared to other SOM models or to models with different architectures.
- The mapping from the topographic space to the data space in the original SOM is only defined at the locations of the nodes.

These problems would be resolved in a **probabilistic** setting, i.e. using **generative** models. In a generative model the data is assumed to arise by first probabilistically picking a point in a low-dimensional space, mapping the point to the observed high-dimensional input space (via a smooth function), then adding noise in that space. The parameters of the low-dimensional probability distribution, the smooth map and the noise are all learned from the training data using the Expectation-Maximization (EM) algorithm.

Generative models are defined stochastically and try to estimate the distribution of data by defining a density model with low intrinsic dimensionality within the multivariate data space. Possibly, Factor Analysis (FA) [113, 124] is the most widely used generative model. It must be noted though, that FA is sometimes confused with rotated variations of PCA [100] and both are used in similar applications.

Most of the interest in generative models stems from the fact that they fit naturally into the Statistical Machine Learning category and, in general, to the much wider framework of probability theory and statistics. Furthermore, generative models can directly make use of well-founded techniques for fitting them to data, combining different models, missing data imputation, outlier detection, etc.

GTM is a non-linear generative model introduced in [25]. In short, it was defined to retain all the useful properties of Kohonen's Self-Organizing Maps (SOM) [114], such as the simultaneous clustering and visualization of multivariate data, while eluding most of its limitations through a fully probabilistic formulation.

Basing on the above, we decide to choose the GTM [25–27] as a local step for collaborative clustering. In this chapter, we present the standard model of GTM, then we present an approach to use GTM in collaborative clustering [66, 67].

3.2 Expectation-Maximization (EM)

In some ways, the Expectation Maximization (EM) [50] approach to clustering can be seen as an extension of K-means, with a more solid theoretical underpinning. What is the model that K-means is applying to the data? It is that the each data point belongs to one of K clusters that are defined by a set of K points. EM relaxes the assumption that every point comes from a single cluster, and instead models the data as the result of some ***generative*** process. For example, typically EM uses a model that says that the data is being generated by a mixture of Gaussian (Normal) distributions. Each distribution gives a probability density over the whole of the space for generating points. If there are K such distributions, then the probability density function comes from taking the scaled union of these individual densities. Each of the distributions can have different parameters, in the case of Gaussians, these need only be the mean and standard deviation for one dimension; for higher dimensions, then there are more parameters to describe the shape of the distribution.

The Expectation Maximization stage is, given the model and the data, to find the settings of the parameters of the model that best explain the data. That is, they are the most likely settings of the parameters given the data. The result of this means that we do not allocate points to clusters but rather for each data point we can evaluate the produced model at that point and see the relative probabilities that this point came from each of the K different distributions. It is this model which represents the clustering, and which can be used to predict future outcomes.

In order to generate the maximum likelihood settings of the parameters, various algorithms can be employed which, at a high level, resemble K-means. From an initial guess of the settings of the parameters, successive passes over the data refine these guess and improve the fit of the data to the current model. The details depend on the distributions used in the model (Gaussian, Log-Normal, Poisson, Discrete). For a model with a single Gaussian distribution, the sample mean is the maximum likelihood estimator. For two or more Gaussians, one can write out the expression for the mixture of these distributions, and, based on the current estimates of the parameters, compute the likelihood that each input point was generated by each of the distributions. Based on these likelihoods, we can create new settings of the parameters, and iterate. Each step increases the likelihood of the observed data given the current parameters, until a maximum is reached. Note that this maximum may be a local maximum, rather than the global maximum. The maximum that is reached depends on the initial setting of parameters. Hence we see the connection to K-means, the principal differences being the greater emphasis on an underlying model, and the way that each point has a probability or likelihood of belonging to each cluster, rather than a unique parent cluster.

3.2.1 The Gaussian Mixtures

The Gaussian, also known as the normal distribution, is a widely used model for the distribution of continuous variables. In the case of a single variable x, the Gaussian distribution can be written in the form

$$\mathcal{N}(x|m,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-m)^2\right\} \tag{3.1}$$

where m is the mean and σ^2 is the variance.

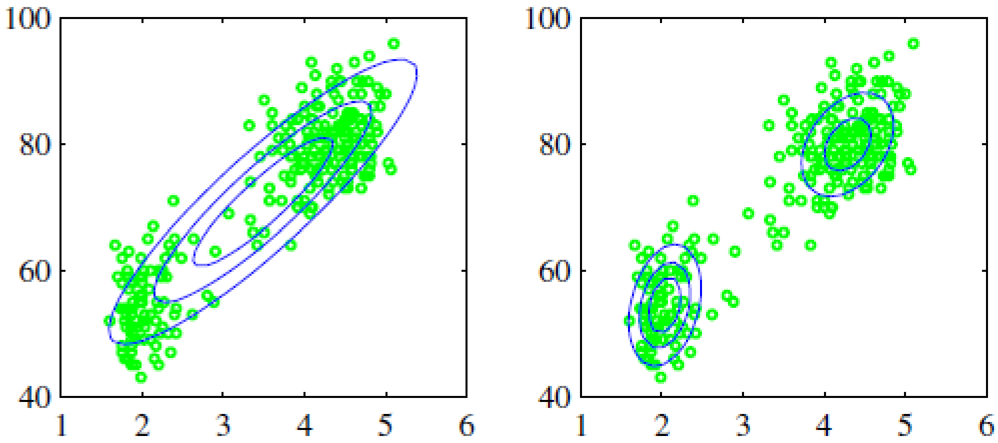


FIGURE 3.1: Plots of the 'old faithful' data in which the blue curves show contours of constant probability density. On the left is a single Gaussian distribution which has been fitted to the data using maximum likelihood. On the right the distribution is given by a linear combination of two Gaussians which has been fitted to the data by maximum likelihood using the EM technique, and which gives a better representation of the data. [24]

For a D-dimensional vector x, the multivariate Gaussian distribution take the form

$$\mathcal{N}(x|\mu,\Sigma) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\} \tag{3.2}$$

where μ is a D-dimensional mean vector, Σ is a $D \times D$ covariance matrix, and $|\Sigma|$ denotes the determinant of Σ.

The Gaussian distribution arises in many different contexts and can be motivated from a variety of different perspectives. But it suffers from significant limitations when it comes to modeling real data sets. Consider the example shown in Figure 3.1 applied on the 'Old Faithful' data set described previously in this chapter. We see that the data set forms two dominant clumps, and that a simple Gaussian distribution is unable to capture this structure, whereas a linear superposition of two Gaussians gives a better characterization of the data set. Such superpositions, formed by taking linear combinations of more basic distributions such as Gaussians, can be formulated as probabilistic models known as *mixture distributions* [131, 133].

We therefore consider a superposition of K Gaussian densities of the form

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k,\Sigma_k) \tag{3.3}$$

which is called a *mixture of Gaussians.* Each Gaussian density $\mathcal{N}(x|\mu_k, \Sigma_k)$ is called a *component* of the mixture and has its own mean μ_k and covariance Σ_k.

The parameters π_k are called *mixing coefficients.* They verify the conditions

$$\sum_{k=1}^{K} \pi_k = 1 \quad \text{and} \quad 0 \leq \pi_k \leq 1 \tag{3.4}$$

In order to find an equivalent formulation of the Gaussian mixture involving an explicit latent variable, let us introduce a K-dimensional binary random variable z having a 1-of-K representation in which a particular element z_k is equal to 1 and all other elements are equal to 0. The values of z_k therefore satisfy $z_k \in \{0, 1\}$ and $\sum_k z_k = 1$, and we see that there are K possible states for the vector z according to which element is nonzero. The marginal distribution over z is specified in terms of the mixing coefficients π_k , such that

$$p(z_k = 1) = \pi_k$$

The conditional distribution of x given a particular value for z is a Gaussian

$$p(x|z_k = 1) = \mathcal{N}(x|\mu_k, \Sigma_k) \tag{3.5}$$

The joint distribution is given by $p(z)p(x|z)$, and the marginal distribution of x is then obtained by summing the joint distribution over all possible states of z to give

$$p(x) = \sum_{z} p(z)p(x|z) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \tag{3.6}$$

Now, we are able to work with the joint distribution $p(x|z)$ instead of the marginal distribution $p(x)$. This leads to significant simplification, most notably through the introduction of the Expectation-Maximization (EM) algorithm.

Another quantity that play an important role is the conditional probability of z given x. We shall use $r(z_k)$ to denote $p(z_k = 1|x)$, whose value can be found using Bayes' theorem

$$
\begin{aligned}
r(z_k) = p(z_k = 1|x) &= \frac{p(z_k = 1)p(x|(z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(x|(z_j = 1)} \\
&= \frac{\pi_k \mathcal{N}(x|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x|\mu_j, \Sigma_j)}
\end{aligned} \tag{3.7}
$$

We shall view π_k as the prior probability of $z_k = 1$, and the quantity $r(z_k)$ as the corresponding posterior probability once we have observed x. As we shall see in next section, $r(z_k)$ can also be viewed as the *responsibility* that component k takes for 'explaining' the observation x.

3.2.2 EM for Gaussian Mixtures

Suppose we have a data set of observations $\{x_1, \ldots, x_N\}$, which gives a data set X of size $N \times D$ like described previously in this chapter, and we wish to model this data using a mixture of Gaussians. Similarly, the corresponding latent variable are denoted by an $N \times K$ matrix Z with rows z_n^K.

If we assume that the data points are i.i.d. (independent and identically distributed), then we can calculate the log of the likelihood function, which is given by

$$
\ln p(X|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k) \right\} \tag{3.8}
$$

An elegant and powerful method for finding maximum likelihood solutions for this models with latent variables is called the expectation-maximization algorithm, or EM algorithm [50, 132].

Setting the derivatives of $\ln p(X|\pi, \mu, \Sigma)$ in (3.8) respectively with respect to the μ_k, Σ_k and π_k to zero, we obtain

$$
\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r(z_{nk}) x_n \tag{3.9}
$$

where

$$
N_k = \sum_{n=1}^{N} r(z_{nk})
$$

is the effective number of points assigned to cluster k.

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} r(z_{nk})(x_n - \mu_k)(x_n - \mu_k)^T \tag{3.10}$$

and

$$\pi_k = \frac{N_k}{N} \tag{3.11}$$

We first choose some initial values for the means, covariances, and mixing coefficients. Then we alternate between the following two updates that we shall call the E step and the M step. In the *expectation* step, or E step, we use the current values for the parameters to evaluate the posterior probabilities, or responsibilities, given by Eq. 3.7. We then use these probabilities in the *maximization* step, or M step, to re-estimate the means, covariances, and mixing coefficients using the results in Equations 3.9, 3.10 and 3.11. The algorithm of EM for mixtures of Gaussians is shown in Algorithm 6.

The EM algorithm for a mixture of two Gaussians applied to the rescaled Old Faithful data set is illustrated in Figure 3.2. In plot (a) we see the initial configuration, the Gaussian component are shown as blue and red circles. Plot (b) shows the result of the initial E step where we update the responsibilities. Plot (c) shows the M step where we update the parameters. Plots (d), (e), and (f) show the results after 2, 5, and 20 complete cycles of EM, respectively. In plot (f) the algorithm is close to convergence.

3.2.3 The EM Algorithm in General

In this section, we present the general view of the EM algorithm. The goal of the EM algorithm is to find maximum likelihood solutions for models having latent variables. We denote X the data matrix, Z the latent variables matrix. Let us denote θ the set of all model parameters. Then the log likelihood function is given by

$$\ln p(X|\theta) = \ln \left\{ \sum_{Z} p(X, Z|\theta) \right\} \tag{3.12}$$

Note that if the latent variables are continuous we get similar equations, we only replace the over Z with an integral.

The presence of the sum prevents the logarithm from acting directly on the joint distribution, resulting in complicated expressions for the maximum likelihood solution.

Suppose that, for each observation in X, we were told the corresponding value of the latent variable Z. We shall call $\{X, Z\}$ the *complete* data set, and we shall refer to the actual observed data X as *incomplete*. The likelihood function for the complete data

```
Algorithm 6: The EM for Gaussian Mixtures
Data: $\mathbf{X} = \{x_{kd},\ \ k = 1, \ldots, N, d = 1, \ldots, D\}$ where $D$ is the dimension of the feature
      space. $Z$ the latent variables matrix.
Result: Posterior probabilities $r(z_{nk})$ and the model parameters $\mu, \Sigma$ and $\pi$.
Initialization:
-Choose a value for $K$, $1 < K < N$
-Initialize the means $\mu_k$, the covariances $\Sigma_k$ and mixing coefficients $\pi_k$ randomly
-Evaluate the initial value of the log likelihood.
Learning: repeat
for $l = 1, 2, \ldots$ do
    -E step: Evaluate the responsibilities using the current parameter values:
        $$r(z_{nk}) = \frac{\pi_k \mathcal{N}(x|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x|\mu_j, \Sigma_j)}$$
    -M step: Re-estimate the parameters using the current responsibilities:
        $$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r(z_{nk}) x_n$$
        $$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} r(z_{nk})(x_n - \mu_k)(x_n - \mu_k)^T$$
        $$\pi_k = \frac{N_k}{N}$$
        $$\text{where} \quad N_k = \sum_{n=1}^{N} r(z_{nk})$$
    -Evaluate the log likelihood
        $$\ln p(X|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k) \right\}$$
Until convergence of either the parameters or the log likelihood. If the convergence
criterion is not satisfied return to E step.
```

set simply takes the form $\ln p(X, Z|\theta)$, and we shall suppose that maximization of this complete-data log likelihood function is straightforward.

In practice, however, we are not given the complete data set $\{X, Z\}$, but only the incomplete data X. Our state of knowledge of the values of the latent variables in Z is given only by the posterior distribution $p(Z|X, \theta)$. Because we cannot use the complete-data log likelihood, we consider instead its expected value under the posterior distribution of the latent variable, which corresponds to the E step of the EM algorithm. In the subsequent M step, we maximize this expectation. If the current estimate for the parameters is denoted θ^{old}, then a pair of successive E and M steps gives rise to a

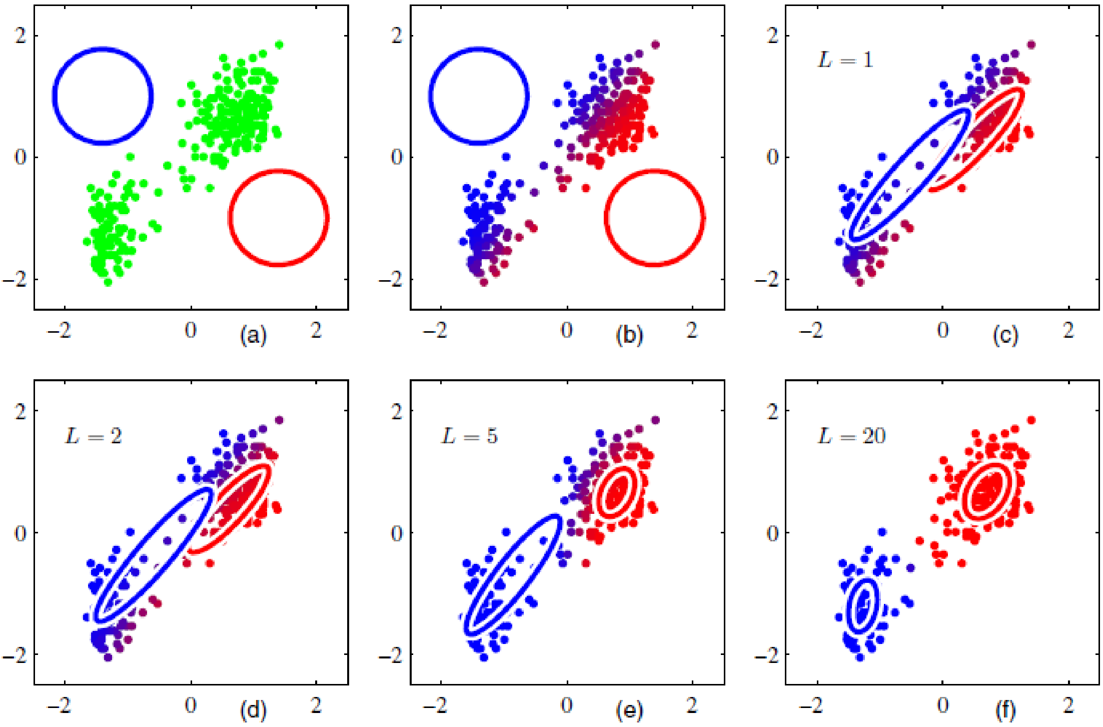


Figure 3.2: Illustration of the EM algorithm using the Old Faithful set as used for the illustration of the K-means algorithm in Figure 1.2. A mixture of two Gaussians is used. [24]

revised estimate θ^{new}. The algorithm is initialized by choosing some starting value for the parameters θ_0.

In the E step, we use the current parameter values θ^{old} to find the posterior distribution of the latent variables given by $p(Z|X, \theta^{\text{old}})$. We then use this posterior distribution to find the expectation of the complete-data log likelihood evaluated for some general parameter value θ. This expectation, denoted $\mathcal{Q}(\theta, \theta^{\text{old}})$, is given by

$$\mathcal{Q}(\theta, \theta^{\text{old}}) = \sum_{Z} p(Z|X, \theta^{\text{old}}) \ln p(X, Z|\theta) \tag{3.13}$$

In the M step, we determine the revised parameter estimate θ^{new} by maximizing this function

$$\theta^{\text{new}} = \underset{\theta}{\operatorname{argmax}}\, \mathcal{Q}(\theta, \theta^{\text{old}}) \tag{3.14}$$

Note that in the definition of $\mathcal{Q}(\theta, \theta^{\text{old}})$, the logarithm acts directly on the joint distribution $p(X, Z|\theta)$, so the corresponding M-step maximization will, by supposition, be tractable.

```
Algorithm 7: The General EM Algorithm
Data: $\mathbf{X} = \{x_{kd},\ \ k = 1, \dots, N, d = 1, \dots, D\}$ where $D$ is the dimension of the feature
      space. $Z$ the latent variables matrix. The joint distribution $p(X, Z|\theta)$ is over $X$
      and $Z$ is given, governed by parameters $\theta$.
Result: Posterior probabilities $r(z_{nk})$ and the model parameters $\theta$.
Initialization:
-Choose an initial setting for the parameters $\theta^{\mathrm{old}}$.
Learning: repeat
for $l = 1, 2, \dots$ do
    -E step: Evaluate $p(Z|X, \theta^{\mathrm{old}})$
    -M step: Evaluate $\theta^{\mathrm{new}}$ given by
        $\theta^{\mathrm{new}} = \underset{\theta}{\operatorname{argmax}}\, \mathcal{Q}(\theta, \theta^{\mathrm{old}})$
    where
        $\mathcal{Q}(\theta, \theta^{\mathrm{old}}) = \sum_{Z} p(Z|X, \theta^{\mathrm{old}}) \ln p(X, Z|\theta)$
Until convergence of either the parameters or the log likelihood. If the convergence
criterion is not satisfied then let
        $\theta^{\mathrm{old}} \leftarrow \theta^{\mathrm{new}}$
and return to the E step.
```

The general EM algorithm is summarized in Algorithm 7. It has the property that each cycle of EM will increase the incomplete-data log likelihood (unless it is already at a local maximum).

3.3 The GTM standard model

GTM is defined as a mapping from a low dimensional latent space onto the observed data space. The mapping is carried through by a set of basis functions generating a constrained mixture density distribution. It is defined as a generalized linear regression model:

$$y = y(z, W) = W\Phi(z) \tag{3.15}$$

where y is a prototype vector in the D-dimensional data space, Φ is a matrix consisting of M basis functions $(\phi_1(z), \dots, \phi_M(z))$, introducing the non-linearity, W is a $D \times M$ matrix of adaptive weights w_{dm} that defines the mapping, and z is a point in latent space.

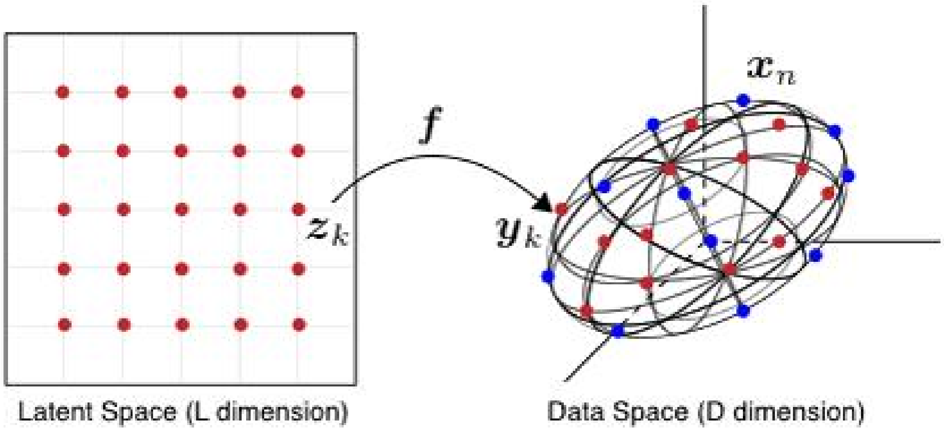


FIGURE 3.3: In order to formulate a latent variable model which is similar in spirit to the SOM, we consider a prior distribution $p(x)$ consisting of a superposition of delta functions, located at the nodes of a regular grid in latent space. Each node z_k is mapped to a corresponding point $y_k = y(z_k; W)$ in data space, and forms the center of a corresponding Gaussian distribution [25].

The standard definition of GTM considers spherically symmetric Gaussians as basis functions, defined as,

$$\phi_m(z) = \exp\left\{-\frac{\|z - \mu_m\|^2}{2\sigma^2}\right\} \tag{3.16}$$

where μ_m the centres of the basis functions and σ their common width.

Let $\mathcal{D} = (x_1, \ldots, x_N)$ the data set of N data points. A probability distribution of a data point $x_n \in \Re^D$ is then defined as an isotropic Gaussian noise distribution with a single common inverse variance β:

$$\begin{aligned} p(x_n|z, W, \beta) &= \mathcal{N}(y(z, W), \beta) \\ &= \left(\frac{\beta}{2\pi}\right)^{D/2} \exp\left\{-\frac{\beta}{2}\|x_n - y(z, W)\|^2\right\} \end{aligned} \tag{3.17}$$

Note that other models for $p(x|W, \beta)$ might also be appropriate, such as Bernoulli 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 ([25]), or even combination of these.

