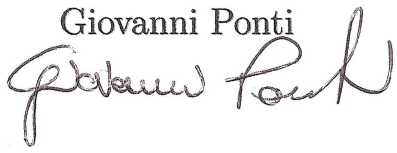
UNIVERSITÀ DELLA CALABRIA

Dottorato di Ricerca in
Ingegneria dei Sistemi e Informatica
XXII Ciclo

Ph.D. Thesis

Advances in Mining Complex Data: Modeling and Clustering

Giovanni Ponti



Director
Prof. Luigi Palopoli

Advisor
Prof. Sergio Greco

DIPARTIMENTO DI ELETTRONICA, INFORMATICA E SISTEMISTICA
Settore Scientifico Disciplinare: ING-INF/05

Acknowledgments

It is a honor for me to be part of a research group where people work with devotion and passion in a friendly atmosphere. This environment has provided me with incitements and support during my studies.

The first acknowledgment goes to my advisor Prof. Sergio Greco. He was the first who believed in my capabilities and introduced me in the scientific research world. His suggestions were very precious and his intuitions opened my mind to many interesting research opportunities.

A special thanks to Andrea Tagarelli, who led me carefully to and through my research activities. His critical observations and his constructive remarks contributed to the growth of my research capabilities. He also revealed to be a friend who provided encouragements in difficulties. His presence is invaluable, a reference point in work and in life.

I also acknowledge Prof. Luigi Palopoli, who directed the Ph.D. course. He was always interested in the evolution of my activities and his suggestions were valuable to enrich my studies.

The majority of the topics shown in this thesis are result of research activities conducted with my room mate, Francesco Gullo. Such results could not be obtained without his contribution. We worked together sharing opinions and providing precious remarks each other. Apart from work, we shared many moments of friendship. For these reasons, he deserves a great thanks.

There were many people at DEIS who stood by me during these years. All of them gave me an important contribution to improve my knowledge with helpful feedback, discussions, and complementary points of view. In particular, I would thank Bettina Fazzinga, Sergio Flesca, Filippo Furfaro, Massimiliano Mazzeo, Cristian Molinaro, Francesco Parisi, Andrea Pugliese, Francesca Spezzano, and Ester Zumpano for work and joy moments spent together. I also thank Mario Cannataro, Pierangelo Veltri, and Giuseppe Tradigo from UNICZ, who provided me a valuable support in several issues of my research experience. I acknowledge Giovanni Costabile and Gabriele Gigliotti for their help in bureaucratic affairs.

Apart from the work environment, I would thank people with whom I spent my leisure time. For this reason, a great thanks goes to all my friends of "Post Cresima". I get known them since the youth and we grew up together, sharing lots of joyful and spiritual moments.

An invaluable thanks goes to my parents Luigi and Mena, as well as my sister Barbara. They always supported me in pursuing my objectives. A whole life would not be enough to express all my gratitude to them.

Last but not least, my deepest thanks is for Mari, my better half. She is the most important person in my life, and her love gave me the strength to overcome difficulties I faced during these years. Also, discussions and moments spent together were very important for my overall growth. My life is better since we met.

to Mari

Abstract

In the last years, there has been a great production of data that come from different application contexts. However, although technological progress provides several facilities to digitally encode any type of event, it is important to define a suitable representation model which underlies the main characteristics of the data. This aspect is particularly relevant in fields and contexts where data to be archived can not be represented in a fix structured scheme, or that can not be described by simple numerical values. We hereinafter refer to these data with the term *complex data*.

Although it is important define ad-hoc representation models for complex data, it is also crucial to have analysis systems and data exploration techniques. Analysts and system users need new instruments that support them in the extraction of patterns and relations hidden in the data. The entire process that aims to extract useful information and knowledge starting from raw data takes the name of *Knowledge Discovery in Databases* (KDD). It starts from raw data and consists in a set of specific phases that are able to transform and manage data to produce models and knowledge. There have been many knowledge extraction techniques for traditional structured data, but they are not suitable to handle complex data.