The distribution in x-space, for a given value of W, is then obtained by integration over the z-distribution

$$p(x|W, \beta) = \int p(x|z, W, \beta)p(z)\,dz \tag{3.18}$$

and this integral can be approximated defining $p(z)$ as a set of K equally weighted delta functions on a regular grid,

$$p(z) = \frac{1}{K} \sum_{i=1}^{K} \delta(z - z_i) \tag{3.19}$$

So, equation (3.18) becomes

$$p(x|W, \beta) = \frac{1}{K} \sum_{i=1}^{K} p(x|z_i, W, \beta) \tag{3.20}$$

For the data set $\mathcal{D}$, we can determine the parameter matrix W, and the inverse variance β, using maximum likelihood. In practice it is convenient to maximize the log likelihood, given by

$$\begin{aligned} \mathcal{L}(W, \beta) &= \ln \prod_{n=1}^{N} p(x_n|W, \beta) \\ &= \sum_{n=1}^{N} \ln \left\{ \frac{1}{K} \sum_{i=1}^{K} p(x_n|z_i, W, \beta) \right\} \end{aligned} \tag{3.21}$$

The EM Algorithm

The maximization of (3.21) can be regarded as a missing-data problem in which the identity i of the component which generated each data point x_n is unknown. The EM algorithm for this model is formulated as follows.

E-step

The posterior probabilities, or responsibilities, of each Gaussian component i for every data point x_n using Bayes' theorem are calculated in the E-step of the algorithm in this form

$$\begin{aligned} r_{in} &= p(z_i|x_n, W_{old}, \beta_{old}) \\ &= \frac{p(x_n|z_i, W_{old}, \beta_{old})}{\sum_{i'=1}^{K} p(x_n|z'_i, W_{old}, \beta_{old})} \\ &= \frac{\exp\{-\frac{\beta}{2}\|x_n - W\phi(z_i)\|^2\}}{\sum_{i'=1}^{K} \exp\{-\frac{\beta}{2}\|x_n - W\phi(z'_i)\|^2\}} \end{aligned} \tag{3.22}$$

M-step

As for the M-step, we consider the expectation of the complete-data log likelihood in the form

$$\mathbb{E}[\mathcal{L}_{comp}(W, \beta)] = \sum_{n=1}^{N}\sum_{i=1}^{K} r_{in} \ln\{p(x_n|z_i, W, \beta)\} \tag{3.23}$$

The parameters W and β are now estimated maximizing (3.23), so the weight matrix W is updated according to

$$\Phi^T G \Phi W_{new}^T = \Phi^T R X \tag{3.24}$$

where, Φ is the $K \times M$ matrix of basis functions with elements $\Phi_{ij} = \phi_j(z_i)$, R is the $K \times N$ responsibility matrix with elements r_{in}, X is the $N \times D$ matrix containing the data set, and G is a $K \times K$ diagonal matrix with elements

$$g_{ii} = \sum_{n=1}^{N} r_{in} \tag{3.25}$$

The parameter β is updated according to

$$\frac{1}{\beta_{new}} = \frac{1}{ND}\sum_{n=1}^{N}\sum_{i=1}^{K} r_{in}\|x_n - W^{new}\phi(z_i)\|^2 \tag{3.26}$$

The summary of the GTM algorithm is presented in Algorithm 8.

3.3.1 Comparison of GTM and SOM

A comparison by figures between GTM and SOM is provided in [15, 16]. They concluded that when evaluating different data sets, it is found that: The GTM method is preferable

Algorithm 8: The GTM algorithm

```
The sequence of steps for constructing a GTM model:
Generate the grid of latent points $\{z_k\}, k = 1, \ldots, K$.
Generate the grid of basis function centres $\{\mu_m\}, m = 1, \ldots, M$.
Select the basis functions width $\sigma$.
Compute the matrix of basis functions activation, $\Phi$.
Initialize $W$, randomly or using PCA.
Initialize $\beta$.
Compute $\Delta$, $\Delta_{in} = \|x_n - W\phi(z_i)\|^2$.
repeat
E-step
Compute $R$ from (3.22) using $\Delta$ and $\beta$.
Compute $G$ from (3.25) using $R$.
$W^T = (\Phi^T G \Phi)^{-1} \Phi^T R X$.
M-step
Compute $\Delta$, $\Delta_{in} = \|x_n - W\phi(z_i)\|^2$.
Update $\beta$ according to (3.26), using $R$ and $\Delta$.
until convergence.
```

when looking for representatives of the data. The GTM method yields decidedly smaller quantization errors [115, 175] and much higher topological errors [115, 173, 175] as the SOM does. Generally, the topology of both representations looks similar. In table 3.1 we show a comparison between the two methods.

3.3.2 Data visualization through GTM

As mentioned in the introduction, the GTM is embodied with clustering and visualization capabilities that are akin to those of the SOM. Data points can be summarily visualized in the low-dimensional latent space (1 or 2 dimensions) of GTM by two methods:

- The mode of the posterior distribution in the latent space:

$$z_n^{mode} = \operatorname*{argmax}_{\{z_n\}} r_{kn} \tag{3.27}$$

  Where r_{kn} is defined in 3.22. This method also provides an assignment of each data point x_n to a cluster representative z_k, it is called *posterior-mode* projection. The distribution of the responsibility over the latent space of states can also be directly visualized.

Table 3.1: Comparison between SOM and GTM

	SOM	GTM
Internal representation of manifold	Nodes $\{w_j\}_{j=1}^{C}$ in L-dimensional array, held together by neighborhood function h	Point grid $\{z_k\}_{k=1}^{K}$ in L-dimensional latent space that keeps its topology through smooth mapping f
Definition of manifold in data space	Indirectly by locations of reference vectors	Continuously by mapping f
Objective function	No	Yes: log-likelihood
Self-organization	Difficult to quantify	Smooth mapping f preserves topology
Convergence	Not guaranteed	Yes, by the EM algorithm
Smoothness of manifold	Depends on the neighborhood function	Depends on basis function parameters and prior distribution $p(x)$
Generative model	No, hence no density function	Yes
Additional parameters to select	$\mathcal{K}_{k,l}(.)$	None
Speed of training	Comparable according to Bishop et al. [25]	
Magnification factors	Approximated by the difference between reference vectors	Exactly computable anywhere

- The mean of the posterior distribution in the latent space:

$$z_n^{mean} = \sum_{k=1}^{K} r_{kn} z_k \tag{3.28}$$

known as the *posterior-mean* projection.

3.4 Collaborative Generative Topographic Mapping

As we mentioned before, according to the structure of data sets to collaborate, there are three main types of collaboration principle: horizontal, vertical and hybrid collaboration. In this chapter, we are specifically interested in horizontal and vertical collaborations, as the hybrid collaboration is not more than a combination of the both horizontal and vertical collaboration. We recall that the vertical collaboration is to collaborate the clustering results obtained from different data sets described by the same variables, but having different objects. Horizontal collaboration is more difficult since in such cases, the groups of data are described in different feature spaces: each data set is described by different variables, but has the same objects (samples) as other data sets. In this case the problem is *how to collaborate the clusters derived out of a set of classifications from different characteristics? and how to manipulate the collaborative/confidence parameter where no information is available about the distant classification?* Also, the computing of the similarity between two maps becomes impossible as they are in different feature space. In this chapter, we present a formalism to the collaboration between Generative Topographic Mappings [66, 67]. Each data set is clustered through a GTM, and to simplify the formalism, the maps built from various data sets will have the same dimensions and the same structure.

To do the task, we will use a method of penalization of EM algorithm, since GTM is based on EM, for more details see [72]. We consider the term of penalization as a collaboration term, which will penalize the distance between the prototypes of different data sets in the vertical case, while it will minimize the difference between the posterior probabilities of the latent variables in different data sets in the horizontal case. By penalizing a distance, the learning process leads to minimize this distance, that's why it is useful for doing the collaboration between different data sets. Minimizing the distance between prototypes in the vertical case means that we seek to obtain similar prototypes as clustering results. As same as for the posterior probabilities in the horizontal case.

Let $\mathcal{L}(\theta)$ the log-likelihood (Eq. 3.21 for GTM), where θ is the set of parameters ($\theta = \{W, \beta\}$ for GTM), we shall estimate θ by $\tilde{\theta}$ maximizing

$$\mathcal{L}(\theta) - \alpha J(\theta) \tag{3.29}$$

where we regard $\exp\{-\alpha J(\theta)\}$ as proportional to a prior for θ.

The EM algorithm for $\tilde{\theta}$ is then obtained by repeatedly replacing a trial estimate θ by that θ' maximizing

$$\mathcal{Q}(\theta'|\theta) - \alpha J(\theta') \tag{3.30}$$

where $\mathcal{Q}(\theta'|\theta)$ is the expectation of the complete-data log-likelihood (3.23). At convergence, we have $\theta = \tilde{\theta} = \theta'$.

This shows that we can add a collaboration term to the complete-data log-likelihood (3.23). This collaboration term corresponds for each case, either vertical or horizontal. This means that all the modifications will happen in the M-step, while the E-step will stay at it is. For more information about penalized maximum likelihood see [57, 157]. Other papers discussed the use of EM for distributed data clustering, [23, 162, 184].

We will consider that α is the coefficient of collaboration and θ is the vector of parameters of the GTM.

3.4.1 Horizontal Collaboration

In the case of horizontal collaboration, all data sets are described by the same observations but in different feature space, i.e different number of variables, $D[ii] \neq D[jj]$. The main idea of the horizontal collaboration is that the latent point z_i responsible for the model generation of the data point $x_n^{[ii]}$ in the data set $[ii]$ is also responsible for the model generation of the data point $x_n^{[jj]}$ in the data set $[jj]$.

So, in the M-step of the EM algorithm, we find $W^{[ii]}$ and $\beta^{[ii]}$ maximizing

$$\begin{aligned} \mathcal{L}^{hor}[ii] = \mathbb{E}[\mathcal{L}_{comp}(W^{[ii]}, \beta^{[ii]})] - \\ \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \alpha_{[ii]}^{[jj]} \sum_{n=1}^{N} \sum_{i=1}^{K} \frac{\beta^{[ii]}}{2} (r_{in}^{[ii]} - r_{in}^{[jj]})^2 \|x_n - W^{[ii]} \phi^{[ii]}(z_i)\|^2 \end{aligned} \tag{3.31}$$

$$\begin{aligned} \mathcal{L}^{hor}[ii] = \sum_{n=1}^{N} \sum_{i=1}^{K} \Bigg[\ln\{p(x_n | z_i, W^{[ii]}, \beta^{[ii]})\} - \\ \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \alpha_{[ii]}^{[jj]} \frac{\beta^{[ii]}}{2} \underbrace{(r_{in}^{[ii]} - r_{in}^{[jj]})^2}_{h_{in}^{[jj]}} \|x_n - W^{[ii]} \phi^{[ii]}(z_i)\|^2 \Bigg] \end{aligned} \tag{3.32}$$

Let us call $h_{in}^{[jj]} = (r_{in}^{[ii]} - r_{in}^{[jj]})^2$.

Maximization of $W^{[ii]}$

By derivation of (3.32) w.r.t $W^{[ii]}$ and putting it equal to 0, we obtain

$$\begin{aligned} \sum_{n=1}^{N} \sum_{i=1}^{K} \Bigg[r_{in}^{[ii]} \{x_n - W^{[ii]} \phi^{[ii]}(z_i)\} \phi^{[ii]^T}(z_i) - \\ \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \alpha_{[ii]}^{[jj]} h_{in}^{[jj]} \{W^{[ii]} \phi^{[ii]}(z_i) - x_n\} \phi^{[ii]^T}(z_i) \Bigg] = 0 \end{aligned} \tag{3.33}$$

And this can be conveniently be written in matrix notation in the form

$$W_{new}^{[ii]^T} = \left(\Phi^{[ii]^T} G \Phi^{[ii]} + \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \alpha_{[ii]}^{[jj]} \Phi^{[ii]^T} F^{[jj]} \Phi^{[ii]}\right)^{-1} \times \left(\Phi^{[ii]^T} R X + \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \alpha_{[ii]}^{[jj]} \Phi^{[ii]^T} H^{[jj]} X\right) \tag{3.34}$$

where, Φ is the $K \times M$ matrix of basis functions with elements $\Phi_{ij} = \phi_j(z_i)$, R is the $K \times N$ responsibility matrix with elements r_{in}, X is the $N \times D[ii]$ matrix containing the data set, G is a $K \times K$ diagonal matrix, $H^{[jj]}$ is a $K \times N$ matrix, and $F^{[jj]}$ is $K \times K$ diagonal matrix with elements

$$g_{ii} = \sum_{n=1}^{N} r_{in}^{[ii]} \tag{3.35}$$

$$h_{in}^{[jj]} = (r_{in}^{[ii]} - r_{in}^{[jj]})^2 \tag{3.36}$$

$$f_{ii}^{[jj]} = \sum_{n=1}^{N} h_{in}^{[jj]} \tag{3.37}$$

Maximization of $\beta^{[ii]}$

By derivation of (3.32) w.r.t $\beta^{[ii]}$ and putting it equal to 0, we obtain

$$\sum_{n=1}^{N} \sum_{i=1}^{K} \left[r_{in}^{[ii]} \frac{D[ii]}{\beta^{[ii]}} - r_{in}^{[ii]} \|x_n - W^{[ii]} \phi^{[ii]}(z_i)\|^2 - \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \alpha_{[ii]}^{[jj]} h_{in}^{[jj]} \|x_n - W^{[ii]} \phi^{[ii]}(z_i)\|^2 \right] = 0 \tag{3.38}$$

Then,

$$\frac{1}{\beta_{new}^{[ii]}} = \frac{1}{ND[ii]} \sum_{n=1}^{N} \sum_{i=1}^{K} \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} (r_{in}^{[ii]} + \alpha_{[ii]}^{[jj]} h_{in}^{[jj]}) \|x_n - W_{new}^{[ii]} \phi^{[ii]}(z_i)\|^2 \tag{3.39}$$

The horizontal collaboration algorithm of GTM is presented in Algorithm 9.

Algorithm 9: The horizontal GTM collaboration algorithm: *HCo-GTM*

```
1. Local step:
For each site [ii], ii= 1 to P:
we obtain for each site [ii]:
The matrix of basis functions $\Phi^{[ii]}$.
The posterior probabilities matrix $R^{[ii]}$.
The weight matrix $W^{[ii]}$.
The variance $\beta^{[ii]}$.
The matrix $G^{[ii]}$ using (3.25).
2. Horizontal collaboration step:
Objective: find a new GTM for each site [ii], ie find new $R^{[ii]}$, $W^{[ii]}$ and $\beta^{[ii]}$.
For each site [ii], ii= 1 to P :
Compute $\Delta^{[ii]}$, $\Delta_{in}^{[ii]} = \|x_n - W^{[ii]}\phi(z_i)\|^2$. ($x_n$ is a data point from the site [ii], of dimension $D[ii]$, $n = 1, \ldots, N$).
Compute $H_{[ii]}^{[jj]}$ of elements $h_{[ii]}^{[jj]}(i, n)$, for $[jj] = 1, \ldots, P$, $h_{[ii]}^{[jj]}(i, n) = (r_{in}^{[ii]} - r_{in}^{[jj]})^2$, for $i = 1, \ldots, K$ and $n = 1, \ldots, N$.
Compute $F_{[ii]}^{[jj]}$ diagonal matrices of elements $f_{[ii]}^{[jj]}(i, i) = \sum_{n=1}^{N} h_{[ii]}^{[jj]}(i, n)$ repeat
E-step
Compute $R^{[ii]}$ from (3.22) using $\Delta^{[ii]}$ and $\beta^{[ii]}$.
Compute $G^{[ii]}$ from (3.25) using $R^{[ii]}$.
M-step
Compute $W^{[ii]}$ using (3.34).
Compute $\Delta^{[ii]}$, $\Delta_{in}^{[ii]} = \|x_n - W^{[ii]}\phi^{[ii]}(z_i)\|^2$.
Update $\beta^{[ii]}$ according to ( (3.39), using $R^{[ii]}$ and $\Delta^{[ii]}$.
until convergence.
```

3.4.2 Vertical Collaboration

In the vertical case, all data sets have the same variables (same description space), but have different observations. In this case, the observations of these data sets have the same size, and the dimension of the the prototype vectors for all the GTMs will be the same, $N[ii] \neq N[jj]$. Suppose that we seek to find the GTM of the data set $[ii]$ collaborating it with P other data sets, the E-step stays as it is, in which we find the posterior probabilities

$$\begin{aligned} r_{in} &= p(z_i|x_n, W_{old}^{[ii]}, \beta_{old}^{[ii]}) \\ &= \frac{p(x_n|z_i, W_{old}^{[ii]}, \beta_{old}^{[ii]})}{\sum_{i'=1}^{K} p(x_n|z_i', W_{old}^{[ii]}, \beta_{old}^{[ii]})} \\ &= \frac{\exp\{-\frac{\beta^{[ii]}}{2}\|x_n - W^{[ii]}\phi^{[ii]}(z_i)\|^2\}}{\sum_{i'=1}^{K} \exp\{-\frac{\beta^{[ii]}}{2}\|x_n - W^{[ii]}\phi^{[ii]}(z_i')\|^2\}} \end{aligned} \tag{3.40}$$

where $n \in \{1, \ldots, N[ii]\}$.

In the M-step, we find $W^{[ii]}$ and $\beta^{[ii]}$ maximizing

$$\begin{aligned}\mathcal{L}^{ver}[ii] = \mathbb{E}[\mathcal{L}_{comp}(W^{[ii]}, \beta^{[ii]})] - \\ \sum_{\substack{jj=1\\jj\neq ii}}^{P} \alpha_{[ii]}^{[jj]} \sum_{n=1}^{N[ii]} \sum_{i=1}^{K} r_{in} \frac{\beta^{[ii]}}{2} \|W^{[ii]}\phi^{[ii]}(z_i) - W^{[jj]}\phi^{[jj]}(z_i)\|^2\end{aligned} \tag{3.41}$$

$$\begin{aligned}\mathcal{L}^{ver}[ii] = \sum_{n=1}^{N[ii]} \sum_{i=1}^{K} r_{in} \ln\{p(x_n|z_i, W^{[ii]}, \beta^{[ii]})\} - \\ \sum_{\substack{jj=1\\jj\neq ii}}^{P} \alpha_{[ii]}^{[jj]} \sum_{n=1}^{N[ii]} \sum_{i=1}^{K} r_{in} \frac{\beta^{[ii]}}{2} \|W^{[ii]}\phi^{[ii]}(z_i) - W^{[jj]}\phi^{[jj]}(z_i)\|^2\end{aligned} \tag{3.42}$$

$$\begin{aligned}\mathcal{L}^{ver}[ii] = \sum_{n=1}^{N[ii]} \sum_{i=1}^{K} \Bigg[r_{in} \ln\{p(x_n|z_i, W^{[ii]}, \beta^{[ii]})\} - \\ \sum_{\substack{jj=1\\jj\neq ii}}^{P} \alpha_{[ii]}^{[jj]} r_{in} \frac{\beta^{[ii]}}{2} \|W^{[ii]}\phi^{[ii]}(z_i) - W^{[jj]}\phi^{[jj]}(z_i)\|^2 \Bigg]\end{aligned} \tag{3.43}$$

Maximization of $W^{[ii]}$

By derivation of (3.43) w.r.t $W^{[ii]}$ and putting it equal to 0, we obtain

$$\begin{aligned}\sum_{n=1}^{N[ii]} \sum_{i=1}^{K} \Bigg[r_{in}\{x_n - W^{[ii]}\phi^{[ii]}(z_i)\}\phi^{[ii]^T}(z_i) - \\ \sum_{\substack{jj=1\\jj\neq ii}}^{P} \alpha_{[ii]}^{[jj]} r_{in}\{W^{[ii]}\phi^{[ii]}(z_i) - W^{[jj]}\phi^{[jj]}(z_i)\}\phi^{[ii]^T}(z_i) \Bigg] = 0\end{aligned} \tag{3.44}$$

This can be conveniently be written in matrix notation in the form

$$\begin{aligned}\Phi^{[ii]^T}\Big(G\Phi^{[ii]}+\sum_{\substack{jj=1\\jj\neq ii}}^{P}\alpha_{[ii]}^{[jj]}G\Phi^{[ii]}\Big)W_{new}^{[ii]^T} = \\ \Phi^{[ii]^T}RX-\Phi^{[ii]^T}\sum_{\substack{jj=1\\jj\neq ii}}^{P}\alpha_{[ii]}^{[jj]}G\Phi^{[jj]}W^{[jj]^T}\end{aligned} \tag{3.45}$$

where, Φ is the $K \times M$ matrix of basis functions with elements $\Phi_{ij} = \phi_j(z_i)$, R is the $K \times N[ii]$ responsibility matrix with elements r_{in}, X is the $N[ii] \times D$ matrix containing the data set, and G is a $K \times K$ diagonal matrix with elements

$$g_{ii} = \sum_{n=1}^{N[ii]} r_{in} \tag{3.46}$$

Then

$$\begin{aligned}W_{new}^{[ii]^T} = \Big(\Phi^{[ii]^T}\big(G\Phi^{[ii]}+\sum_{\substack{jj=1\\jj\neq ii}}^{P}\alpha_{[ii]}^{[jj]}G\Phi^{[ii]}\big)\Big)^{-1} \\ \times\left(\Phi^{[ii]^T}RX-\Phi^{[ii]^T}\sum_{[jj]=1,[jj]\neq[ii]}^{P}\alpha_{[ii]}^{[jj]}G\Phi^{[jj]}W^{[jj]^T}\right)\end{aligned} \tag{3.47}$$

Maximization of $\beta^{[ii]}$

By derivation of (3.43) w.r.t $\beta^{[ii]}$ and putting it equal to 0, we obtain

$$\begin{aligned}\sum_{n=1}^{N[ii]}\sum_{i=1}^{K}\Big[r_{in}\frac{D}{\beta^{[ii]}}-r_{in}\|x_n-W^{[ii]}\phi^{[ii]}(z_i)\|^2- \\ \sum_{\substack{jj=1\\jj\neq ii}}^{P}\alpha_{[ii]}^{[jj]}r_{in}\|W^{[ii]}\phi^{[ii]}(z_i)-W^{[jj]}\phi^{[jj]}(z_i)\|^2\Big]=0\end{aligned} \tag{3.48}$$

Therefore,

$$\frac{1}{\beta_{new}^{[ii]}} = \frac{1}{N[ii]D} \sum_{n=1}^{N[ii]} \sum_{i=1}^{K} \Bigg[r_{in} \| x_n - W_{new}^{[ii]} \phi^{[ii]}(z_i) \|^2 - \sum_{\substack{jj=1 \\ jj \neq ii}}^{P} \alpha_{[ii]}^{[jj]} r_{in} \| W_{new}^{[ii]} \phi^{[ii]}(z_i) - W^{[jj]} \phi^{[jj]}(z_i) \|^2 \Bigg] \tag{3.49}$$

The vertical collaboration algorithm of GTM is presented in Algorithm 10.

Algorithm 10: The vertical GTM collaboration algorithm: *VCo-GTM*

```
1. Local step:
For each site [ii], ii= 1 to P :
we obtain for each site [ii]:
The matrix of basis functions $\Phi^{[ii]}$.
The posterior probabilities matrix $R^{[ii]}$.
The weight matrix $W^{[ii]}$.
The variance $\beta^{[ii]}$.
The matrix $G^{[ii]}$ using (3.25).
2. Vertical collaboration step:
Objective: find a new GTM for each site [ii], ie find new $R^{[ii]}$, $W^{[ii]}$ and $\beta^{[ii]}$.
For each site [ii], ii= 1 to P :
Compute $\Delta^{[ii]}$, $\Delta_{in}^{[ii]} = \|x_n - W^{[ii]}\phi(z_i)\|^2$. ($x_n$ is a data point from the site [ii], of dimension $D$, $n = 1, \dots, N[ii]$).
Compute $\Psi_{[ii]}^{[jj]}$, for $[jj] = 1, \dots, P$, $\Psi_{[ii]}^{[jj]}(i) = \|W^{[ii]}\phi^{[ii]}(z_i) - W^{[jj]}\phi^{[jj]}(z_i)\|^2$, for $i = 1, \dots, K$.
repeat
E-step
Compute $R^{[ii]}$ from (3.22) using $\Delta^{[ii]}$ and $\beta^{[ii]}$.
Compute $G^{[ii]}$ from (3.25) using $R^{[ii]}$.
M-step
Compute $W^{[ii]}$ using (3.47).
Compute $\Delta^{[ii]}$, $\Delta_{in}^{[ii]} = \|x_n - W^{[ii]}\phi^{[ii]}(z_i)\|^2$.
Update $\beta^{[ii]}$ according to (3.49), using $R^{[ii]}$ and $\Delta^{[ii]}$.
until convergence.
```

3.5 Experimental Results

3.5.1 Results on horizontal approach

To evaluate our proposed approach we applied the algorithm on several data sets of different sizes and complexity: Waveform, Wisconsin Diagnostic Breast Cancer (wdbc), Glass and Spambase data set. All data sets are available in [7].

As criteria to validate the approach we used an internal validity index and an external one [76, 156]. External validation is based on previous knowledge about data, i.e real labels. Internal validation is based on the information intrinsic to the data alone. The internal criterion [125] we used is the Davies-Bouldin (DB) index [49]. The external criterion is the Purity index (accuracy) explained in the previous chapter, section 2.4.1.

Davies-Bouldin index

The Davies-Bouldin (DB) index [49] is an internal validity index aiming to identify sets of clusters that are compact and well separated. It is calculated as follows:

A similarity measure R_{ij} between clusters c_i and c_j is defined basing on a measure of scatter within cluster c_i, called s_i, and a separation measure between two clusters, called d_{ij}. Then R_{ij} is defined as follows:

$$R_{ij} = \frac{(s_i + s_j)}{d_{ij}}$$

Then, the DB index is defined as:

$$DB_K = \frac{1}{K} \sum_{i=1}^{K} \max_{j:i \neq j} R_{ij} \tag{3.50}$$

where K denotes the number of clusters.

The DB_K is the average similarity between each cluster $c_i, i = 1, \ldots, K$ and its most similar one. So, smaller value of DB indicates a better clustering solution, thus having minimum possible similarity with the clusters. In order to compute the DB index of the obtained results, we applied a Hierarchical Clustering [44, 99] on the prototypes matrix of the map in order to cluster the map's cells, in this way we obtain a clustering of each data set (before and after the collaboration). We performed several experiments on four data sets from the UCI Repository [7] of machine learning databases.

Data sets

- *Glass Identification*: Glass Identification data set was generated to help in criminological investigation. At the scene of the crime, the glass left can be used as evidence, but only if it is correctly identified. This data set contains 214 instances, 10 numeric attributes and class name. Each instance has one of 7 possible classes.

- The other used data sets *Waveform* and *Wisconsin Diagnostic Breast Cancer (WDBC)* are described in the previous chapter in section 2.4.2. In Figure 3.4 we visualize the *Waveform* data set with its real labels, it shows the 3 original classes partially separated.

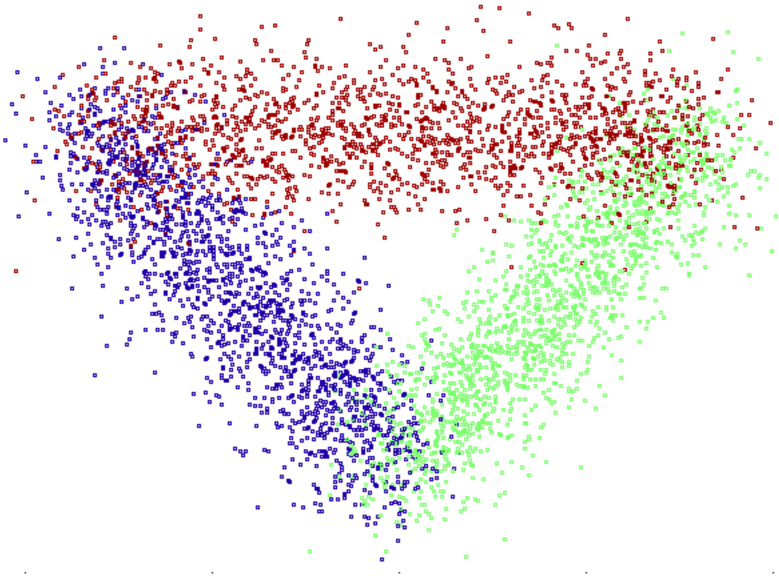


FIGURE 3.4: Waveform original data set, 3 classes of waves are shown.

In the following, we will explain the results obtained after applying The Collaborative GTM algorithms, *HCo-GTM* and *VCo-GTM* on these data sets. The data sets mentioned above are unified and need to be divided into subsets (or views) in order to have distributed data "scenarios". We divided every data set into two views (subsets) so that the algorithm operates on different features considering, however, the same set of individuals, i.e. Figure 2.3(right).

First, we applied the local phase, to obtain a GTM map for every subset. We call the resultant maps GTM_1 and GTM_2 respectively for the first and the second subset. The size of all the used maps were fixed to 10×10 except for the Glass data set whose map size is 5×5. Then we applied the collaboration phase, in which we seek a new GTM for the subset but collaborating it with the other subset. We call $\mathrm{GTM}_{2\rightarrow 1}$ the map representing subset 1 and receiving information (clustering results) from subset 2.

As described before, the waveform data set is composed from two subsets of variables: the variables from 1 to 21 representing relevant characteristics, variables from 22 to 40 are noisy. This data structure allows us to divide the data set in two views: first one containing relevant variables and the second one containing only the noisy variables. Results of local phases using GTM for these two views are presented in Figures 3.5 and 3.6 respectively for first and second view. These figures were obtained by projecting the data into two dimensional space using Principal Component Analysis [93, 144] applied on the waveform data set, but the color of the points represent the class of each object obtained using GTM and followed by a majority vote rule on the first subset (Figure 3.5) and on the second noisy set respectively (Figure 3.6).

Note, that for a better understanding of the results, the figures should be analyzed in a color mode.

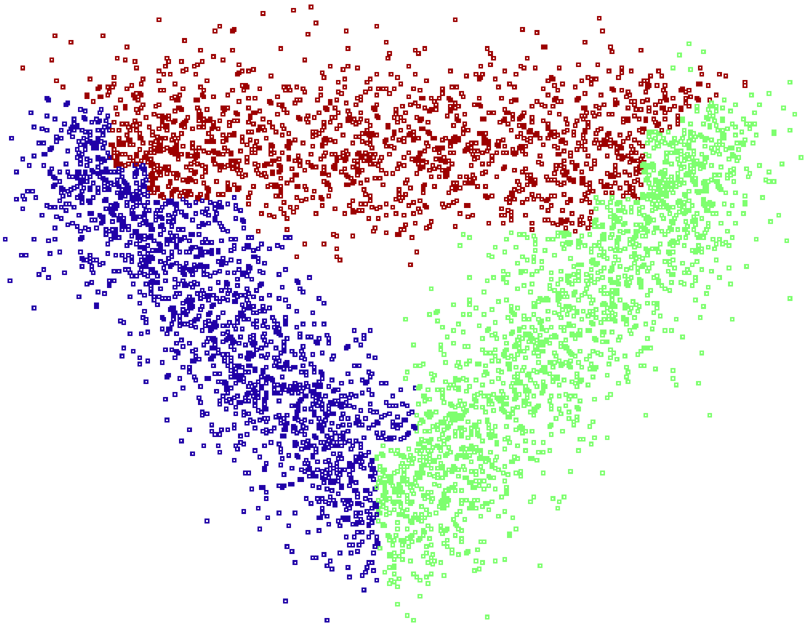


FIGURE 3.5: Waveform subset 1, relevant variables: labeling data using GTM_1

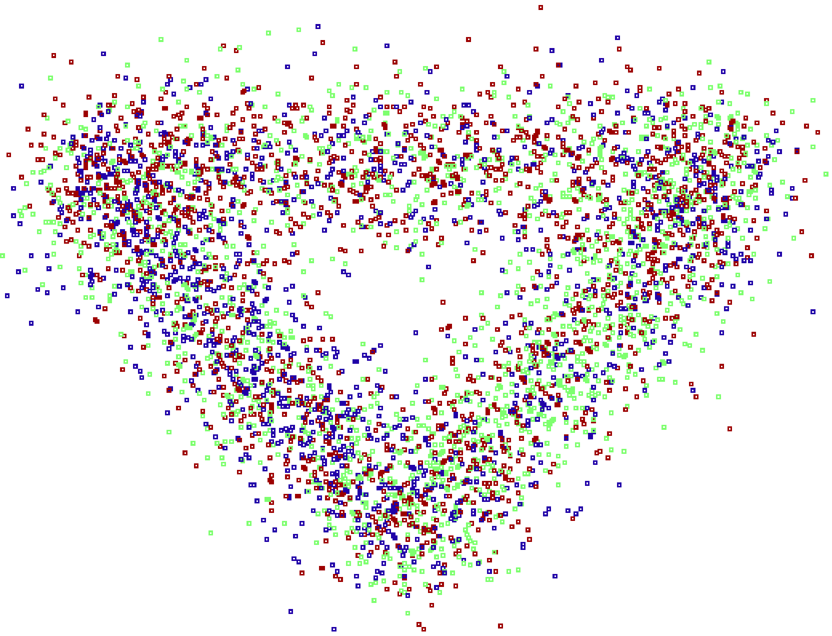


FIGURE 3.6: Waveform subset 2, noisy variables: labeling data using GTM_2

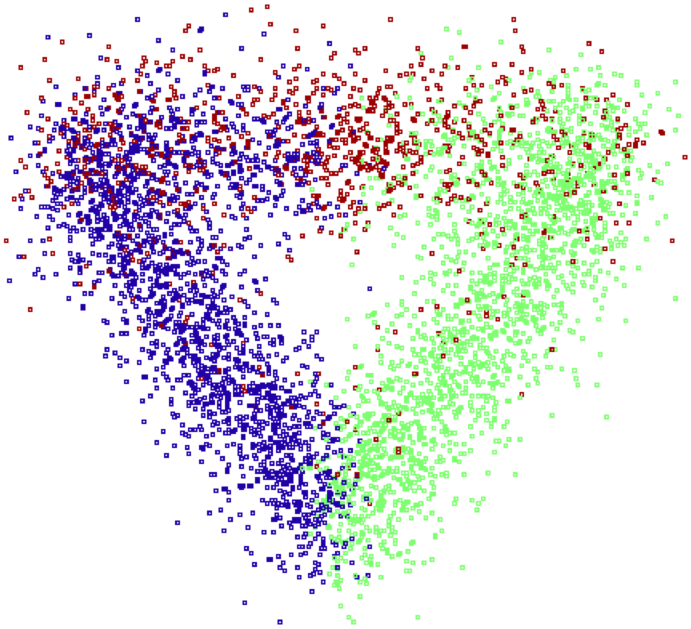


FIGURE 3.7: Waveform subset 1 after collaboration with subset 2: labeling data using $GTM_{2\rightarrow 1}$. We can see that the results sent from subset 2 reduce the quality of clustering of the subset 1. Clusters are nor more well separated.

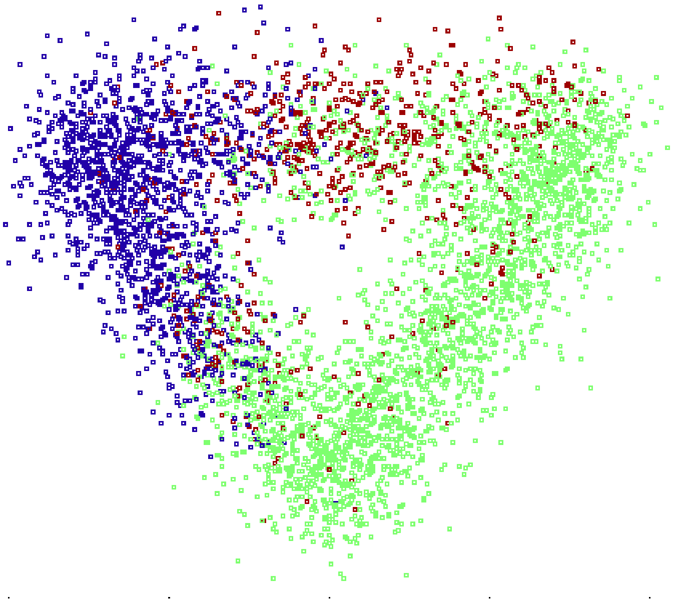


FIGURE 3.8: Waveform subset 2 after collaboration with subset 1: labeling data using $GTM_{1\rightarrow 2}$. It is obvious that the results sent from subset 1 help subset 2 to ameliorate its clustering results.

The three classes of the waveform data set are well represented and separated on Figure 3.5. While they are not in Figure 3.6 due to the variables noisiness of this view.

After applying the collaboration to exchange the clustering information between all the maps without sharing data between them, we obtained the following:

After the collaboration of the first view (relevant variables of the waveform data set) with the noisy variables clustered by GTM_2 map, the purity index decreased from 86.25% to 72.78% (Table 3.3). Figure 3.8 shows the projection of data by labeling them using the results of the collaborated map $\text{GTM}_{2\to1}$, we can see that clusters are not well separated comparing to what we have obtain before collaboration in Figure 3.5.

Contrarily, by applying the collaboration in the opposite direction, the purity index of the $\text{GTM}_{1\to2}$ map increased from 38.47% to 57.12% compared to the GTM_2. Results are shown in Figure 3.7 in which we can see that clusters are better separated now after collaboration of noisy variables with relevant variables.

Results explained above are reasonable and show the importance of collaboration. When a clustering of a set described by relevant variables collaborate with a clustering of a set containing noisy variables, the quality of clustering decreases. While in the opposite case, sending clustering results of a set described by relevant variables to the clustering of noisy set increases its quality.

As for the other data sets, we divided them all to two views. We computed the purity index and the DB index before and after collaboration and the results are shown in Table 3.3.

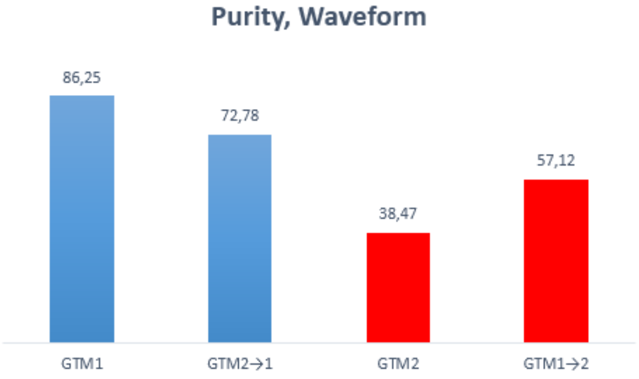


FIGURE 3.9: Comparison of the purity obtained for Waveform subsets, before and after the collaboration

In most of the cases, we remark that the purity of the map is getting higher or do not change drastically after the collaboration and strongly depends on the relevance of the collaborative map (the quality of the collaborative classification). The same analysis can be made for the DB index which decreases after the collaboration using a relevant

Table 3.2: Experimental results of the Horizontal Collaborative approach on different data sets

Data set	Map	Purity (%)	DB Index
Waveform	GTM_1	86.25	1.14
4000x21	GTM_2	38.47	3.75
4000x19	$GTM_{1\to2}$	57.12	1.73
	$GTM_{2\to1}$	72.78	1.31
Glass	GTM_1	92.32	0.74
214x5	GTM_2	64.02	1.28
214x5	$GTM_{1\to2}$	73.42	1.05
	$GTM_{2\to1}$	83.18	0.97
Wdbc	GTM_1	94.07	0.97
569x16	GTM_2	96.27	0.87
569x16	$GTM_{1\to2}$	95.88	0.9
	$GTM_{2\to1}$	94.92	0.92
SpamBase	GTM_1	80.17	1.12
4601x28	GTM_2	84.26	0.95
4601x28	$GTM_{1\to2}$	83.35	0.98
	$GTM_{2\to1}$	82.61	1.06

map. For example the DB index of the $GTM_{2\to1}$ for Glass data set obtained using the information from GTM_2 during the learning of the GTM_1 decreases from 1.28 to 0.97 (Table 3.3). This shows an amelioration of the clustering results.

This conclusion corresponds to the intuitive understanding of the principle and to the consequences of such cooperation. However, note that the goal was not to improve the clustering accuracy but to take into account the distant information and to build a new map using another view of the same data, and this procedure can decrease sometimes the quality of clustering which depends on the variables relevance of the view to collaborate.

3.5.2 Results on vertical approach

To evaluate the vertical approach we applied the algorithm on the following data sets: *Waveform, Wisconsin Diagnostic Breast Cancer (wdbc), Isolet* and *Spambase.*

The data sets are unified and need to be divided in subsets in order to have distributed data scenarios. So, we divide every data set into two subsets, having the same features, but with different observations, i.e. Figure 2.3(left).

First, we applied the local phase, to obtain a GTM map for every subset. Then we started the collaboration phase, in which we seek a new GTM for the subset but collaborating it with the other subset. We calculated the purity index of the new GTMs after collaboration, we obtained the following results:

Table 3.3: Experimental results of the vertical collaborative approach on different data sets

Data set	Map	Purity
Waveform	GTM_1	86.44
	GTM_2	86.52
	$GTM_{1\to2}$	87.16
	$GTM_{2\to1}$	87.72
Wdbc	GTM_1	96
	GTM_2	86.34
	$GTM_{1\to2}$	96.15
	$GTM_{2\to1}$	96.15
Isolet	GTM_1	87.17
	GTM_2	86.83
	$GTM_{1\to2}$	87.29
	$GTM_{2\to1}$	85.87
SpamBase	GTM_1	52.05
	GTM_2	51.68
	$GTM_{1\to2}$	52.41
	$GTM_{2\to1}$	52.17

In most of the cases, we remark that the purity of the map is getting higher after collaboration.

3.6 GTM regularized models

The optimization of the GTM model parameters through Maximum Likelihood (ML) does not take into account model complexity and consequently, the risk of data overfitting [6, 88] is elevated as Svensèn remarked in his PhD thesis [166]. An advantage of the probabilistic setting of the GTM is the possibility of introducing regularization in the mapping.

This procedure automatically regulates the level of map smoothing necessary to avoid data overfitting, resorting to either a single regularization term [27], or to multiple ones (Selective Map Smoothing : [171]). The first case entails the definition of a penalized log-likelihood of the form:

$$\mathcal{L}^{\text{pen}}(W,\beta) = \sum_{n=1}^{N} \ln\left[\frac{1}{K}\sum_{i=1}^{K} p(x_n|z_i,W,\beta)\right] - \frac{1}{2}\lambda\|w\|^2 \tag{3.51}$$

where λ is a regularization coefficient and w is a vector shaped by concatenation of the different column vectors of the weight matrix W.

A Bayesian approach to the estimation of the regularization coefficient λ as well as the inverse variance β is introduced in [128]. These parameters are usually named *hyperparameters* since they control other parameter distributions. The Bayesian approach applied to λ and β is developed in [27]. In this procedure, Bayes' theorem is used to estimate the distribution of the hyperparameters given the data points:

$$p(\lambda,\beta|X) = \frac{p(X|\lambda,\beta)p(\lambda,\beta)}{p(X)} \tag{3.52}$$

In a practical implementation, λ and β are iteratively estimated during the training of W. The use of a regularization term changes the M-step of the EM algorithm, which, for the estimation of W, yields the expression:

$$\left(\Phi^T G\Phi + \frac{\lambda}{\beta}\right)W^T = \Phi^T RX \tag{3.53}$$

where Φ, G and R are defined in section 3.3.

The second case is to use multiple regularization terms, one for each basis function. This method is named *Selective Map Smoothing* (SMS) and it was originally introduced in [171]. In SMS, there is M coefficients λ_m, where every λ_m defines a regularization coefficient for each basis function. The use of multiple regularization term changes the M-step of the EM algorithm for updating W, which will be calculated using the expression:

$$\left(\Phi^T G\Phi + \frac{1}{\beta}\Lambda\right)W^T = \Phi^T RX \tag{3.54}$$

where Λ is a square matrix $M \times M$ with elemets λ_m in the diagonal and zeros elsewhere.

But these regularization methods are not effective in all cases of overfitting. A more flexible and effective solution to avoid overfitting is presented in [139] using Variational Bayesian framework as a principled solution to deal with this problem. In next chapter, we present the Variational Bayesian framework and we make use of the Variational Bayesian GTM to apply it to Collaborative Clustering.

3.7 Conclusion

In this chapter we proposed a methodology to apply a Collaborative Clustering on distributed data using a generative model, which is the Generative Topographic Mapping (GTM). The proposed algorithm is based on GTM as a local phase of clustering, and an extension of it in the collaboration phase. A horizontal approach is adapted to the problem of collaboration of several data sets containing the same observations described by different variables. The vertical approach is adapted to the problem of collaboration of several data sets containing the same variables but with different observations.

During the collaboration phase, we do not need the share the data between sites but only the results of the distant clustering. Thus, each site uses its clustering results and the information from other clustering, which would provide a new clustering that is as close as possible to that which would be obtained if we had centralized the data sets.

We presented and approach basing on probabilistic model to cluster the data, which is the Generative Topographic Mapping. We presented the formalism of Collaborative Clustering using an adapted extension of this method. The approaches were validated on multiple data sets and the experimental results have shown promising performance.

In next chapter, we present an approach of Collaborative Clustering using Variational Bayesian GTM, which is supposed to be a solution to avoid overfitting.

Chapter 4

Collaborative Fuzzy Clustering of Variational Bayesian GTM

4.1 Bayesian modeling

Bayes' theorem, independently discovered by Reverend Thomas Bayes [10] and Pierre-Simon, marquis de Laplace [76], is one of the core tools in Statistical Machine Learning (SML) and the starting point of several methods. The theorem is expressed as follows:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} \tag{4.1}$$

This simple relationship is a powerful expression in Statistical Machine Learning (SML): it is a prescription on how to systematically update ones knowledge of a problem domain given the data observed. That means, the understanding of y after seeing data x (the posterior $p(y|x)$) is the previous knowledge of y (the prior$p(y)$) modified by how likely the observation x is under that previous model (the likelihood $p(x|y)$). The denominator, $p(x)$, is a normalizing term called the marginal likelihood or evidence. Therefore, Bayes' theorem allows one to infer knowledge that would otherwise be difficult to obtain (e.g. hidden or latent knowledge) as well as to assess underlying models.