Investigating and solving representation problems for complex data and defining proper algorithms and techniques to extract models, patterns and new information from such data in an effective and efficient way are the main challenges which this thesis aims to face. In particular, two main aspects related to complex data management have been investigated, that are the way in which complex data can be modeled (i.e., *data modeling*), and the way in which homogeneous groups within complex data can be identified (i.e., *data clustering*). The application contexts that have been objective of such studies are *time series data*, *uncertain data*, *text data*, and *biomedical data*.

It is possible to illustrate research contributions of this thesis by dividing them into four main parts, each of which concerns with one specific area and data type:

Time Series — A time series representation model has been developed, which is conceived to support accurate and fast similarity detection. This model is called *Derivative time series Segment Approximation (DSA)*, as it achieves a concise yet feature-rich time series representation by combining the notions of *derivative estimation*, *segmentation* and *segment approximation*.

Uncertain Data — Research in uncertain data mining went into two directions. In a first phase, a new proposal for partitional clustering has been defined by introducing the *Uncertain K-medoids (UK-medoids)* algorithm. This approach provides a more accurate way to handle uncertain objects in a clustering task, since a cluster representative is an uncertain object itself (and not a deterministic one). In addition, efficiency issue has been addressed by defining a distance function between uncertain objects that can be calculated offline once per dataset.

In a second phase, research activities aimed to investigate issues related to hierarchical clustering of uncertain data. Therefore, an agglomerative centroid-based linkage hierarchical clustering framework for uncertain data (*U-AHC*) has been proposed. The key point lies in equipping such scheme with a more accurate distance measure for uncertain objects. Indeed, it has been resorted to *information theory* field to find a measure able to compare probability distributions of uncertain objects used to model uncertainty.

Text Data — Research results on *text data* can be summarized in two main contributions. The first one regards clustering of *multi-topic documents*, and a framework for hard clustering of documents according to their mixtures of topics has been proposed. Documents are assumed to be modeled by a generative process, which provides a mixture of probability mass functions (pmfs) to model the topics that are discussed within any specific document. The framework combines the expressiveness of generative models for document representation with a properly chosen information-theoretic distance measure to group the documents.

The second proposal concerns *distributional clustering of XML documents*, focusing on a the development of a distributed framework for efficiently clustering XML documents. The distributed environment consists of a peer-to-peer network where each node in the network has access to a portion of the whole document collection and communicates with all the other nodes to perform a clustering task in a collaborative fashion. The proposed framework is based on modeling and clustering XML documents by structure and content. Indeed, XML documents are transformed into *transactional data* based on the notion of *tree tuple*. The framework is based on the well-known paradigm of *centroid-based partitional clustering* to conceive the distributed, transactional clustering algorithm.

Biomedical Data — Research results on time series and uncertain data have been involved to support effective and efficient biomedical data management. The focus regarded both proteomics and genomics, investigat-

ing Mass Spectrometry (MS) data and microarray data. In the specific, a *Mass Spectrometry Data Analysis (MaSDA)* system has been defined. The key idea consists in exploiting temporal information implicitly contained in MS data and model such data as time series. The major advantages of this solution are the dimensionality and the noise reduction. As regards micrarray data, U-AHC has been employed to perform clustering of microarray data with probe-level uncertainty. A strategy to model probe-level uncertainty has been defined, together with a hierarchical clustering scheme for analyzing such data. This approach performs a gene-based clustering to discover clustering solutions that are well-suited to capture the underlying gene-based patterns of microarray data.

The effectiveness and the efficiency of the proposed techniques in clustering complex data are demonstrated by performing intense and exhaustive experiments, in which such proposals are extensively compared with the main state-of-the-art competitors.

Rende, November 30, 2009
Giovanni Ponti

Contents

List of Figures

List of Tables

Part I

The Basics

1

Introduction

1.1 Data heterogeneity and Information Systems

In the last years, there has been a relevant technological progress in many fields of business and commerce. This phenomenon has lead to the production of a large amount of data results of business activities. While in the past data were collected by hand using papers and notes, nowadays it is no longer possible to use such storing methods because data are more complex and come from applications that have a high rate of info processing.