In a Bayesian approach, inference (or learning) entails the calculation of the posterior probability density over the model parameters and the possible hidden variables. The model parameters and the hidden variables are included into the learning process as prior probability densities with further parameters called hyperparameters. Eventually, the hyperparameters can be also estimated as part of the learning process. So, if a model is built to infer knowledge from the observable data, Bayes theorem can be used as follows:

$$p(\Theta|X, \mathcal{M}) = \frac{p(X|\Theta, \mathcal{M})p(\Theta|\mathcal{M})}{p(X|\mathcal{M})} \tag{4.2}$$

where Θ represents the parameters and hidden variables of the model, X represents the observed data, and $\mathcal{M}$ embodies all the other assumptions and beliefs about the model (i.e. the structure and the hyperparameters). The prior probability, $p(\Theta|\mathcal{M})$, captures all the information known about the parameters and acts as a regularizer. Consequently, a Bayesian approach model limits overfitting in a natural way because takes into account the model complexity. The posterior probability, $p(\Theta|X, \mathcal{M})$ is a measure of what is known after the data is seen and quantifies any new knowledge acquired. The likelihood, $p(X|\Theta, \mathcal{M})$, is a measure of how well the model predicted the data. The marginal likelihood or evidence, $p(X|\mathcal{M})$, ensures the posterior is normalized and is estimated by the following expression:

$$p(X|\mathcal{M}) = \int p(X|\Theta, \mathcal{M})p(\Theta|\mathcal{M})d\Theta \tag{4.3}$$

In theory, learning the model is simply computing the posterior over the parameters. Unfortunately, the integration in Eq. 4.3 is intractable for almost all the most common models. Consequently, an increasing number of approximation methods to Bayesian inference is being defined and becoming popular among the SML community.

We saw in the previous chapter that the central task in the application of *probabilistic* models is the evaluation of the posterior distribution $p(Z|X)$ of the latent variables Z given the observed (visible) data variables X, and the evaluation of expectations computed with respect to this distribution. For instance, in the EM algorithm we need to evaluate the expectation of the complete-data log likelihood with respect to the posterior distribution of the latent variables. For many models of practical interest, it will be unfeasible to evaluate the posterior distribution or indeed to compute expectation with respect to this distribution. This could be because the dimension of the latent space is too high to work with directly or because the posterior distribution has a high complex form for which expectations are not analytically tractable. In the case of continuous variables, the required integration may not have closed-form analytical solutions, while the dimension of the space and the complexity of the integrand may prohibit numerical integration.

In such situations, we need to resort to approximation schemes, and these fall broadly into two classes, according to whether they rely on stochastic or deterministic approximations. Stochastic techniques such as Markov chain Monte Carlo [69, 94], have enabled the widespread use of Bayesian methods across many domains. They generally have the property that given infinite computational resource, they can generate exact results, and the approximation arises from the use of a finite amount of processor time. In practice, sampling methods can be computationally demanding, often limiting their use to small-scale problems. Also, it can be difficult to know whether a sampling scheme is generating independent samples from the required distribution.

In this chapter, we are interested in a new and elegant method to approximate the evidence, which is known as the *variational* framework. It manages to avoid the limitations of all other approximation methods. The variational approximation has its origin in the *calculus of variations* [28, 137] and was originally used in statistical physics [35] to model gases and systems of particles.

EM is expressed in terms of optimization in the quantity being optimized is a function. While variational inference is an optimization problem in which the quantity being optimized is a *functional*. The solution is obtained by exploring all possible input functions to find the one that maximizes, or minimizes, the functional. The functional is a function that takes a vector as its input argument, and returns a scalar. Commonly the vector space is a space of functions, thus the functional takes a function for its input argument, then it is sometimes considered a *function of a function*. Its use originates in the *calculus of variations* where one searches for a function that minimizes a certain functional.

In next section, we detail the Variational Bayesian inference. In the rest of the chapter, we present the Variational Bayesian version of GTM (VBGTM), introduced by [139], then we apply a fuzzy clustering of VBGTM in order to group the data into clusters, we call it F-VBGTM. And finally, we propose a collaborative clustering scheme based on F-VBGTM.

4.2 Variational Bayesian inference

The variational method to approximating intractable computation encompasses a whole gamut of tools for evaluating integrals and functionals. The variational method usually employed in SML uses the *mean-field* theory, very popular in statistical physics [35]. In the context of Bayesian inference, this framework is known as *variational Bayes*. The central idea of variational Bayesian inference is to introduce a set of distributions over the parameters into the marginal likelihood, in such a way that the integral Equation 4.3 becomes tractable. Variational Bayesian inference has quickly become a popular way to learn otherwise intractable models (See for reference: [3, 63, 96]).

The starting point of the variational Bayesian framework is the marginal likelihood, which, in logarithmic form, can be expressed as follows:

$$\ln p(X) = \ln \frac{p(X, \Theta)}{p(\Theta|X)} \tag{4.4}$$

where the model structure $\mathcal{M}$ is assumed to be implicit. At this point, a distribution q over the parameters Θ can be introduced, which will be henceforth called *variational distribution*, given that the log marginal likelihood does not depend on Θ:

$$\ln p(X) = \int q(\Theta) \ln \frac{p(X, \Theta)}{p(\Theta|X)} d\Theta \tag{4.5}$$

After some mathematical transformations, Eq. 4.5 can be rewritten as:

$$\begin{aligned} \ln p(X) &= \int q(\Theta) \ln \frac{p(X, \Theta)}{q(\Theta)} d\Theta + \int q(\Theta) \ln \frac{q(\Theta)}{p(\Theta|X)} d\Theta \\ &= F(q(\Theta)) + D_{KL}[q(\Theta)||p(\Theta|X)] \end{aligned} \tag{4.6}$$

where $D_{KL}[q(\Theta)||p(\Theta|X)]$ is the Kullback-Leibler (KL) divergence between the variational and the posterior distributions. Given that KL divergence is a strictly non-negative term, $F(q(\Theta))$ becomes a lower bound function on the log marginal likelihood. As a result, the convergence of the former guarantees the convergence of the latter:

$$\ln p(X) \geq F(q(\Theta)) \tag{4.7}$$

In turn, $F(q(\Theta))$ can be expressed as:

$$\begin{aligned} F(q(\Theta)) &= \int q(\Theta) \ln p(X, \Theta) d\Theta + \int q(\Theta) \ln \frac{1}{q(\Theta)} d\Theta \\ &= \mathbb{E}_{q(\Theta)}(\ln p(X, \Theta)) + \mathcal{H}(\Theta) \end{aligned} \tag{4.8}$$

where $\mathcal{H}(\Theta)$ is the entropy [105] of $q(\Theta)$. Thus, the ultimate goal in variational Bayesian inference is choosing a suitable form for the variational distribution $q(\Theta)$ in such a way that $F(q)$ can be readily evaluated and yet which is sufficiently flexible that the bound is reasonably tight. In the case of latent variable models, the latent or hidden variables Z can be easily incorporated into the variational Bayesian framework as an additional set of model parameters. In this manner, a prior distribution $p(Z)$ over the hidden variables will be also required. Taking as inspiration the expectation-maximization (EM) algorithm [50], an efficient variational Bayesian expectation-maximization (VBEM) algortihm [17] that could be applied to many SML latent variable models can be defined by assuming independent variational distributions over Z and Θ, i.e. $q(Z, \Theta) = q(Z)q(\Theta)$. Thereby, the VBEM algorithm can be derived by maximization of F as follows:

VBE-Step:

$$q(Z)^{(new)} \leftarrow \underset{q(Z)}{\operatorname{argmax}} F\big(q(Z)^{(old)}, q(\Theta)\big) \tag{4.9}$$

VBM-Step:

$$q(\Theta)^{(new)} \leftarrow \underset{q(\Theta)}{\operatorname{argmax}} F\big(q(Z)^{(new)}, q(\Theta)\big) \tag{4.10}$$

In summary, variational Bayesian inference offers an elegant framework within which inference can be performed in a closed way, and which allows efficient Bayesian inference of the model parameters and hidden variables. Next section applies these concepts to GTM to yield new powerful analytic methods.

4.2.1 Compared with EM

Variational Bayes (VB) is often compared with expectation maximization (EM). The actual numerical procedure is quite similar, in that both are alternating iterative procedures that successively converge on optimum parameter values. The initial steps to derive the respective procedures are also vaguely similar, both starting out with formulas for probability densities and both involving significant amounts of mathematical manipulations.

EM	**VB-EM**
Goal: maximize $p(\theta\|x)$ w.r.t. θ **E Step**: compute $q^{(t+1)}(Z) = p(Z\|X, \theta^{(t)})$ **M Step**: $\theta^{(t+1)} = \operatorname{argmax}_\theta \int q^{(t+1)}(Z) \ln p(X, Z, \theta) dZ$	**Goal**: lower bound $p(X)$ **VB-E Step**: compute $q^{(t+1)}(Z) \propto \exp\left[\int q^{(t+1)}(\theta) \ln p(X, Z\|\theta) d\theta\right]$ **VB-M Step**: $q^{(t+1)}(\theta) \propto \exp\left[\int q^{(t+1)}(Z) \ln p(X, Z, \theta) dZ\right]$

Table 4.1: Comparison of Variational Bayesian EM and EM for maximum a posteriori (MAP) estimation.

However, there are a number of differences, most important is what is being computed.

- EM computes point estimates of posterior distribution of those random variables that can be categorized as "parameters", but estimates of the actual posterior distributions of the latent variables. The point estimates computed are the modes of these parameters; no other information is available.

- VB, on the other hand, computes estimates of the actual posterior distribution of all variables, both parameters and latent variables. When point estimates need to be derived, generally the mean is used rather than the mode, as is normal in Bayesian inference. Simultaneous with this, it should be noted that the parameters computed in VB do not have the same significance as those in EM. EM computes optimum values of the parameters of the Bayes network itself. VB computes optimum values of the parameters of the distributions used to approximate the parameters and latent variables of the Bayes network. For example, a typical Gaussian mixture model will have parameters for the mean and variance of each of the mixture components. EM would directly estimate optimum values for these parameters. VB, however, would first fit a distribution to these parameters typically in the form of a prior distribution, e.g. a normal-scaled inverse gamma distribution and would then compute values for the parameters of this prior distribution, i.e. essentially hyperparameters. In this case, VB would compute optimum estimates of the four parameters of the normal-scaled inverse gamma distribution that describes the joint distribution of the mean and variance of the component.

A comparison of VB and EM is presented in Table 4.1.

Related methods

A related method for approximating the integrand for Bayesian learning is based on an idea known as assumed density filtering (ADF) [11, 18, 32], and is called the Expectation Propagation (EP) algorithm [135, 136]. This algorithm approximates the integrand of interest with a set of terms, and through a process of repeated deletion-inclusion of term expressions, the integrand is iteratively refined to resemble the true integrand as closely as possible. Therefore the key to the method is to use terms which can be tractably integrated. This has the same flavour as the variational Bayesian method described here, where we iteratively update the approximate posterior over a hidden state $q(Z)$ or over the parameters $q(\theta)$. The key difference between EP and VB is that in the update process (i.e. deletion inclusion) EP seeks to minimise the KL divergence which averages according to the true distribution, $D_{KL}[p(Z,\theta|X)||q(Z,\theta)]$ (which is simply a moment-matching operation for exponential family models), whereas VB seeks to minimize the KL divergence according to the approximate distribution, $D_{KL}[p(Z,\theta)||p(Z,\theta|X)]$. Therefore, EP is at least attempting to average according to the correct distribution, whereas VB has the wrong cost function at heart. However, in general the KL divergence in EP can only be minimized separately one term at a time, while the KL divergence in VB is minimized globally over all terms in the approximation. The result is that EP may still not result in representative posterior distributions (for example, see [136], figure 3.6, p. 6). Having said that, it may be that more generalized deletion-inclusion steps can be derived for EP, for example removing two or more terms at a time from the integrand, and this may alleviate some of the 'local' restrictions of the EP algorithm. As in VB, EP is constrained to use particular parametric families with a small number of moments for tractability. An example of EP used with an assumed Dirichlet density for the term expressions can be found in [135].

4.3 Variational Bayesian GTM

The original version of GTM, described in Chapter 4, used the Maximum Likelihood method to estimate its model parameters. However, as Svensèn remarked in his PhD thesis [166], this model version is too susceptible to overfit the data. A regularized version of the GTM using the evidence approximation was in fact introduced in that work. A MCMC method using Gibbs sampling [167], as well as a first approximation using a variational framework, were applied to improve the parameter estimation of the GTM model in [170]. In [139], a full variational version for the GTM was presented based on the GTM with a Gaussian process (GP) [155] prior outlined in [27], to which a Bayesian estimation of the parameters is added.

4.3.1 A Gaussian process formulation of GTM

The original formulation of GTM described in the previous chapter has a hard constraint imposed on the mapping from the latent space to the data space due to the finite number of basis functions used. An alternative approach is introduced in [27], where the regression function using basis functions is replaced by a smooth mapping carried out by a Gaussian Process (GP) prior.

So, a different formulation is assumed, a GP formulation introducing a prior multivariate Gaussian distribution over Y, defined as:

$$p(Y) = (2\pi)^{-KD/2}|\mathbf{C}|^{-D/2}\prod_{d=1}^{D}\exp\left(-\frac{1}{2}y_{(d)}^{T}\mathbf{C}^{-1}y_{(d)}\right) \tag{4.11}$$

where $y_{(d)}$ is each of the row vectors of the matrix Y, and $\mathbf{C}$ is a matrix where each of its elements is a covariance function defined as:

$$\mathbf{C}(i,j) = \mathbf{C}(z_i, z_j) = \epsilon \exp\left(- \frac{\|z_i - z_j\|^2}{2\alpha^2} \right), \quad i,j = 1, \ldots, K \tag{4.12}$$

and where hyperparameter ϵ is usually set to a value of 1. The α hyperparameter controls the flexibility of the mapping from the latent space to the data space.

Note that, this way, the likelihood takes the form:

$$p(X|Z,Y,\beta) = \left(\frac{\beta}{2\pi}\right)^{ND/2} \prod_{n=1}^{N} \prod_{k=1}^{K} \left\{ \exp\left(- \frac{\beta}{2} \|x_n - y_k\|^2 \right) \right\}^{z_{nk}} \tag{4.13}$$

Eq. 4.13 leads to the definition of a log-likelihood and parameters Y and β of this model can be optimized using the EM algorithm.

4.3.2 Bayesian GTM

The optimization of the GTM parameters through ML does not take into account model complexity and consequently, the risk of data overfitting is elevated. The Bayesian approach takes into account the complexity of the model and consequently avoids the risk of data overfitting, by treating the model parameters as hidden variables, automatically penalizing those models with more parameters than necessary.

A full Bayesian model of GTM is specified by defining priors over the hidden variables Z and the parameters which are integrated out to form the marginal likelihood as follows:

$$p(X) = \int p(X|Z,\Theta) p(Z) p(\Theta) dZ d\Theta \tag{4.14}$$

where $\Theta = (Y, \beta)$. A suitable choice for prior distributions is such as will yield a tractable variational Bayesian solution. Since z_{kn} are defined as binary values, a multinomial distribution can be chosen for Z:

$$p(Z) = \prod_{n=1}^{N} \prod_{k=1}^{K} \gamma_{kn}^{z_{kn}} \tag{4.15}$$

where γ_{kn} is an hyperparameter controlling the distribution over each of z_{kn}. The prior over parameters Θ could be defined as:

$$p(\Theta) = p(Y)p(\beta) \tag{4.16}$$

that is assuming the parameters Y and β are statistically independent. The prior $p(Y)$ was set in the previous section (Eq. 4.11). Finally, a Gamma distribution is chosen to be the prior over β:

$$p(\beta) = \Gamma(\beta|d_\beta, s_\beta) \tag{4.17}$$

where d_β and s_β are the hyperparameters of the parameter β. A graphical representation of the Bayesian GTM, including the hidden variables, parameters and hyperparameters, is shown in Fig. 4.1.

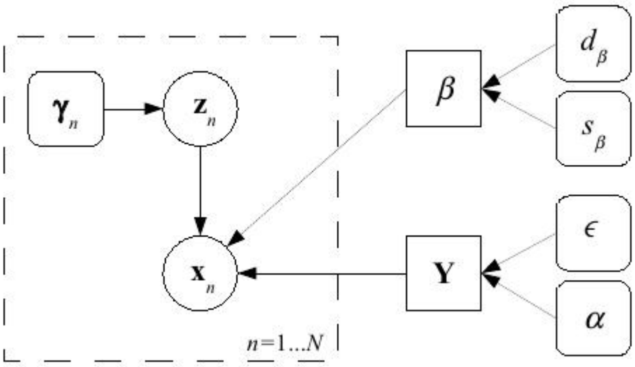


FIGURE 4.1: Graphical model representation of the Bayesian GTM. [139]

4.3.3 Variational Bayesian approach of GTM

As described in section 4.2, variational inference allows approximating the marginal log-likelihood through Jensen's inequality:

$$\ln p(X) \geq F(q(Z, \Theta)) \tag{4.18}$$

The function $F(q(Z, \Theta))$ is a lower bound such that its convergence guarantees the convergence of the marginal likelihood. The goal is choosing a suitable form for the variational distribution $F(q(Z, \Theta))$ in such way that $F(q)$ can be readily evaluated. We assume that the hidden membership variable Z and the parameters Θ are i.i.d., i.e. $q(Z, \Theta) = q(Z)q(\Theta)$. Thereby, a Variational EM algorithm can be derived [17]: a *VBE-step* as shown in Eq. 4.9 and a *VBM-step* as in Eq. 4.10.

VBEM for GTM

VBE Step

The form chosen for the variational distribution $q(Z)$ is similar of that the prior distribution $p(Z)$:

$$p(Z) = \prod_{n=1}^{N} \prod_{k=1}^{K} \tilde{\gamma}_{kn}^{z_{kn}} \tag{4.19}$$

where the variational parameter $\tilde{\gamma}_{kn}$ is given by:

$$\tilde{\gamma}_{kn} = \frac{\exp\left\{-\frac{\langle\beta\rangle}{2}\left\langle\|x_n - y_k\|^2\right\rangle\right\}}{\sum_{k'=1}^{K} \exp\left\{-\frac{\langle\beta\rangle}{2}\left\langle\|x_n - y_{k'}\|^2\right\rangle\right\}} \tag{4.20}$$

where the angled brackets $\langle . \rangle$ denote expectation with respect to the variational distribution $q(Z, \Theta)$.

VBM Step

The variational distribution $q(\Theta)$ can be approximated to the product of the variational distribution of each one of the parameters if they are assumed to be i.i.d. If so, $q(\Theta)$ is expressed as:

$$q(\Theta) = q(Y)q(\beta) \tag{4.21}$$

where the natural choices of $q(Y)$ and $q(\beta)$ are similar to the priors $p(Y)$ and $p(\beta)$ respectively. Thus,

$$q(Y) = \prod_{d=1}^{D} \mathcal{N}\big(y_{(d)} | \tilde{m}_{(d)}, \tilde{\Sigma}\big), \tag{4.22}$$

and

$$p(\beta) = \Gamma(\beta | \tilde{d}_{\beta}, \tilde{s}_{\beta}) \tag{4.23}$$

```
Algorithm 11: The VBGTM Algorithm
Data: The data set $\mathbf{X} = \{x_{kd},\ \ k = 1, \ldots, N, d = 1, \ldots, D\}$ where $D$ is the dimension
      of the feature space. The number of cells $K$. The maximum number of
      iterations $maxIter$.
Result: Responsibilities $\tilde{\gamma}_{kn}$ and the approximated variational distribution $q(\Theta)$ and
        $q(Z)$.
Initialization:
-Choose an initial setting for the parameters and the hyperparameters $\epsilon, \alpha, d, s, \Sigma, m$.
Learning: repeat
for $l = 1, 2, \ldots, maxIter$ do
    -VB-E step:
    -Compute the responsibilities $\tilde{\gamma}_{kn}$ using Eq. 4.20.
    -VB-M step:
    Compute the intermediate values
        $\left\langle \|x_n - y_k\|^2 \right\rangle = D\tilde{\Sigma}_{kk} + \|x_n - \tilde{\mathbf{m}}_k\|^2$
    -Calculation of the hyperparameters:
    - Compute $\tilde{\Sigma}$ using Eq. 4.24.
    - Compute $\tilde{\mathbf{m}}_{(d)}$ using Eq. 4.25.
    - Compute $\tilde{d}_\beta$ using Eq. 4.26.
    - Compute $\tilde{s}_\beta$ using Eq. 4.27.
    - Re-Compute
        $\langle \beta \rangle = \frac{\tilde{d}_\beta}{\tilde{s}_\beta}$
end for
```

Using these expressions in Eq. 4.10, the formulation for the variational parameters can be obtained:

$$\tilde{\Sigma} = \left(\langle \beta \rangle \sum_{n=1}^{N} \mathbf{G}_n + \mathbf{C}^{-1} \right)^{-1} \tag{4.24}$$

$$\tilde{\mathbf{m}}_{(d)} = \langle \beta \rangle \tilde{\Sigma} \sum_{n=1}^{N} x_{nd} \langle z_n \rangle \tag{4.25}$$

$$\tilde{d}_\beta = d_\beta + \frac{ND}{2} \tag{4.26}$$

$$\tilde{s}_\beta = s_\beta + \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{K} \langle z_{kn} \rangle \left\langle \|x_n - y_k\|^2 \right\rangle \tag{4.27}$$

where z_n corresponds to each column vector of Z and $\mathbf{G}_n$ is a diagonal matrix of size $K \times K$ with elements $\langle z_n \rangle$. The moments in the previous equations are defined as: $\langle z_{kn} \rangle = \tilde{\gamma}_{kn}$, $\langle \beta \rangle = \frac{\tilde{d}_\beta}{\tilde{s}_\beta}$, and $\left\langle \|x_n - y_k\|^2 \right\rangle = D\tilde{\Sigma}_{kk} + \|x_n - \tilde{\mathbf{m}}_k\|^2$.

Lower bound function

According to Eq. 4.18, the lower bound function $F(q)$ is derived from:

$$F(q) = \int q(Z)q(Y)q(\beta) \ln \frac{p(X|Z,Y,\beta)p(Z)p(Y)p(\beta)}{q(Z)q(Y)q(\beta)} dZdYd\beta \tag{4.28}$$

Integrating out, we obtain:

$$F(q) = \langle \ln p(X|Z,Y,\beta)\rangle - D_{KL}[q(Z)||p(Z)] - D_{KL}[q(Y)||p(Y)] - D_{KL}[q(\beta)||p(\beta)] \tag{4.29}$$

where the operator $D_{KL}[q||p]$ is the Kullback-Leibler (KL) divergence between q and p. This equation implies that only the computation of the KL-divergence between the variational and the prior distribution for each parameter and the expectation of the log-likelihood are required to evaluate the lower bound function. The expectation of the log-likelihood is calculated as follows:

$$\langle \ln p(X,Z,Y,\beta)\rangle = \frac{ND}{2}\langle \ln \beta\rangle - \frac{ND}{2}\ln(2\pi) - \frac{\langle\beta\rangle}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}\langle z_{kn}\rangle\langle \|x_n - y_k\|^2\rangle \tag{4.30}$$

Details of calculations can be found in [139]. The algorithm of VBGTM is presented in Algorithm 11.

4.4 Fuzzy Clustering of VBGTM

Variational Bayesian Generative Topographic Mapping (VBGTM) produces posterior probabilities for the centres of Gaussian components, but it doesn't itself provide grouping function based on the latent variables and posterior probabilities. Fuzzy C-means (FCM) algorithm has grouping function and produces posterior probabilities that indicate the membership of the data points to clusters, but it doesn't provide visualization if the data dimension is large. VBGTM is more robust than FCM when processing data set with large variations in probability distributions. So it is ideal to apply an extension to make use of VBGTM technique for fuzzy clustering, so we propose a combination of VBGTM and FCM algorithms with the goal of simultaneous visualization and clustering of data set. To this end, we use the result of VBGTM in the high-dimensional input space, that constitute a Gaussian mixture model (centres of Gaussian components) for initialization of the FCM algorithm. The result of the FCM clustering is a cluster assignment and cluster centers. The VBGTM therefore provides a low-dimensional mapping for visualization of the data and the FCM algorithm calculates the clustering. We call the extension F-VBGTM.

Therefore, the goal of F-VBGTM is to train a VBGTM model and use FCM to help VBGTM to cluster the input data into a desired number of clusters. The approach consists of four consecutive step, like follows:

1. **Train the VBGTM model**

   We train the model as described in section 4.3.3. The output of the model include the centres of the Gaussian components in the input space (Eq. 4.25), which can be used as candidate seeds for FCM. The output includes also the posterior probabilities (Eq. 4.20)

   $$\tilde{\gamma}_{kn} = p(k/x_n), \;\; 1 \leq k \leq K, \, 1 \leq n \leq N$$

   where x_n $(n = 1, \ldots, N)$ are D-dimensional data vectors.

2. **Clustering $\tilde{\boldsymbol{m}}_{(d)}$ using FCM**

   Suppose there are C clusters. After clustering, the FCM algorithm produces two outputs:

   - The cluster seeds: ν_c, $1 \leq c \leq C$.
   - The membership function for $\tilde{m}_{(d)}$: $p(\nu_c/k), \;\; 1 \leq k \leq K, \, 1 \leq c \leq C$.

3. **Bayes Theorem**

   Calculate the membership of x_n in ν_c using Bayes theorem.

   $$u_{cn} = p(\nu_c/x_n) = \sum_{k=1}^{K} p(\nu_c/k) \times p(k/x_n)$$

4. **Adjusting**

   After step 3, the data vectors x_n are assigned to clusters. As a result, the centres have to be adjusted and the distances between data vectors and cluster centres have to be calculated using the following equations

   $$\nu_c = \frac{\sum_{n=1}^{N} u_{cn} x_n}{\sum_{n=1}^{N} u_{cn}}, \;\; 1 \leq c \leq C \;\; \text{and} \;\; D_{cn} = \|x_n - \nu_c\|^2$$

The algorithm of combining FCM and VBGTM is presented in Algorithm 12.

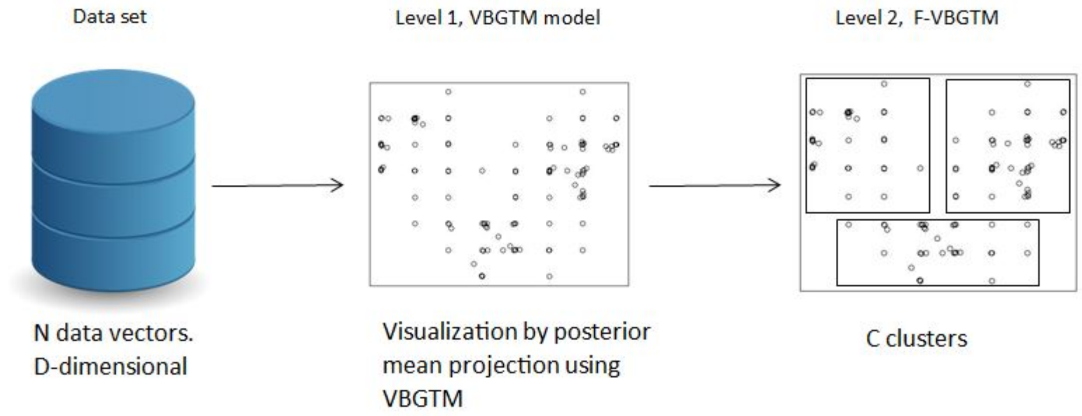


Figure 4.2: Illustration of the method, in level 1 we train a VBGTM model and visualize data in the latent space (2-dimensional) using posterior-mean projection. Then fuzzy clustering of VBGTM in level 2 to obtain C clusters.

4.4.1 Experiments of F-VBGTM

As explained in the previous section, our hybrid method permits data visualization and grouping at the same time. So we will apply it on several data sets with different size and complexity, then we will compare it with the original FCM to test its performance. The chosen data sets are: Wine, Glass, Iris (all three are available from the UCI machine learning repository [7]) and Oil flow data set (available from Netlab package [138]). We will use two internal validity indexes as a criteria to compare the two methods. Internal validation is based on the information intrinsic to the data alone, without taking into account the real labels. The chosen indexes are: Xie and Beni's index (XB) and Dunn's index (DI) calculated using Fuzzy Clustering and Data Analysis Toolbox [8] for Matlab.

Data sets

- *Wine*: This data set consists of 13 attributes and 179 cases, describing the results of the chemical analysis of samples corresponding to three types of wine.
- *Glass identification*: A data frame with 214 observation containing examples of the chemical analysis of 7 different types of glass. The problem is to forecast the type of class on basis of the chemical analysis. The study of classification of types of glass was motivated by criminological investigation. At the scene of the crime, the glass left can be used as evidence (if it is correctly identified!).

```
Algorithm 12: Fuzzy clustering of VBGTM: The F-VBGTM Algorithm
Data: The data set $\mathbf{X} = \{x_{kd},\ \ k = 1, \dots, N, d = 1, \dots, D\}$ where $D$ is the dimension
        of the feature space. The number of cells $K$.
Result: The VBGTM model of the data set & $C$ clusters with membership degrees.
Initialization:
-Choose an initial setting for the parameters and the hyperparameters of VBGTM.
Learning:
-First step: Train the VBGTM model like described in Algorithm 11.
-Second step: Clustering of $\tilde{m}_{(d)}$ using FCM.
-Choose the number of clusters C.
for $l = 1, 2, \dots$ do
    -Compute the cluster centres $\nu_c$, $1 \leq c \leq C$.
    -Compute the membership function for $\tilde{m}_{(d)}$: $p(\nu_c/k),\ \ 1 \leq k \leq K,\ 1 \leq c \leq C$.
Until convergence
for $1 \leq c \leq C$ and $1 \leq n \leq N$ do
    -Calculate the membership of $x_n$ in $\nu_c$ using Bayes theorem.
        $$u_{cn} = p(\nu_c/x_n) = \sum_{k=1}^{K} p(\nu_c/k) \times p(k/x_n)$$
    -Adjust the cluster centres
        $$\nu_c = \frac{\sum_{n=1}^{N} u_{cn} x_n}{\sum_{n=1}^{N} u_{cn}},\ \ 1 \leq c \leq C \text{ and } D_{cn} = \|x_n - \nu_c\|^2$$
```

- *Iris*: This data set consists of 50 samples from each of three species of Iris flowers (Iris setosa, Iris virginica and Iris versicolor). Four features were measured from each sample: the length and the width of the sepals and petals, in centimeters.

- *Oil*: This data set consisting of 12 attributes and 1,000 data points was artificially generated from the dynamical equations of a pipeline section carrying a mixture of oil, water and gas, which can belong to one of three equally distributed geometrical configurations. It was originally used in [25].

Cluster Validation

As criterion to validate our method and compare it with FCM we use two internal indexes, since internal criteria is used to measure the goodness of a clustering structure without referring to external information (i.e. real labels). We chose two indexes that suit the fuzzy family algorithms. The indexes are the following:

- *Xie and Beni's Index (XB)*: This index aims to quantify the ratio of the total variation within clusters and the separation of clusters [179]. A lower value of XB

indicates better clustering. It is equal to

$$XB(C) = \frac{\sum_{c=1}^{C}\sum_{n=1}^{N}(u_{cn})^m \|x_n - \nu_c\|^2}{N \times \min_{c,n} \|x_n - \nu_c\|^2} \tag{4.31}$$

Where C is the number of clusters.

- *Dunn's Index (DI)*: This index is part of a group of validity indexes including the Davies-Bouldin index, in that it is an internal evaluation scheme. The aim is to identify if clusters are compact, with a small variance between members of the cluster, and well separated. For a given assignment of clusters, a higher Dunn index indicates better clustering.

$$DI(C) = \min_{c\in C}\left(\min_{k\in C, k\neq c}\left(\frac{\min_{x\in C_c, y\in C_k} d(x,y)}{\max_{k\in C}\{\max_{x,y\in C} d(x,y)\}}\right)\right) \tag{4.32}$$

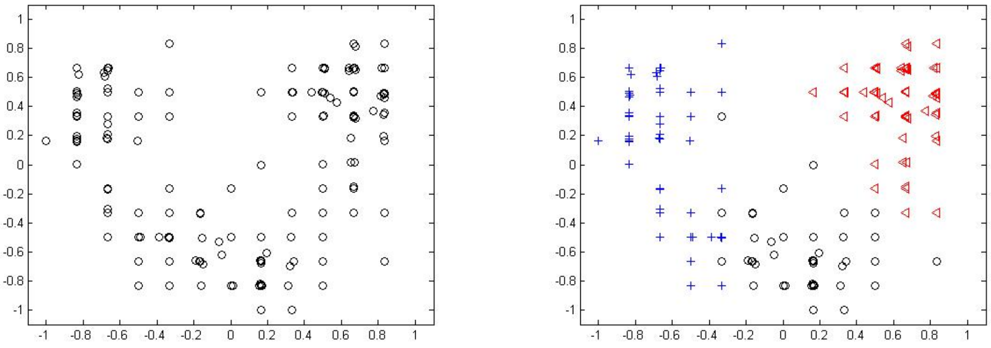


FIGURE 4.3: Visualization of the *Wine* data set using posterior mean projection, with no labels (left) and with labels obtained by applying F-VBGTM (right).

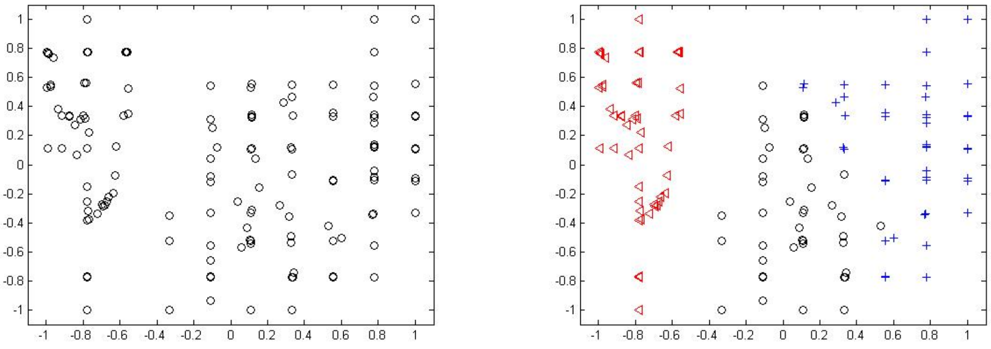
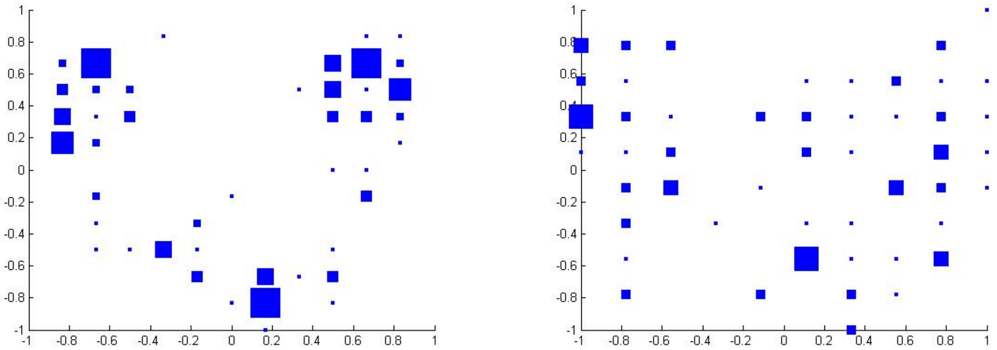


FIGURE 4.4: Visualization of the *Iris* data set using posterior mean projection, with no labels (left) and with labels obtained by applying F-VBGTM (right).



(a) *Wine*

(b) *Iris*

FIGURE 4.5: Visualization of the *Wine* and *Iris* data sets using posterior mode projection. Each square represents a latent point of size proportional to the number of data points assigned to it.

Figures 4.3 and 4.4 show the advantage of our method. First, at level 1, we train a VBGTM model to the data set, we chose 169 latent points (grid of 13×13) for the *Wine* data and 100 (10×10) latent points for *Iris* data. After training the model, we visualize data using the posterior mean projection, results are shown for these two data sets in Figures 4.3 and 4.4 (left).

Then, at level 2, for each VBGTM model we fix a number C of clusters and we apply our method F-VBGTM to group the data, we assign every data point to a cluster (by the highest membership degree) and we visualize the clusters into the same figure obtained by VBGTM's posterior mean projection. Results on data sets *Wine* and *Iris* are shown in Figures 4.3 and 4.4 (right).

Comparison

As we mentioned above, a lower value of XB indicates better clustering. Table 4.2 shows that for all the data sets, F-VBGTM has a lower XB value than FCM, which shows its better performance based on this index. As for DI index, a higher Dunn index indicates better clustering, this is the case for the data sets Iris and Glass.

4.5 Collaborative Clustering of Variational Bayesian GTM

Now let us suppose, like we previously saw in this thesis, that data is distributed among different sites. We will consider both cases, horizontal and vertical. In vertical collaboration, data sites come from different population but we have the same variables. In this

Table 4.2: Clustering evaluation using *XB* and *DI* for several data sets.

Dataset	Index	FCM	F-VBGTM
Wine	*XB*	0.785	**0.716**
	DI	0.664	0.172
Iris	*XB*	3.794	**2.856**
	DI	0.034	**0.053**
Glass	*XB*	1.131	**0.692**
	DI	0.022	**0.025**
Oil	*XB*	1.594	**1.517**
	DI	0.051	0.048

case, we can compute the distance between prototypes of different data sites and hence reduce this distance by minimizing it during the collaboration process. In horizontal collaboration, data come form the same population, i.e. same objects, but described by different variables, therefore data sites in this case have different dimension, which complicates the task of collaboration since computing the distances between prototypes of different data sites is impossible. A solution to this case is to minimize the distance between partition matrices of different sites during the collaboration process (see Chapter 1, Algorithm 3).

In this section, the objective is to apply a collaborative clustering scheme based on the Variational Bayesion GTM. To do this, we will make use of Fuzzy clustered VBGTM, described in the previous section. The algorithm will be divided to two phases, local and global (collaboration), described in the following:

Local Phase In the local phase, each site will apply the F-VBGTM on it, which will combine VBGTM and FCM to produce both clustering and visualization of the data set. The considered outputs of each site are the VBGTM model, the cluster prototypes and the computed partition matrices.

Collaboration Phase Next, we start the collaboration phase (by pairs). Each site will receive the results from a distant site, depending whether data come from same of different population, then re-compute its cluster prototypes and its partition matrix taking by consideration these distant results. If data come from the same population we call it horizontal collaboration. If data come from different population but have same variables we call it vertical collaboration.

4.5.1 Horizontal Collaboration of VBGTM: HCo-F-VBGTM

Suppose we have P data sets coming from the same population, so the number of objects in each data set is N, the number of Gaussian centres is K. Each data set is referred with an index $[ii], ii = 1, \ldots, P$. The data sets do not have the same variables, so the dimension of the feature space is different from data set to another, let $D[ii]$ be the dimension of the data set $[ii]$, so $D[ii] \neq D[jj]$ if $[ii] \neq [jj]$.

In the horizontal case, as we mentioned above, the collaboration should be done by exchanging the partition matrices between the sites, and not the prototypes because prototypes in this case do not have the same dimension and hence we cannot calculate the distances between them.

Let us suppose that, after training the VBGTM model on the sites $[ii]$ and $[jj]$ then applying the F-VBGTM algorithm on these two sites, the *partition matrix* $U[jj]$ has been sent to the data site $[ii]$. Now we can compute the new cluster prototypes and partition matrix of site $[ii]$ following Algorithm 3 described in Chapter 1.

So, considering we have two data sets $[ii]$ and $[jj]$, the approach of horizontal collaborative clustering of $[ii]$ having received results of $[jj]$ consists of 3 consecutive steps:

1. Train the VBGTM model using Algorithm 11.
2. Fuzzy clustering of the Gaussian centres $\tilde{m}_{(d)}$ like described in Algorithm 12. This step will produce the cluster centres and the partition matrices:
   - $\nu_i[ii]$ and $U[ii]$ for data set $[ii]$.
   - $\nu_i[jj]$ and $U[jj]$ for data set $[jj]$.

   The optimized objective function in this step is

$$J[ii] = \sum_{k=1}^{K} \sum_{i=1}^{C} (u_{ik}[ii])^2 \|\tilde{m}_{(k)} - \nu_i[ii]\|^2 \tag{4.33}$$

3. Collaborative fuzzy clustering of data set $[ii]$ taking into consideration the partition matrix $U[jj]$ of data set $[jj]$. The objective function to be optimized in this step is

$$J[ii, jj] = \sum_{k=1}^{K} \sum_{i=1}^{C} (u_{ik}[ii])^2 \|\tilde{m}_{(k)} - \nu_i[ii]\|^2 + \beta[ii, jj] \sum_{k=1}^{K} \sum_{i=1}^{C} (u_{ik}[ii] - u_{ik}[jj])^2 \|\tilde{m}_{(k)} - \nu_i[ii]\|^2 \tag{4.34}$$

Optimizing this objective with respect to $\nu_i[ii]$ and $u_{ik}[ii]$ leads to the following updates equations:

$$u_{ik}[ii] = \frac{1}{\sum_{s=1}^{C} \frac{\|\tilde{m}_{(k)} - \nu_i[ii]\|^2}{\|\tilde{m}_{(k)} - \nu_s[ii]\|^2}} \left[1 - \frac{1}{1+\beta[ii,jj]} \sum_{s=1}^{C} \beta[ii,jj] u_{sk}[jj] \right] + \frac{\beta[ii,jj] u_{ik}[jj]}{1+\beta[ii,jj]} \tag{4.35}$$

$$\nu_{id}[ii] = \frac{\sum_{t=1}^{K} u_{it}^2[ii] \tilde{m}_{(t)} + \beta[ii,jj] \sum_{t=1}^{K} (u_{it}[ii] - u_{it}[jj])^2 \tilde{m}_{(t)}}{\sum_{k=1}^{K} u_{ik}^2[ii] + \beta[ii,jj] \sum_{k=1}^{K} (u_{ik}[ii] - u_{ik}[jj])^2} \tag{4.36}$$

for $i = 1, \ldots, C$, $k = 1, \ldots, K$ and $d = 1, \ldots, D[ii]$.

4.5.2 Vertical Collaboration of VBGTM: VCo-F-VBGTM

In the vertical case of collaboration, data come from different population. So each data set has its own observations, described by the same variables of other data sets. If we have P data sets, $N[ii] \neq N[jj]$ and $K[ii] \neq K[jj]$ for $[ii] \neq [jj]$, $ii, jj = 1, \ldots, P$. While all data sets have the same dimension D.

The vertical collaboration should be done by exchanging the prototypes between the sites. This is feasible since prototypes of two different sites have the same dimension. By doing this, the distance between prototypes of different sites is reduced after running the algorithm, depending on the strength of the collaboration, i.e. the coefficients of collaboration β.

Let us suppose that, after training the VBGTM model on the sites $[ii]$ and $[jj]$ then applying the F-VBGTM algorithm on these two sites, the *prototypes* $\nu_c[jj]$ have been sent to the data site $[ii]$. Now we can compute the new cluster prototypes and partition matrix of site $[ii]$ following Algorithm 3 described in Chapter 1.

So, considering we have two data sets $[ii]$ and $[jj]$, the approach of vertical collaborative clustering of $[ii]$ having received results of $[jj]$ consists of 3 consecutive steps:

1. Train the VBGTM model using Algorithm 11.
2. Fuzzy clustering of the Gaussian centres $\tilde{m}_{(d)}$ like described in Algorithm 12. This step will produce the cluster centres and the partition matrices:
   - $\nu_i[ii]$ and $U[ii]$ for data set $[ii]$.

- $\nu_i[jj]$ and $U[jj]$ for data set $[jj]$.

The optimized objective function in this step is

$$J[ii] = \sum_{k=1}^{K}\sum_{i=1}^{C}(u_{ik}[ii])^2\|\tilde{m}_{(k)} - \nu_i[ii]\|^2 \tag{4.37}$$

3. Collaborative fuzzy clustering of data set $[ii]$ taking into consideration the partition matrix $U[jj]$ of data set $[jj]$. The objective function to be optimized in this step is

$$J[ii,jj] = \sum_{k=1}^{K[ii]}\sum_{i=1}^{C}(u_{ik}[ii])^2\|\tilde{m}_{(k)}-\nu_i[ii]\|^2+\beta[ii,jj]\sum_{k=1}^{K[ii]}\sum_{i=1}^{C}(u_{ik}[ii])^2\|\nu_i[ii]-\nu_i[jj]\|^2 \tag{4.38}$$

Optimizing this objective with respect to $\nu_i[ii]$ and $u_{ik}[ii]$ leads to the following updates equations:

$$u_{ik}[ii] = \frac{1}{\displaystyle\sum_{s=1}^{C}\frac{\|\tilde{m}_{(k)} - \nu_i[ii]\|^2 + \beta[ii,jj]\|\nu_i[ii] - \nu_i[jj]\|^2}{\|\tilde{m}_{(k)} - \nu_s[ii]\|^2 + \beta[ii,jj]\|\nu_s[ii] - \nu_s[jj]\|^2}} \tag{4.39}$$

$$\nu_{id}[ii] = \frac{\beta[ii,jj]\displaystyle\sum_{k=1}^{K[ii]}(u_{ik}[ii])^2\nu_{id}[jj] - 2\sum_{k=1}^{K[ii]}(u_{ik}[ii])^2\tilde{m}_{(k)}}{(\beta[ii,jj]-1)\displaystyle\sum_{k=1}^{K[ii]}(u_{ik}[ii])^2} \tag{4.40}$$

for $i = 1,\ldots,C$, $k = 1,\ldots,K[ii]$ and $d = 1,\ldots,D$.

4.5.3 Experiments

To show the effect of the collaborative clustering on the data sites we test the algorithm, like previous chapter, on a split *Waveform* data set. We choose this data set because of its structure, it contains 21 relevant variables and 19 noisy variables, and 5000 observations. We split the data set into two subsets, the first subset contains the relevant variables (of dimension 5000×21) and the second subset contains the noisy variables (of dimension 5000×19). By doing this, we get distributed data on two sites, data coming from same population. The first has good clustering results since data are are separable when they are described by the relevant variable. The second site has bad clustering results since its variables are noisy.

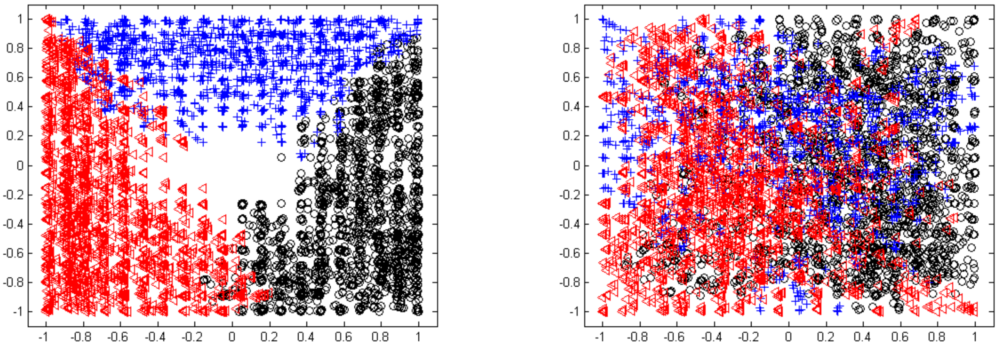


FIGURE 4.6: Visualization of the two subsets of the Waveform data set using posterior mean projection, with labels obtained using F-VBGTM before collaboration. We can see good clustering results on the first subset (left). The results of clustering on the second subset of noisy variables are bad (right).

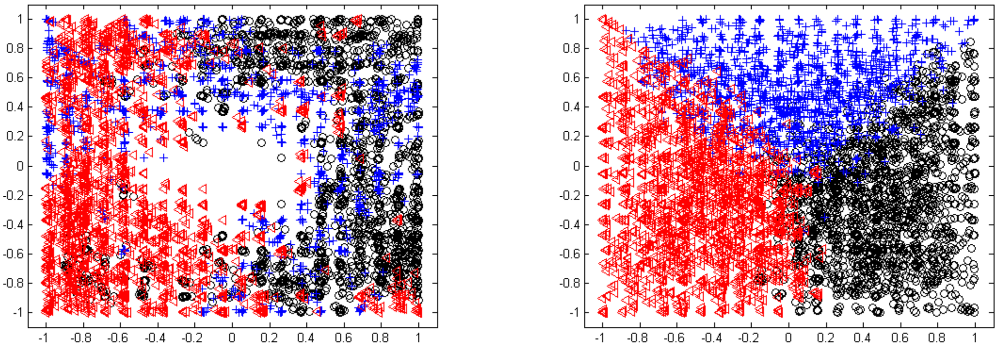


FIGURE 4.7: Effect of the Collaborative Clustering. When the second subset (bad) sends results to first subset (good), clusters structure is deteriorated (left figure). While clustering is ameliorated when the first subset collaborates with the second subset (right figure).

The clustering results by F-VBGTM on the two subsets of the Waveform data set before collaboration are shown in Figure 4.6.

Now moving to the collaboration phase, the way we divided the data set, i.e. two subsets with same observations and different variables, permits us to apply the horizontal collaborative clustering (see section 4.5.1). We expect that when we send the partition matrix of the first subset (relevant) to the second subset (noisy) and then we compute the new prototypes and partition matrix, the results must be better comparing to what we got before the collaboration, because we send results from a good clustered data site to a bad clustered one. The inverse is also true, i.e. when we send the results of the second subset to the first one, this will deteriorate the results of the first subset. The results of the clustering after the collaboration are shown in Figure 4.7.

4.6 Collaborative Clustering of VBGTM with variable interaction level

So far, we considered that the coefficient of collaboration β, also called *interaction level*, is fixed by the user before the collaboration process

So far, the collaborative clustering algorithm described in the previous section requires that the user sets the interaction level, β, which is used for all pairs of data sites and kept constant (fixed) during the collaboration stages. In this section, we present some methods that are capable of automatically estimating the interaction levels from data. In some cases, these methods compute a corresponding interaction level before the collaboration process, i.e. basing on the clustering results. In other cases, they dynamically adjusts, during the collaboration process, a particular β value for each pair of data sites.

In brief, if the cluster structures in two data sites are very different from each other, then β should be low, leading to a weak collaboration between the two data sites under consideration.

On the other hand, a pair of similar data sites should lead to a hight value for β, which suggests that a strong collaboration between them can be accomplished.

4.6.1 In the horizontal case

4.6.1.1 PSO

In this approach, we focus on finding similar cluster compositions across companies. It consists on learning the optimal collaboration matrix during the clustering analysis by applying the evolutionary optimization technique of Particle Swarm Optimization (PSO).

PSO is an evolutionary optimization technique developed by Kennedy et al. [110], inspired by the swarming behaviour of bird flocks and fish schools.

The optimization algorithm first initializes Z particles x_z, each particle representing a possible solution to the optimization problem. Next, the particles start to fly through the solution space and at each time interval t, the fitness of the solution is evaluated by means of a fitness function. During their flight, each particle remembers its own best position p_z. The direction of a particle in the solution space is influenced by the particle's current location $x_z(t)$, the particle's current velocity $v_z(t)$, the particle's own best position p_z and the global best position among all particles p_g. The particle's new position $x_z(t+1)$ is calculated by Eq. 4.41 and Eq. 4.41.

$$v_z(t+1) = wv_t(t) + c_1 r_1(p_z - x_z(t)) + c_2 r_2(p_g - x_z(t)) \tag{4.41}$$

$$x_z(t+1) = x_z(t) + v_z(t+1) \tag{4.42}$$

where w is the inertia weight and c_1, c_2 are the acceleration constants drawing the particle toward the local and global best locations, respectively. The stochastic component of the PSO meta-heuristic is given by r_1 and r_2, which stand for two uniformly distributed random numbers. All particles keep moving in the solution space until some criterion is met. The global best position at the end is the solution to the optimization problem. For a broader insight about this widespread optimization technique, refer to [33].

In our particular case, a single particle will represent a collaboration coefficient and the flight of the particles represents the search for a collaboration matrix which optimizes the similarity of the cluster compositions across data locations. To achieve such optimization, we formulate an appropriate fitness function which represents the dissimilarity in cluster composition across data locations. The goal of the PSO algorithm will be to minimize this function. We redefine a cluster $C_i[ii]$ as a set of membership degrees $\{u_{1i}[ii], \ldots, u_{Ni}[ii]\}$, where N is the number of data (remind that we are considering the horizontal case). Now we can express the dissimilarity between cluster i from data site $[ii]$ and cluster j from data site $[jj]$ as follows:

$$d(C_i[ii], C_j[jj]) = \frac{1}{N} \sum_{k=1}^{N} |u_{ik}[ii] - u_{jk}[jj]| \tag{4.43}$$

This dissimilarity measure will become zero, which is the lower bound, when all patterns belong to both clusters with the same degree. On the other hand, it will become 1, which is the upper bound, when both clusters are crisp and don't have any pattern in common. Furthermore, this measure is also symmetric. Next, to measure the dissimilarity between the entire cluster solution of data site $[ii]$ and data site $[jj]$, we compare each cluster of data site $[ii]$ with each cluster of data site $[jj]$ and only consider the smallest dissimilarity for each cluster (cf. Eq. 4.44). Note that this measure equals to 0 when both cluster solutions are identical.

$$D[ii, jj] = \frac{1}{C} \sum_{i=1}^{C} \min_{j=1}^{C} [d(C_i[ii], C_j[jj])] \tag{4.44}$$

If we are dealing with P data sites at one time, the final fitness measure, which we will term as ρ, can be envisioned as the mean dissimilarity of the cluster solutions across all data sites.

$$\rho = \frac{2}{P(P-1)} \sum_{ii=1}^{P} \sum_{jj>1}^{P} D[ii, jj] \tag{4.45}$$

Given this fitness measure, we can use PSO to determine the optimal set of collaboration links. Aside from the data locations, which we will call data nodes, we will need a computing location which performs the PSO algorithm. This location will act as the coordination node. It should be noted that the coordination node can be the same physical location as a specific data node, but this isn't necessary. Algorithm 13 shows how the collaborative clustering scheme and the particle swarm optimization can be integrated to automate the determination of the collaboration links.

Algorithm 13: The Collaborative clustering Algorithm using PSO for the coefficients of collaboration.

```
Initialization:
-Initialize $Z$ particles $x_z$.
Learning: repeat
for each particle $x_z$ do
    -Perform The Collaborative clustering algorithm with the collaboration coefficient
    represented by $x_z$ (data nodes).
    -Send the partition matrices to the coordination node.
    -Calculate $D[ii, jj]$ using Eq. 4.44 (coordination node).
    -Calculate the new position $x_z(t+1)$ (coordination node).
    -Update $p_z$ (coordination node).
end for
Until some termination criterion is reached (coordination node).
-Send the optimal collaboration links to the data nodes.
-Perform The algorithm of collaboration with the optimal collaboration coefficients
(data nodes).
```

4.6.1.2 Auto weighted Collaboration

In this method, weights (coefficients of collaboration) are automatically determined according to the considerations of partition matrices. Even though it is difficult to directly give the weights, it is easy to give some principle for determining the weights. For example, we may expect to think a lot of the influence of data sets that have similar clustering result to that of the reference data set, we may also expect to think a lot of the influence of data sets that have different clustering result, or expect to treat the external data sets equally without discrimination. These expectations may be used as principles for determining weights, while they are easy to be given out. In this section, we will give some approaches to automatically calculate the weights according to these principles.

Measure of Partition Similarity

First, a similarity measure to measure the similarity between the clustering results of the reference data set and those of external data sets is given. If the similarity between two clustering results can be calculated, then the weights can be determined.

Let U and V two partition matrices on C clusters. We define two similarity measures of partition matrices:

$$S_1(U,V) = \frac{\sum_{i=1}^{C} \max_{j=1}^{C} \left\{ \sum_{k=1}^{N} (u_{ik} \wedge v_{jk}) \right\}}{\sum_{i=1}^{C} \min_{j=1}^{C} \left\{ \sum_{k=1}^{N} (u_{ik} \vee v_{jk}) \right\}} \tag{4.46}$$

$$S_2(U,V) = \frac{\sum_{i=1}^{C} \max_{j=1}^{C} \left\{ \sum_{k=1}^{N} (u_{ik} \wedge v_{jk}) \right\}}{N} \tag{4.47}$$

It is easy to verify that $S_1(U,V)$ and $S_2(U,V)$ verify both the conditions of similarity measures. Examples on these similarity measures are presented in [181].

There is two approaches in which we can make use of the similarity measures. The first is an encouragement approach and the second is a penalty approach.

Encouragement approach

In this approach, an encouragement principle is assumed: *The more similar the partition matrix to the reference partition matrix, the larger the effect of the external data set.*

In terms of this principle, we can choose a suitable similarity measure of partition matrices, and calculate the similarity of the external partition matrices to the reference partition matrix. The following three approaches to calculate weights are all encouragement approaches.

Suppose $r_[ii]$ is the similarity of partition matrix $U[ii]$ to the reference partition matrix $U, ii = 1, \ldots, P$, which is calculated under a similarity measure. Then the weight $\beta[ii]$

expressing the degree of influence that $U[ii]$ exerts onto $U, ii = 1, \ldots, P$ may be given by the following three approaches:

$$\text{Approach 1} : \beta[ii] = \frac{r[ii]}{\sum_{ii=1}^{P} r[ii]} \tag{4.48}$$

$$\text{Approach 2} : \beta[ii] = \frac{r[ii]}{\min_{ii=1}^{P} r[ii]} \tag{4.49}$$

$$\text{Approach 3} : \beta[ii] = \frac{r[ii]}{\max_{ii=1}^{P} r[ii]} \tag{4.50}$$

Penalty approach

Similar to the encouragement approach, similarity between reference partition matrix and external ones should first be calculated under some similarity measure. But in contrast to the encouragement approach, weights are calculated by such a principle: *the more similar the partition matrix to the reference partition matrix, the smaller the corresponding weight.*

The following approaches are for determining weights in penalty way:

$$\text{Approach 1} : \beta[ii] = \frac{(1 - r[ii])}{\sum_{ii=1}^{P}(1 - r[ii])} \tag{4.51}$$

$$\text{Approach 2} : \beta[ii] = \frac{(1 - r[ii])}{\min_{ii=1}^{P}(1 - r[ii])} \tag{4.52}$$

$$\text{Approach 3} : \beta[ii] = \frac{(1 - r[ii])}{\max_{ii=1}^{P}(1 - r[ii])} \tag{4.53}$$

4.6.2 In the Vertical Case

Let us consider two data site $[ii]$ and $[jj]$ and their respective prototypes $\nu_i[ii]$ and $\nu_i[ii]$, $i = 1, \ldots, C$. Let us suppose that the prototypes $\nu_i[ii]$ have been sent to the data site $[ii]$.

First, we compute the induced partition matrix $\tilde{U}[ii, jj]$. This matrix is computed using the prototypes communicated by the jjth data site and the objects from the iith data site using the update equation of the FCM algorithm:

$$\tilde{u}_{ij}[ii, jj] = \left[\sum_{l=1}^{C} \left(\frac{\|\tilde{m}_j[ii] - \nu_i[jj]\|}{\|\tilde{m}_j[ii] - \nu_l[jj]\|} \right)^2 \right]^{-1} \tag{4.54}$$

From this matrix, we can compute the value of the induced objective function:

$$\tilde{Q}[ii,jj] = \sum_{k=1}^{K[ii]} \sum_{i=1}^{C} \tilde{u}_{ik}^2[ii,jj] \|\tilde{m}_k[ii] - \nu_i[jj]\|^2 \tag{4.55}$$

where $\tilde{u}_{ik}^2[ii,jj]$ is an element of the induced partition matrix $\tilde{U}[ii,jj]$ and $\tilde{m}_k[ii]$ is a centre of Gaussian belonging the site $[ii]$ computed in the VBGTM model.

The objective function for site $[ii]$ is denoted by $Q[ii]$ and calculated like in Eq. 4.37. Then the interaction level $\beta[ii,jj]$ between two data sites $[ii]$ and $[jj]$, at a given collaboration stage, is defined as in [43]:

$$\beta[ii,jj] = \min \left\{ 1, \frac{Q[ii]}{\tilde{Q}[ii,jj]} \right\} \tag{4.56}$$

The value for $\tilde{Q}[ii,jj]$ is typically greater than $Q[ii]$ because it is computed based on the prototypes sent from the jjth data site, whereas $Q[ii]$ has been optimized for the iith data site itself. Thus, the values for $\beta[ii,jj]$ close to 0 suggest that the data sites are very different and, so, this implies that the collaboration should be low. Contrarily, if one assumes that the prototypes standalone convey all the information that describes the cluster structures, then it is legitimate to consider that the partitions in data sites $[ii]$ and $[jj]$ will be very similar when $\beta[ii,jj]$ is close to 1, and vice-versa. In this case, the collaboration level should be high.

The idea captured by Eq. 4.56 has been incorporated into the collaborative algorithm in order to dynamically adjust the interaction level between every pair of data sites at every collaboration stage. Now, $u_{ik}[ii]$ and $\nu_{id}[ii]$ are computed like follows:

$$u_{ik}[ii] = \frac{\beta[ii,jj]\tilde{u}_{ik}[ii,jj]}{1+\beta[ii,jj]} + \frac{1}{\displaystyle\sum_{s=1}^{C} \frac{\|\tilde{m}_{(k)} - \nu_i[ii]\|^2}{\|\tilde{m}_{(k)} - \nu_s[ii]\|^2}} \left[1 - \frac{1}{1+\beta[ii,jj]} \sum_{s=1}^{C} \beta[ii,jj]\tilde{u}_{sk}[ii,jj] \right] \tag{4.57}$$

$$\nu_{id}[ii] = \frac{\displaystyle\sum_{t=1}^{K[ii]} u_{it}^2[ii]\tilde{m}_{(t)} + \beta[ii,jj] \sum_{t=1}^{K[ii]} (u_{it}[ii] - \tilde{u}_{it}[ii,jj])^2 \tilde{m}_{(t)}}{\displaystyle\sum_{k=1}^{K[ii]} u_{ik}^2[ii] + \beta[ii,jj] \sum_{k=1}^{K[ii]} (u_{ik}[ii] - \tilde{u}_{ik}[ii,jj])^2} \tag{4.58}$$

Following this idea, a more computationally efficient algorithm can be developed. Such an algorithm, estimates values of $\beta[ii, jj]$ only once before the collaboration process takes place and keep them fixed throughout the collaboration stages. When using these two algorithms (as well as the original collaborative alogorithm) the user implicitly assumes that the data from different sites comes from different populations, because he or she expects that only data sites that are similar (in some relative sense) should take advantage of the collaborative clustering process. In other words, it is assumed, *a priori*, that for very different data sites there is no reason to incorporate the shared information provided by their respective prototypes into the collaborative process.

4.7 Summary

In this chapter, we presented the variational approximation principle, and the variational Bayesian version of GTM (VBGTM). Its major advantage is to control data overfitting. Then we developed a new method for fuzzy clustering by combining the Variational Bayesian Generative Topographic Mapping VBGTM and the Fuzzy C-means FCM. Then we used FCM to produce a desired number of clusters based on the output of VBGTM. FCM is a tool for fuzzy clustering. VBGTM is mostly used for data visualization of the distribution of data sets. By combining the two algorithms, we developed a method than can do data visualization and grouping at the same time. Compared to the combination of K-means and SOM, the method proposed in this paper provides membership functions to indicate the likelihood of a data item belonging to a cluster. The membership function is capable of revealing valuable information when performing clustering in applications such as customers segmentation. Experiments showed that the proposed F-VBGTM method consistently performed better than the FCM algorithm.

In the rest of the chapter, we made use of the proposed algorithm F-VBGTM to apply it in the case of distributed data, more specifically in a collaborative clustering scheme. We presented both approaches of it, horizontal and vertical. An example of the effect of the collaboration is presented. Then we presented some methods for calculating the collaboration links during the collaboration stage. We can apply these methods in order to estimate the confidence between different data sites.

Summary and Conclusion

The main thrust of this thesis is to formulate Collaborative Clustering schemes based on topological methods, such as Self-Organizing Map (SOM), Generative Topographic Mappings (GTM) and Variational Bayesian Generative Topographic Mappings (VBGTM). Collaborative Clustering intend to reveal the overall structure of distributed data (i.e. data residing at different repositories) but, at the same time, complying with the restrictions preventing data sharing. The fundamental concept of Collaborative Clustering is: *the clustering algorithms operate locally (namely, on individual data sets) but collaborate by exchanging information about their findings.* The strength of collaboration is precised by a parameter called coefficient of collaboration, more it is high more the collaboration is strong.

The main novel contributions of this thesis are briefly summarized next. This is followed by an outline of some possible future directions of research stemming from our investigation.

Summary of Contributions

Self-organizing maps (SOMs) are a data visualization technique which reduce the dimensions of data through the use of self-organizing neural networks. The problem that data visualization attempts to solve is that humans simply cannot visualize high dimensional data as is so techniques are created to help us understand this high dimensional data. Chapter 2 formulates a horizontal and a vertical collaborative clustering schemes based on SOM as local step of clustering. In the collaboration phase, data sites share the parameters obtained in the local phase, which are for SOM the prototypes and the neighborhood function values. Another contribution in this chapter is the automatic estimation of the coefficient of collaboration. This parameter quantifies the confidence between data sites, precising the strength of contribution of each site in the consensus building procedure.

We presented in chapter 3 a collaborative clustering scheme based on a generative model. The used generative model is GTM, which was proposed as a probabilistic counterpart of SOM. GTM was defined to retain all the useful properties of SOM, such as the simultaneous clustering and visualization of multivariate data, while eluding most of its limitations through a fully probabilistic formulation. GTM is based on the EM algorithm. To collaborate using GTM we modified the M-step by adding a collaboration term to the complete log-likelihood function. This led to a modification in the update formulas of the GTM parameters (the weight matrix and the variance). The collaboration term depends whether the scenario of collaboration is horizontal or vertical. But the risk of overfitting is elevated because the optimization of the GTM models through EM does not take into account the model complexity. A solution for this is presented in the next chapter.

Chapter 4 presents a collaborative clustering algorithm based on variational Bayesian model. This model was presented as a solution to avoid overfitting of GTM. Is is called VBGTM. We start the chapter by introducing the variational Bayesian inference. Then we introduce VBGTM and its VB-EM algorithm. Next, we propose an extension to make use of VBGTM for fuzzy clustering, the extension applies FCM on the centres of Gaussian components obtained by VBGTM, then assign data to the clusters by applying Bayes theorem on the posterior probabilities obtained by VBGTM and the membership values obtained by FCM. By combining the two algorithms, we develop a method that can do data visualization and grouping at the same time. After all, we propose a collaborative clustering schemes using this extension. Experiments show the advantage of the proposed method. We propose also some methods of collaborative clustering using VBGTM with variable interaction level.

Future Directions

This thesis creates some clear opening for future lines of research. We consider an important open perspective:

The collaborative clustering schemes described in this thesis consider the same clustering algorithm applied in the local phase, and extend it to be applied in the distributed data case. An important question to be asked is: what if the different data sources uses different clustering algorithms? Or what if the same clustering algorithm is not suitable for all the sources? We consider an important open perspective in this case, in which we seek an algorithm taking into consideration different clustering algorithms and collaborate them.

Also, clustering algorithms in this thesis pertain to off-line (or batch) processing, in which the clustering process repeatedly sweeps through a set of data samples in an attempt to capture its underlying structure in a compact and efficient way. However, many recent applications require that the clustering algorithm be online, or incremental, in that there is no a priori set of samples to process but rather samples are provided one iteration at a time. Accordingly, the clustering algorithm is expected to gradually improve its prototype (or centroid) constructs. Several problems emerge in this context, particularly relating to the stability of the process and its speed of convergence. So what if we require a clustering algorithm applied on distributed data streams?

Other perspectives are also considered:

- Combine horizontal and vertical approaches to get a new hybrid collaboration approach.
- Fusion the obtained maps (SOM, GTM or VBGTM) after the collaboration to construct a clustering "consensus" for all the sites.
- Test the impact of diversity between different models on the quality of the collaboration.
- Use the diversity between different models to guide the collaboration to be selective.
- Integrating background knowledge into collaborative clustering.
- Test the relation between collaborative clustering and transfer learning.

We are working on these perspectives in the ANR project COCLICO[1], in collaboration with: ICUBE[2], AgroParisTech[3], LIVE[4] and UMR Espace Dev[5].

[1] COllaboration, CLassification, Incrmentalit et COnnaissances. http://icube-coclico.unistra.fr/

[2] Laboratoire des sciences de l'Ingnieur, de l'Informatique et de l'Imagerie, Université de Strasbourg

[3] Institut des sciences et industries du vivant et de l'environnement

[4] Laboratoire Image, Ville, Environnement, Université de Strasbourg

[5] L'espace au service du dveloppement, Université Montpellier 2

Conclusion et perspectives

Dans cette thèse nous avons présenté plusieurs nouveaux algorithmes de Clustering Collaboratif basés sur des méthodes à base de prototypes. Les méthodes utilisées sont les cartes auto-organisatrices (SOM), les cartes topographiques génératives (GTM), et les GTM Variationnelles Bayésiennes (VBGTM). Une caractéristique commune entre ces trois méthodes est la visualisation des donnés de grande dimension. Ayant une collection de bases de données distribuées sur plusieurs sites différents, le clustering collaboratif consiste à partitionner chacune de ces bases en considérant les données locales et les classifications obtenues par les autres sites pour améliorer/enrichir la classification locale, sans toutefois avoir recours au partage des données entre les différents centres. La force de la collaboration est précisée par un paramètre appelé coefficient de collaboration, ou confiance, plus sa valeur est grande plus la collaboration est forte et pertinente.

La première contribution dans cette thèse est un algorithme de clustering collaboratif basé sur SOM. Deux approches de collaboration sont proposées, l'approche horizontale où les sites ont les mêmes observations mais différentes variables, et l'approche verticale où les sites ont les mêmes variables mais différentes observations. Dans la phase de collaboration, les sites partagent les résultats obtenus lors de la phase de classification locale, qui sont dans le cas de SOM les prototypes et les valeurs de la fonction de voisinage.

Pour le coefficient de collaboration, nous avons proposé un algorithme permettant de l'estimer automatiquement en ajoutant une étape à la phase de collaboration. En estimant ce paramètre, les sites peuvent automatiquement préciser la confiance qu'ils font aux autres sites, par suite choisir les sites avec lesquels ils décident de collaborer.

Par contre, SOM souffre de quelques limitations dont l'abscense d'un modèle probabiliste en particulier. Pour cela, nous avons proposé un algorithme de clustering collaboratif basé cette fois sur GTM qui a été définie pour conserver toutes les propriétés utiles de SOM tout en évitant le plus de ses limitations. Pour ce faire, nous avons ajouté un terme de collaboration à l'étape M de l'algorithme EM de GTM. Par conséquent, les formules de mise à jour des paramètres du GTM sont modifiées (la matrice des poids et la variance).

Néanmoins, le risque de sur-apprentissage en utilisant GTM est élevé. Nous avons alors proposé un algorithm de clustering collaboratif en utilisant la version Variationnelle de GTM (VBGTM) qui a été proposée comme solution évitant le sur-apprentissage de GTM. Nous avons proposé une extension à VBGTM en appliquant une classification floue (FCM) sur les centres des Gaussiennes obtenues lors du calcul de VBGTM. Ensuite, l'agorithme de clustering collaboratif proposé utilise les résultats de cette classification floue pour pouvoir faire collaborer les sites entre eux.

Enfin, nous avons présenté un choix de méthodes permettant d'estimer les coefficients de collaboration d'une manière automatique.

Perspectives

Plusieurs perspectives de recherche peuvent être envisagées suite à ces travaux :

Les algorithmes de clustering collaboratif proposés dans cette thèse utilisent la même méthode de clustering lors de la phase locale et suppose le même nombre de groupes (clusters). Une extension consiste à faire une collaboration entre différentes méthodes de classification en utilisant uniquement les matrices de partition. Le problème du nombre variable de clusters peut être abordé à travers une mesure de similarité entre partition permettant ainsi de choisir un nombre restreint de clusters de la partition la plus proche lors de la collaboration.

Par ailleurs, les algorithmes de clustering proposés dans cette thèse traitent des données d'une maniere hors-ligne (batch), c'est-a-dire que l'échantillon des données est traité en une seule passe et la présence de l'ensemble des données au même temps est nécessaire. Cependant, de nombreuses applications récentes exigent que l'algorithme de clustering soit en-ligne, ou incrémentale, en particulier dans le cas de flux de données, i.e. les données sont fournies de manière continue. En conséquent, l'algorithme de clustering devra progressivement améliorer ses résultats en fonction de l'arrivée des données. Plusieurs problèmes apparaissent dans ce contexte dont la stabilité de la procédure de classification et sa vitesse de convergence.

D'autres perspectives sont aussi envisageables, comme :

- Combiner les approches horizontales et verticales pour avoir une nouvelle approche de collaboration hybride.
- Fusionner les cartes obtenues (SOM, GTM ou VBGTM) après la collaboration pour

construire une classification "consensus" pour tous les sites.
- Etudier l'impact de la diversité entre les différents modèles sur la qualité de la collaboration.
- Utiliser la diversité entre les différents modèles pour guider la collaboration et la rendre sélective.
- Guider la collaboration et l'enrichir en intégrant des connaissances.
- Analyser l'effet de la collaboration "négative" et proposer des approches permettant de la détecter et ainsi de l'éviter pendant l'apprentissage.
- Etudier les liens entre l'apprentissage collaboratif et l'apprentissage par transfert.

Nous travaillons sur certaines de ces perspectives dans le cadre du projet ANR COCLICO[6], en collaboration avec : ICUBE[7], AgroParisTech[8], LIVE[9] et Espace Dev[10].

[6] COllaboration, CLassification, Incrémentalité et COnnaissances. http://icube-coclico.unistra.fr/

[7] Laboratoire des sciences de l'Ingénieur, de l'Informatique et de l'Imagerie, Université de Strasbourg

[8] Institut des sciences et industries du vivant et de l'environnement

[9] Laboratoire Image, Ville, Environnement, Université de Strasbourg

[10] L'espace au service du développement, Université Montpellier 2

Appendix A

Cluster Validity, Choosing the Number of Clusters

The result of a clustering algorithm can be very different from each other on the same data set as the other input parameters of an algorithm can extremely modify the behavior and execution of the algorithm. The aim of the cluster validity is to find the partitioning that best fits the underlying data. Usually 2D data sets are used for evaluating clustering algorithms as the reader easily can verify the result. But in case of high dimensional data the visualization and visual validation is not a trivial tasks therefore some formal methods are needed.

The process of evaluating the results of a clustering algorithm is called cluster validity assessment. Two measurement criteria have been proposed for evaluating and selecting an optimal clustering scheme [19]:

- *Compactness*: The member of each cluster should be as close to each other as possible. A common measure of compactness is the variance.
- *Separation*: The clusters themselves should be widely separated. There are three common approaches measuring the distance between two different clusters: distance between the closest member of the clusters, distance between the most distant members and distance between the centres of the clusters.

There are three different techniques for evaluating the result of the clustering algorithms [60], and several *Validity measures* are proposed: Validity measures are scalar indices that assess the goodness of the obtained partition. Clustering algorithms generally aim at locating well separated and compact clusters. When the number of clusters is chosen equal to the number of groups that actually exist in the data, it can be expected that the clustering algorithm will identify them correctly. When this is not the case, misclassifications appear, and the clusters are not likely to be well separated and compact. Hence, most cluster validity measures are designed to quantify the separation and the compactness of the clusters.

There are two types of clustering validity techniques [77], which are based on external criteria and internal criteria.

- External Criteria: Based on previous knowledge about data.
- Internal Criteria: Based on the information intrinsic to the data alone.

If we consider these two types of cluster validation to determine the correct number of groups from a data set, one option is to use external validation indexes for which a priori knowledge of dataset information is required, but it is hard to say if they can be used in real problems (usually, real problems do not have prior information of the dataset in question). Another option is to use internal validity indexes which do not require a priori information from data set.

We describe some of these validity measures in the following:

Internal validity indexes

- *Davies-Bouldin index (DB)* [49]:
  This index aims to identify sets of clusters that are compact and well separated. The Davies-Bouldin index is defined as:

  $$DB(C) = \frac{1}{C}\sum_{i=1}^{C} \max_{i \neq j} \left\{ \frac{d(X_i) + d(X_j)}{d(C_i, C_j)} \right\} \tag{A.1}$$

  Where C denotes the number of clusters, i, j are cluster labels, then $d(X_i)$ and $d(X_j)$ are all samples in clusters i and j to their respective cluster centres, $d(C_i, C_j)$ is the distance between these centres. Smaller value of DB indicates a "better" clustering solution.

- *BIC index* [39]:
  The Bayesian information criterion (BIC) [153] is devised to avoid overfitting, and is defined as:

  $$BIC = -\ln(L) + \nu \ln(n) \tag{A.2}$$

Where n is the number of objects, L is the likelihood of the parameters to generate the data in the model, and ν is the number of free parameters in the Gaussian model. The BIC index takes into account both fit of the model to the data and the complexity of the model. A model that has a smaller BIC is better.

- *Dunn's Index (DI)* [22, 53]:
  Dunn's index is defined as:

$$DI(C) = \min_{1 \leq i \leq C} \left\{ \min \left\{ \frac{d(C_i, C_j)}{\max_{1 \leq k \leq C} \left(d(X_k) \right)} \right\} \right\} \tag{A.3}$$

  Where $d(C_i, C_j)$ defines the intercluster distance between cluster i and j, $d(X_k)$ represents the intracluster distance of cluster k and C is the number of cluster of data set. Large values of index Dunn correspond to good clustering solution.

- *Xie and Beni's index (XB)* [179]:
  This index is adapted for fuzzy clustering methods. It aims to quantify the ratio of the total variation within clusters and the separation of clusters. It is defined as:

$$XB(C) = \frac{\sum_{i=1}^{C} \sum_{k=1}^{N} (u_{ik})^m \|x_k - v_i\|^2}{N \times \min_{i,n} \|x_n - v_i\|^2} \tag{A.4}$$

  The optimal number of clusters should minimize the value of the index.

External validity indexes

- *Purity index*:
  We calculate the purity of a set of clusters. First, we determine the purity in each cluster. For each cluster, we have the purity $P_i = \frac{1}{n_i} \max_j (n_i^j)$ is the number of objects in i with class label j. In other words, P_i is a fraction of the overall cluster size that the largest class of objects assigned to that cluster represents. The overall purity of the clustering solution is obtained as a weighted sum of the individual cluster purities and is given as:

$$Purity = \sum_{i=1}^{C} \frac{n_i}{N} P_i \tag{A.5}$$

  Where n_i is the size of cluster i, C is the number of clusters, and N is the total number of objects.

- *Entropy*:
  Entropy measures the purity of the clusters class labels. Thus, if all clusters

consist of objects with only a single class label, the entropy is 0. However, as the class labels of objects in a cluster become more varied, the entropy increases. To compute the entropy of a dataset, we need to calculate the

$$E_i = \sum_i P_{ij} \ln(P_{ij}) \tag{A.6}$$

Where the sum is taken over all classes. The total entropy for a set of clusters is calculated as the weighted sum of the entropies of all clusters, as shown in the next equation:

$$E = \sum_{i=1}^{C} \frac{n_i}{N} E_i \tag{A.7}$$

Where n_i is the size of cluster i, C is the number of clusters, and N is the total number of objects.

We used some of these validity measures in this thesis. Other measures are used as well and are described in the thesis.

List of Figures

List of Tables

List of Algorithms

Bibliography

[1] Conference on Neural Information Processing Systems (NIPS). In *http://nips.cc/*. 53

[2] Agrawal, R., and Srikant, R. Privacy-Preserving Data Mining, 2000. 26

[3] Attias, H. A Variational Bayesian Framework for Graphical Models. In *In Advances in Neural Information Processing Systems 12* (2000), MIT Press, pp. 209–215. 95

[4] Avriel, M. *Nonlinear Programming: Analysis and Methods*. Prentice-Hall, Englewood Cliffs, NJ, 1976. 41, 45

[5] Babuška, R., and Verbruggen, H. An overview of fuzzy modeling for control. *Control Engineering Practice 4*, 11 (1996), 1593–1606. 23

[6] Babyak, M. A. What you see may not be what you get: a brief, nontechnical introduction to overfitting in regression-type models. *Psychosomatic medicine 66*, 3 (2004), 411–421. 89

[7] Bache, K., and Lichman, M. UCI Machine Learning Repository, 2013. 17, 53, 82, 83, 106

[8] Balasko, B., Abonyi, J., and Feil, B. *Fuzzy Clustering and Data Analysis Toolbox*. Department of Process Engineering, University of Veszprem, Veszprem, Hungary. 106

[9] Baldi, P., and Brunak, S. *Bioinformatics: the machine learning approach*. Bradford Book, 2001. 11

[10] Ball, N. M., and Brunner, R. J. Data mining and machine learning in astronomy. *International Journal of Modern Physics D 19*, 07 (2010), 1049–1106. 11

[11] Barber, D., and Sollich, P. Gaussian fields for approximate inference in layered sigmoid belief networks. *Advances in neural information processing systems 12* (2000), 393–399. 98

[12] Barlow, H. B. Unsupervised learning. *Neural computation 1*, 3 (1989), 295–311. 14

[13] Barni, M., Cappellini, V., and Mecocci, A. Comments on "A possibilistic approach to clustering". *Fuzzy Systems, IEEE Transactions on 4*, 3 (1996), 393–396. 23

[14] Barreto, G. A. Time Series Prediction with the Self-Organizing Map: A Review. 47

[15] Bartkowiak, A. SOM and GTM: Comparison in Figures., 2003. 73

[16] Bartkowiak, A. Visualizing large data by the SOM and GTM methods–what are we obtaining? In *Intelligent Information Processing and Web Mining: Proceedings of the International IIS: IIPWM´04 Conference Held in Zakopane, Poland, May 17-20, 2004* (2004), vol. 25, Springer Verlag, p. 399. 73

[17] Beal, M. J. Variational algorithms for approximate Bayesian inference. Tech. rep., 2003. 97, 101

[18] Bernardo, J., and Girón, F. A Bayesian analysis of simple mixture problems. *Bayesian statistics 3* (1988), 67–78. 98

[19] Berry, M. J., and Linoff, G. *Data mining techniques: for marketing, sales, and customer support.* John Wiley & Sons, Inc., 1997. 130

[20] Bezdek, J. C. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Kluwer Academic Publishers, Norwell, MA, USA, 1981. 21, 22, 28, 29

[21] Bezdek, J. C., and Dunn, J. C. Optimal fuzzy partitions: A heuristic for estimating the parameters in a mixture of normal distributions. *Computers, IEEE Transactions on 100*, 8 (1975), 835–838. 23

[22] Bezdek, J. C., and Pal, N. R. Cluster validation with generalized Dunn's indices. In *Artificial Neural Networks and Expert Systems, 1995. Proceedings., Second New Zealand International Two-Stream Conference on* (1995), IEEE, pp. 190–193. 132

[23] Bickel, S., and Scheffer, T. Estimation of mixture models using Co-EM. In *In Proceedings of the ICML Workshop on Learning with Multiple Views* (2005). 76

[24] Bishop, C. M. *Pattern recognition and machine learning*, vol. 1. springer New York, 2006. 20, 64, 69, 134, 135

[25] Bishop, C. M., Svensén, M., and I.Williams, C. K. GTM: The generative topographic mapping. *Neural Comput 10*, 1 (1998), 215–234. 42, 44, 60, 62, 71, 75, 107, 135

[26] Bishop, C. M., Svensén, M., and Williams, C. K. I. GTM: A Principled Alternative to the Self-Organizing Map. In *In Advances in Neural Information Processing Systems* (1997), Springer-Verlag, pp. 354–360. 61

[27] Bishop, C. M., Svensén, M., and Williams, C. K. I. Developments of the Generative Topographic Mapping. *Neurocomputing 21* (1998), 203–224. 62, 89, 90, 99

[28] Bliss, G. A. *Lectures on the Calculus of Variations*, vol. 850. University of Chicago press Chicago, 1946. 95

[29] Bonarini, A. Evolutionary learning of fuzzy rules: competition and cooperation. In *Fuzzy Modelling*. Springer, 1996, pp. 265–283. 15

[30] Borgelt, C. *Prototype-based classification and clustering*. PhD thesis, Otto-von-Guericke-Universität Magdeburg, Universitätsbibliothek, 2006. 16

[31] Bose, I., and Mahapatra, R. K. Business data mining—a machine learning perspective. *Information & management 39*, 3 (2001), 211–225. 11

[32] Boyen, X., and Koller, D. Tractable inference for complex stochastic processes. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence* (1998), Morgan Kaufmann Publishers Inc., pp. 33–42. 98

[33] Bratton, D., and Kennedy, J. Defining a standard for particle swarm optimization. In *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE* (2007), IEEE, pp. 120–127. 116

[34] Bruske, B. J., Ahrns, I., Sommer, G., Q-learning, H., and Ecml, S. T. Competitive Hebbian Learning Rule Forms Perfectly Topology. 45

[35] Callen, H. B. *Thermodynamics and an Introduction to Thermostatistics*, 2 ed. Wiley, Sept. 1985. 95

[36] Cannon, R. L., Dave, J. V., and Bezdek, J. C. Efficient implementation of the fuzzy c-means clustering algorithms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2 (1986), 248–255. 23

[37] Caruana, R., and Niculescu-Mizil, A. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning* (2006), ACM, pp. 161–168. 14

[38] Chen, N., Lu, W., Yang, J., and Li, G. *Support vector machine in chemistry.* World Scientific, 2004. 11

[39] Chen, S. S., and Gopalakrishnan, P. S. Clustering via the Bayesian information criterion with applications in speech recognition. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on* (1998), vol. 2, IEEE, pp. 645–648. 131

[40] Chen, Y., Qin, B., Liu, T., Liu, Y., and Li, S. The Comparison of SOM and K-means for Text Clustering. *Computer and Information Science 3*, 2 (2010), P268. 44, 47

[41] Cheng, T. W., Goldgof, D. B., and Hall, L. O. Fast fuzzy clustering. *Fuzzy sets and systems 93*, 1 (1998), 49–56. 23

[42] Cho, S.-B., and Won, H.-H. Machine learning in DNA microarray analysis for cancer classification. In *Proceedings of the First Asia-Pacific bioinformatics conference on Bioinformatics 2003-Volume 19* (2003), Australian Computer Society, Inc., pp. 189–198. 11

[43] Coletta, L. F., Vendramin, L., Hruschka, E. R., Campello, R. J., and Pedrycz, W. Collaborative fuzzy clustering algorithms: Some refinements and design guidelines. *Fuzzy Systems, IEEE Transactions on 20*, 3 (2012), 444–462. 120

[44] Corpet, F. Multiple sequence alignment with hierarchical clustering. *Nucleic acids research 16*, 22 (1988), 10881–10890. 83

[45] da Silva, J. C., Giannella, C., Bhargava, R., Kargupta, H., and Klusch, M. Distributed data mining and agents. *Engineering Applications of Artificial Intelligence 18*, 7 (2005), 791–807. 28

[46] da Silva, J. C., and Klusch, M. Inference in distributed data clustering. *Eng. Appl. Artif. Intell. 19*, 4 (June 2006), 363–369. 26, 28

[47] Dahmane, M., and Meunier, J. Real-time video surveillance with self-organizing maps. In *Computer and Robot Vision, 2005. Proceedings. The 2nd Canadian Conference on* (2005), pp. 136–143. 47

[48] Davé, R. N., and Krishnapuram, R. Robust clustering methods: a unified view. *Fuzzy Systems, IEEE Transactions on 5*, 2 (1997), 270–293. 23

[49] Davies, D. L., and Bouldin, D. W. A Cluster Separation Measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI-1*, 2 (1979), 224–227. 83, 131

[50] Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological) 39*, 1 (1977), 1–38. 62, 66, 97

[51] Depaire, B., Falcon, R., Vanhoof, K., and Wets, G. PSO driven collaborative clustering: A clustering algorithm for ubiquitous environments. *Intell. Data Anal. 15*, 1 (Jan. 2011), 49–68. 24, 41

[52] DeWitt, D., and Gray, J. Parallel database systems: the future of high performance database systems. *Commun. ACM 35*, 6 (June 1992), 85–98. 26

[53] Dunn, J. C. A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. *Journal of Cybernetics 3*, 3 (1973), 32–57. 29, 132

[54] El-Sonbaty, Y., and Ismail, M. Fuzzy clustering for symbolic data. *Fuzzy Systems, IEEE Transactions on 6*, 2 (1998), 195–204. 23

[55] Eschrich, S., Ke, J., Hall, L. O., and Goldgof, D. B. Fast accurate fuzzy clustering through data reduction. *Fuzzy Systems, IEEE Transactions on 11*, 2 (2003), 262–270. 23

[56] Falcon, R., Jeon, G., Bello, R., and 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* (Berlin, Heidelberg, 2007), MICAI'07, Springer-Verlag, pp. 483–495. 41

[57] Fan, J., and Li, R. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association 96*, 456 (2001), 1348–1360. 76

[58] Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., and Uthurusamy, R. Advances in knowledge discovery and data mining. 13

[59] Forman, G., and Zhang, B. Distributed data clustering can be efficient and exact. *SIGKDD Explor. Newsl. 2*, 2 (Dec. 2000), 34–38. 26

[60] Fukunaga, K. *Introduction to statistical pattern recognition.* Academic press, 1990. 130

[61] Garcia, H. L., and Gonzalez, I. M. Self-organizing map and clustering for wastewater treatment monitoring. *Engineering Applications of Artificial Intelligence 17*, 3 (2004), 215–225. 47

[62] Gath, I., and Geva, A. B. Unsupervised optimal fuzzy clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 11*, 7 (1989), 773–780. 23

[63] Ghahramani, Z., and Beal, M. J. Variational Inference for Bayesian Mixtures of Factor Analysers. In *In Advances in Neural Information Processing Systems 12* (2000), MIT Press, pp. 449–455. 95

[64] Ghahramani, Z., and Jordan, M. I. Supervised learning from incomplete data via an EM approach. In *Advances in Neural Information Processing Systems 6* (1994), Morgan Kaufmann, pp. 120–127. 14

[65] Ghassany, M., Grozavu, N., and Bennani, Y. Collaborative Clustering Using Prototype-Based Techniques. *International Journal of Computational Intelligence and Applications 11*, 03 (2012), 1250017. 41

[66] Ghassany, M., Grozavu, N., and Bennani, Y. Collaborative Generative Topographic Mapping. In *Neural Information Processing*, vol. 7664 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, pp. 591–598. 60, 62, 76

[67] Ghassany, M., Grozavu, N., and Bennani, Y. Collaborative Multi-View Clustering. In *Neural Networks (IJCNN), The 2013 International Joint Conference on* (2013), pp. 872–879. 60, 62, 76

[68] Ghosh, J., Strehl, A., and Merugu, S. A Consensus Framework for Integrating Distributed Clusterings Under Limited Knowledge Sharing. In *In Proc. NSF Workshop on Next Generation Data Mining* (2002), pp. 99–108. 26

[69] Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. *Markov chain Monte Carlo in practice*, vol. 2. Chapman & Hall/CRC, 1996. 94

[70] Ginsberg, J., Mohebbi, M. H., Patel, R. S., Brammer, L., Smolinski, M. S., and Brilliant, L. Detecting influenza epidemics using search engine query data. *Nature 457*, 7232 (2008), 1012–1014. 13

[71] Goldberg, D. E., and Holland, J. H. Genetic algorithms and machine learning. *Machine learning 3*, 2 (1988), 95–99. 11

[72] Green, P. J. On Use of the EM Algorithm for Penalized Likelihood Estimation. *Journal of the Royal Statistical Society. Series B (Methodological) 52*, 3 (1990), 443–452. 76

[73] Grozavu, N., Ghassany, M., and Bennani, Y. Learning confidence exchange in Collaborative Clustering. In *Neural Networks (IJCNN), The 2011 International Joint Conference on* (31 2011-aug. 5 2011), pp. 872–879. 41

[74] Gustafson, D. E., and Kessel, W. C. Fuzzy clustering with a fuzzy covariance matrix. In *Decision and Control including the 17th Symposium on Adaptive Processes, 1978 IEEE Conference on* (1978), vol. 17, IEEE, pp. 761–766. 23

[75] Hajjar, C., and Hamdan, H. Kohonen Neural Networks for Interval-valued Data Clustering. *International Journal of Advanced Computer Science 2*, 11 (2012). 45, 134

[76] Halkidi, M., Batistakis, Y., and Vazirgiannis, M. On clustering validation techniques. *Journal of Intelligent Information Systems 17*, 2-3 (2001), 107–145. 83

[77] Halkidi, M., Batistakis, Y., and Vazirgiannis, M. Cluster validity methods: part I. *ACM Sigmod Record 31*, 2 (2002), 40–45. 131

[78] Hamdi, F., and Bennani, Y. Learning random subspace novelty detection filters. In *Neural Networks (IJCNN), The 2011 International Joint Conference on* (2011), IEEE, pp. 2273–2280. 16

[79] Han, J., Kamber, M., and Pei, J. *Data mining: concepts and techniques*. Morgan kaufmann, 2006. 12, 13, 134

[80] Hartigan, J. A. *Clustering algorithms*. John Wiley & Sons, Inc., 1975. 14

[81] Hassan, A. H., Lambert-Lacroix, S., and Pasqualini, F. A new approach of One Class Support Vector Machines for detecting abnormal wafers in Semiconductor. 4th Meeting on Statistics and Data Mining, pp. 35–41. 16

[82] Hastie, T., Tibshirani, R., Friedman, J., and Franklin, J. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer 27*, 2 (2005), 83–85. 13

[83] Hathaway, R. J., and Bezdek, J. C. Nerf¡ i¿ c-means: Non-Euclidean relational fuzzy clustering. *Pattern recognition 27*, 3 (1994), 429–437. 23

[84] Hathaway, R. J., and Bezdek, J. C. Fuzzy c-means clustering of incomplete data. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 31*, 5 (2001), 735–744. 23

[85] Hathaway, R. J., Bezdek, J. C., and Hu, Y. Generalized fuzzy c-means clustering strategies using L¡ sub¿ p¡/sub¿ norm distances. *Fuzzy Systems, IEEE Transactions on 8*, 5 (2000), 576–582. 23

[86] Hathaway, R. J., Davenport, J. W., and Bezdek, J. C. Relational duals of the¡ i¿ c-means clustering algorithms. *Pattern recognition 22*, 2 (1989), 205–212. 23

[87] Hathaway, R. J., and Hu, Y. Density-weighted fuzzy c-means clustering. *IEEE Transactions on Fuzzy Systems 17*, 1 (2009), 243–252. 23

[88] Hawkins, D. M., et al. The problem of overfitting. *Journal of chemical information and computer sciences 44*, 1 (2004), 1–12. 89

[89] Hodge, V., and Austin, J. A survey of outlier detection methodologies. *Artificial Intelligence Review 22*, 2 (2004), 85–126. 16

[90] Hofmann, T. Unsupervised learning by probabilistic latent semantic analysis. *Machine learning 42*, 1-2 (2001), 177–196. 14

[91] Holland, J. H. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control and artificial intelligence.* MIT press, 1992. 11

[92] Höppner, F. Speeding up fuzzy¡ i¿ c-means: using a hierarchical data organisation to control the precision of membership calculation. *Fuzzy Sets and Systems 128*, 3 (2002), 365–376. 23

[93] Hotelling, H. Analysis of a complex of statistical variables into principal components. *The Journal of educational psychology* (1933), 498–520. 85

[94] Huelsenbeck, J. P., Ronquist, F., et al. MRBAYES: Bayesian inference of phylogenetic trees. *Bioinformatics 17*, 8 (2001), 754–755. 94

[95] Hung, M.-C., and Yang, D.-L. An efficient fuzzy c-means clustering algorithm. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on* (2001), IEEE, pp. 225–232. 23

[96] Jaakkola, T. S., and Jordan, M. I. Bayesian Parameter Estimation Via Variational Methods, 1999. 95

[97] Jain, A. K., and Dubes, R. C. *Algorithms for clustering data.* Prentice-Hall, Inc., 1988. 14

[98] Johnson, E. L., and Kargupta, H. Collective, Hierarchical Clustering from Distributed, Heterogeneous Data. In *Revised Papers from Large-Scale Parallel Data Mining, Workshop on Large-Scale Parallel KDD Systems, SIGKDD* (London, UK, UK, 2000), Springer-Verlag, pp. 221–244. 26

[99] Johnson, S. C. Hierarchical clustering schemes. *Psychometrika 32*, 3 (1967), 241–254. 83

[100] Jolliffe, I. T. *Principal component analysis*, vol. 487. Springer-Verlag New York, 1986. 62

[101] Kaelbling, L. P., Littman, M. L., and Moore, A. W. Reinforcement learning: A survey. *arXiv preprint cs/9605103* (1996). 14

[102] Kamel, M. S., and Selim, S. Z. New algorithms for solving the fuzzy clustering problem. *Pattern recognition 27*, 3 (1994), 421–428. 23

[103] Kangas, J. A., Kohonen, T., and Laaksonen, J. T. Variants of self-organizing maps. *Neural Networks, IEEE Transactions on 1*, 1 (1990), 93–99. 44

[104] Kanishka Bhaduri, K. D. K. L. H. K. Privacy Preserving Distributed Data Mining Bibliography. 26

[105] Kapur, J. N., and Kesavan, H. K. *Entropy optimization principles with applications*. Academic Pr, 1992. 96

[106] Kargupta, H., and Chan, P., Eds. *Advances in Distributed and Parallel Knowledge Discovery*. MIT Press, Cambridge, MA, USA, 2000. 26

[107] Kaski, S. Data Exploration Using Self-Organizing Maps, 1997. 47

[108] Kaufman, L., and Rousseeuw, P. J. *Finding groups in data: an introduction to cluster analysis*, vol. 344. Wiley-Interscience, 2009. 23

[109] Kaymak, U., and Setnes, M. Fuzzy clustering with volume prototypes and adaptive cluster merging. *Fuzzy Systems, IEEE Transactions on 10*, 6 (2002), 705–712. 23

[110] Kennedy, J., and Eberhart, R. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on* (1995), vol. 4, pp. 1942–1948 vol.4. 41, 115

[111] Kersten, P. R. Implementation issues in the fuzzy c-medians clustering algorithm. In *Fuzzy Systems, 1997., Proceedings of the Sixth IEEE International Conference on* (1997), vol. 2, IEEE, pp. 957–962. 23

[112] Klusch, M., Lodi, S., and Moro, G. Intelligent information agents. Springer-Verlag, Berlin, Heidelberg, 2003, ch. Agent-based distributed data mining: the KDEC scheme, pp. 104–122. 28

[113] Knott, M., and Bartholomew, D. J. *Latent variable models and factor analysis*. No. 7. Edward Arnold, 1999. 62

[114] Kohonen, T. The self-organizing map. *Proceedings of the IEEE 78*, 9 (1990), 1464–1480. 43, 62

[115] Kohonen, T. *Self-organizing Maps*. Springer-Verlag Berlin, Berlin, 1995. 42, 43, 61, 74

[116] Kohonen, T., Schroeder, M., Huang, T., and Maps, S.-O. Springer-Verlag New York. *Inc., Secaucus, NJ* (2001). 43

[117] Kononenko, I. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine 23*, 1 (2001), 89–109. 11

[118] Kotsiantis, S., Zaharakis, I., and Pintelas, P. Supervised machine learning: A review of classification techniques. *Frontiers in Artificial Intelligence and Applications 160* (2007), 3. 14

[119] Kriegel, H.-P., Kroger, P., Pryakhin, A., and Schubert, M. Effective and Efficient Distributed Model-Based Clustering. In *Proceedings of the Fifth IEEE International Conference on Data Mining* (Washington, DC, USA, 2005), ICDM '05, IEEE Computer Society, pp. 258–265. 26

[120] Krishnapuram, R., Joshi, A., and Yi, L. A fuzzy relative of the k-medoids algorithm with application to web document and snippet clustering. In *Fuzzy Systems Conference Proceedings, 1999. FUZZ-IEEE'99. 1999 IEEE International* (1999), vol. 3, IEEE, pp. 1281–1286. 23

[121] Krishnapuram, R., and Keller, J. M. A possibilistic approach to clustering. *Fuzzy Systems, IEEE Transactions on 1*, 2 (1993), 98–110. 23

[122] Krishnapuram, R., and Keller, J. M. The possibilistic c-means algorithm: insights and recommendations. *Fuzzy Systems, IEEE Transactions on 4*, 3 (1996), 385–393. 23

[123] Lavrač, N. *Machine learning for data mining in medicine*. Springer, 1999. 11

[124] Lawley, D. N., and Maxwell, A. E. *Factor analysis as a statistical method*, vol. 18. Butterworths London, 1971. 62

[125] Liu, Y., Li, Z., Xiong, H., Gao, X., and Wu, J. Understanding of Internal Clustering Validation Measures. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on* (2010), pp. 911–916. 83

[126] Liu, Y., Yao, X., and Higuchi, T. Evolutionary ensembles with negative correlation learning. *Evolutionary Computation, IEEE Transactions on 4*, 4 (2000), 380–387. 15

[127] Loia, V., Pedrycz, W., and Senatore, S. P-FCM: a proximity-based fuzzy clustering for user-centered web applications. *International Journal of Approximate Reasoning 34*, 2 (2003), 121–144. 23

[128] MacKay, D. J. A practical Bayesian framework for backpropagation networks. *Neural computation 4*, 3 (1992), 448–472. 90

[129] MacQueen, J., et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* (1967), vol. 1, California, USA, p. 14. 20

[130] MATLAB. *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts, 2010. 46

[131] McLachlan, G., and Basford, K. Mixture models. Inference and applications to clustering. *Statistics: Textbooks and Monographs, New York: Dekker, 1988 1* (1988). 64

[132] McLachlan, G., and Krishnan, T. The EM Algorithm and Extensions. 1997. 66

[133] McLachlan, G., and Peel, D. *Finite mixture models*. Wiley-Interscience, 2004. 64

[134] Merugu, S., and Ghosh, J. A privacy-sensitive approach to distributed clustering, 2005. 28

[135] Minka, T., and Lafferty, J. Expectation-propagation for the generative aspect model. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence* (2002), Morgan Kaufmann Publishers Inc., pp. 352–359. 98, 99

[136] Minka, T. P. Expectation propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence* (2001), Morgan Kaufmann Publishers Inc., pp. 362–369. 98, 99

[137] Morse, M. *The calculus of variations in the large*, vol. 18. Amer Mathematical Society, 1934. 95

[138] Nabney, I. T. *NETLAB: algorithms for pattern recognition*. Springer, 2004. 106

[139] Olier, I., and Vellido, A. Variational GTM. In *Intelligent Data Engineering and Automated Learning - IDEAL 2007*, H. Yin, P. Tino, E. Corchado, W. Byrne, and X. Yao, Eds., vol. 4881 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, ch. 9, pp. 77–86. 42, 90, 95, 99, 101, 104, 135

[140] Pal, N., and Bezdek, J. Complexity reduction for ”large image” processing. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 32*, 5 (2002), 598–611. 23

[141] Pal, N. R., Pal, K., and Bezdek, J. C. A mixed c-means clustering model. In *Fuzzy Systems, 1997., Proceedings of the Sixth IEEE International Conference on* (1997), vol. 1, IEEE, pp. 11–21. 23

[142] Pal, N. R., Pal, K., Keller, J. M., and Bezdek, J. C. A possibilistic fuzzy c-means clustering algorithm. *Fuzzy Systems, IEEE Transactions on 13*, 4 (2005), 517–530. 23

[143] Park, B.-H., and Kargupta, H. Distributed Data Mining: Algorithms, Systems, and Applications. pp. 341–358. 26, 27

[144] Pearson, K. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2*, 11 (1901), 559–572. 85

[145] Pedrycz, W. Collaborative fuzzy clustering. *Pattern Recognition Letters 23*, 14 (2002), 1675–1686. 23, 25, 28, 34

[146] Pedrycz, W. Fuzzy clustering with a knowledge-based guidance. *Pattern Recogn. Lett. 25*, 4 (2004), 469–480. 28

[147] Pedrycz, W. Interpretation of clusters in the framework of shadowed sets. *Pattern Recogn. Lett. 26*, 15 (2005), 2439–2449. 28

[148] Pedrycz, W. *Knowledge-based clustering: from data to information granules*. Wiley-Interscience, 2005. 23

[149] Pedrycz, W. Collaborative and knowledge-based fuzzy clustering. *International Journal of Innovative, Computing, Information and Control 1*, 3 (2007), 1–12. 23

[150] Pedrycz, W., and Hirota, K. A consensus-driven fuzzy clustering. *Pattern Recogn. Lett. 29*, 9 (2008), 1333–1343. 28

[151] Pedrycz, W., and Rai, P. Collaborative clustering with the use of Fuzzy C-Means and its quantification. *Fuzzy Sets Syst. 159*, 18 (Sept. 2008), 2399–2427. 23, 28

[152] Pereira, F., Mitchell, T., and Botvinick, M. Machine learning classifiers and fMRI: a tutorial overview. *Neuroimage 45*, 1 Suppl (2009), S199. 12

[153] Raftery, A. E. A note on Bayes factors for log-linear contingency table models with vague prior information. *Journal of the Royal Statistical Society, Series B 48* (1986), 249–250. 131

[154] Rahimi, S., Zargham, M., Thakre, A., and Chhillar, D. A parallel Fuzzy C-Mean algorithm for image segmentation. In *Fuzzy Information, 2004. Processing NAFIPS '04. IEEE Annual Meeting of the* (2004), vol. 1, pp. 234–237 Vol.1. 23, 26

[155] Rasmussen, C. E. Gaussian processes for machine learning. In (2006), MIT Press. 99

[156] Rendón, E., Abundez, I., Arizmendi, A., and Quiroz, E. M. Internal versus External cluster validation indexes. *International Journal of computers and communications 5*, 1 (2011), 27–34. 83

[157] Ridolfi, A., and Idier, J. Penalized maximum likelihood estimation for normal mixture distributions. *Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland, Tech. Rep 200* (2002), 285. 76

[158] Ritter, H., and Schulten, K. On the stationary state of Kohonen's self-organizing sensory mapping. *Biol. Cybern. 54*, 2 (June 1986), 99–106. 61

[159] Ritter, H., and Schulten, K. Convergence properties of Kohonen's topology conserving maps: fluctuations, stability, and dimension selection. *Biol. Cybern. 60*, 1 (Nov. 1988), 59–71. 61

[160] Rousseeuw, P. J., and Leroy, A. M. *Robust regression and outlier detection*, vol. 589. Wiley, 2005. 16

[161] Ruspini, E. H. Numerical methods for fuzzy clustering. *Information Sciences 2*, 3 (1970), 319–350. 21

[162] Safarinejadian, B., Menhaj, M., and Karrari, M. Distributed data clustering using expectation maximization algorithm. *Journal of Applied Sciences 9*, 5 (2009), 854–864. 76

[163] Samatova, N. F., Ostrouchov, G., Geist, A., and Melechko, A. V. RACHET: An Efficient Cover-Based Merging of Clustering Hierarchies from Distributed Datasets. *Distrib. Parallel Databases 11*, 2 (Mar. 2002), 157–180. 28

[164] Sun, Y. On quantization error of self-organizing map network. *Neurocomputing 34*, 1–4 (2000), 169–193. 52

[165] Sutton, R. S., and Barto, A. G. *Reinforcement learning: An introduction*, vol. 1. Cambridge Univ Press, 1998. 14

[166] Svensén, M. The Generative Topographic Mapping. Tech. rep., PhD thesis, Aston University, 1998. 89, 99

[167] Tanner, M. A. *Tools for Statistical Inference*, third ed. Springer, 1996. 99

[168] Timm, H., Borgelt, C., Döring, C., and Kruse, R. An extension to possibilistic fuzzy cluster analysis. *Fuzzy Sets and Systems 147*, 1 (2004), 3–16. 23

[169] Ultsch, A., and Siemon, H. P. Kohonen's Self Organizing Feature Maps for Exploratory Data Analysis. In *Proceedings of International Neural Networks Conference (INNC)* (Paris, 1990), Kluwer Academic Press, pp. 305–308. 47

[170] Utsugi, A. Bayesian Sampling and Ensemble Learning in Generative Topographic Mapping. *Neural Process. Lett. 12*, 3 (Dec. 2000), 277–290. 99

[171] Vellido, A., El-Deredy, W., and Lisboa, P. J. Selective smoothing of the generative topographic mapping. *Neural Networks, IEEE Transactions on 14*, 4 (2003), 847–852. 89, 90

[172] Verbeek, J., Vlassis, N., and Krose, B. Self-organizing mixture models. *Neurocomputing 63* (2005), 99–123. 44

[173] Vesanto, J. SOM-Based Data Visualization Methods. *Intelligent Data Analysis 3* (1999), 111–126. 47, 74

[174] Vesanto, J., and Alhoniemi, E. Clustering of the Self-Organizing Map, 2000. 47

[175] Vesanto, J., Himberg, J., Alhoniemi, E., and Parhankangas, J. Self-Organizing Map in Matlab: the SOM Toolbox. In *In Proceedings of the Matlab DSP Conference* (2000), pp. 35–40. 46, 74

[176] Walter, J., Ritter, H., and Schulten, K. Nonlinear prediction with self-organizing maps. In *Neural Networks, 1990., 1990 IJCNN International Joint Conference on* (1990), pp. 589–594 vol. 1. 47

[177] Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*. Springer, 1992, pp. 5–32. 14

[178] Witten, I. H., and Frank, E. *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2005. 11, 13

[179] Xie, X. L., and Beni, G. A validity measure for fuzzy clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 13*, 8 (1991), 841–847. 107, 132

[180] Yang, M.-S., and Lai, C.-Y. A robust automatic merging possibilistic clustering method. *Fuzzy Systems, IEEE Transactions on 19*, 1 (2011), 26–41. 23

[181] Yu, F., Tang, J., Wu, F., and Sun, Q. Auto-weighted Horizontal Collaboration Fuzzy Clustering. 592–600. 41, 118

[182] Zehraoui, F., and Bennani, Y. New self-organizing maps for multivariate sequences processings. vol. 5, pp. 439–456. 44

[183] Zhang, S., Zhang, C., and Wu, X. *Knowledge Discovery in Multiple Databases.* SpringerVerlag, 2004. 25

[184] Zhou, A., Cao, F., Yan, Y., Sha, C., and He, X. Distributed Data Stream Clustering: A Fast EM-based Approach. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on* (2007), pp. 736–745. 76

UNIVERSITÉ PARIS 13, SORBONNE PARIS CITÉ

Résumé

Doctor of Philosophy

Contributions à l'Apprentissage Collaboratif non Supervisé

by Mohamad Ghassany

Le travail de recherche exposé dans cette thèse concerne le développement d'approches de clustering collaboratif à base de méthodes topologiques, telles que les cartes auto-organisatrices (SOM), les cartes topographiques génératives (GTM) et les GTM variationnelles Bayésiennes (VBGTM). Le clustering collaboratif permet de préserver la confidentialité des données en utilisant d'autres résultats de classifications sans avoir recours aux données de ces dernières. Ayant une collection de bases de données distribuées sur plusieurs sites différents, le problème consiste à partitionner chacune de ces bases en considérant les données locales et les classifications distantes des autres bases collaboratrices, sans partage de données entre les différents centres. Le principe fondamental du clustering collaboratif est d'appliquer les algorithmes de clustering localement sur les différents sites, puis collaborer les sites en partageant les résultats obtenus lors de la phase locale. Dans cette thèse nous explorons deux approches pour le clustering collaboratif. L'approche horizontale pour la collaboration des bases de données qui décrivent les mêmes individus mais avec des variables différentes. La deuxième approche collaborative est dite verticale pour la collaboration de plusieurs bases de données contenant les mêmes variables mais avec des populations différentes.

Abstract

The research outlined in this thesis concerns the development of collaborative clustering approaches based on topological methods, such as self-organizing maps (SOM), generative topographic mappings (GTM) and variational Bayesian GTM (VBGTM). So far, clustering methods performs on a single data set, but recent applications require data sets distributed among several sites. So, communication between the different data sets is necessary, while respecting the privacy of every site, i.e. sharing data between sites is not allowed. The fundamental concept of collaborative clustering is that the clustering algorithms operate locally on individual data sets, but collaborate by exchanging information about their findings. The strength of collaboration, or confidence, is precised by a parameter called coefficient of collaboration. This thesis proposes to learn it automatically during the collaboration phase. Two data scenarios are treated in this thesis, referred as vertical and horizontal collaboration. The vertical collaboration occurs when data sets contain different objects and same patterns. The horizontal collaboration occurs when they have same objects and described by different patterns.