The spread of Internet and the massive usage of network communications allow the development of systems that are able to store and generate data coming from different sources, that are heterogeneous and that are represented according to different schemes. In fact, although there is the possibility of digitally encoding any type of event, the problem of adopting a suitable representation able to highlight the main characteristics of the data still occurs.

In general, it is possible to think of data as a set of aspects and features that are expression of a specific concept or that are related to a certain event. Traditional database systems are particularly suitable for storing these types of data, and it is possible to use simple statistic techniques to explore such data and finding out patterns, regularities and relations.

However, there are many fields and contexts in which data to be archived can not be represented by a fix structured scheme, or that can not be described by simple numerical values. Data that come from sensors, trajectories of moving objects, text documents or web pages need for proper representation techniques to cover all their features and their complex and not well-defined structure. Modern database systems have to include modules and proper methodologies that are able to handle this particular kind of data. Moreover, the problem of analyzing these data also leads to the development of new techniques and algorithm that are able to discover trends and profiles.

In order to face these issues, computer-based systems are widely used to help users in organizing and managing everything produced by applications and/or business tasks. Within this view, computers and workstations

are largely employed in industries and companies that need to use and analyze data generated by their processes and activities. However, although systems that archive and integrate data have improved their effectiveness and their efficiency in the last few years, it is difficult to explore this huge and heterogeneous amount of data to produce new relevant and strategic concepts. Analysts and system users need new instruments and techniques that support them in the extraction of patterns and relations hidden in the data. Indeed, it results to be very important to distinguish between *data* and *information*. Usually, we refer to "data" as something that contains features and values related to a fact and that are used to describe an event by means of a set of variables. Within this view, *raw data* are useful only for *descriptive* purposes. Instead, with the term "information" we refer to something that has strategical value and that provides indications; in substance, while data are fundamental to describe a phenomenon, information is everything that comes from the interpretation and the analysis of such description. Moreover, information is the only thing that can be used by analysts to understand the data and to outline strategies.

1.2 Knowledge Discovery in Databases

The entire process that aims to extract useful information and knowledge starting from raw data takes the name of *Knowledge Discovery in Databases* (KDD) [FPSS96a, FPSS96b]. The process starts from raw data and consists in a set of specific phases that are able to transform and manage data to produce models and knowledge.

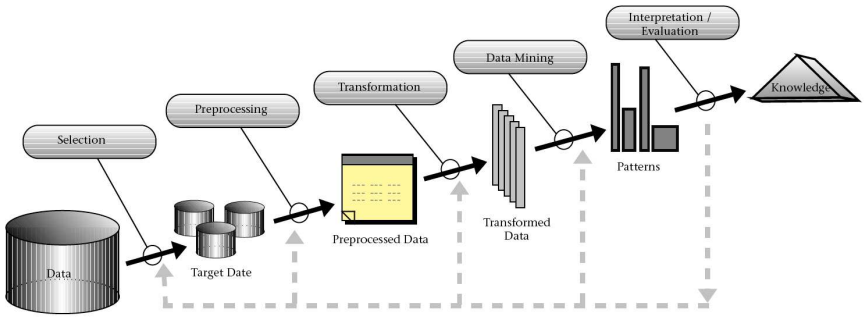


Fig. 1.1. The Knowledge Discovery in Databases (KDD) process

Figure 1.1 shows the main steps involved in the whole KDD process. The first three steps (i.e., *selection*, *preprocessing* and *transformation*) substan-

tially operate on raw data and perform filtering and transformation tasks. These phases are particularly useful to solve all the problems regarding data representation and data integration (as discussed above), and to prepare data for the next analysis processes. *Data mining* techniques (i.e., the fourth step of the KDD process) furnish a collection of procedures and algorithms to find out relations and hidden patterns within the data. This step has a great impact in the entire KDD process since it is responsible of producing good models that can summarize main data features and that are able to underline new trends, patterns and relations. The last step of the KDD process consists in the *interpretation* and *validation* of the results. It is important to note that this step takes, as an input, the output of data mining tasks, and not the raw data and/or the transformed ones; therefore, it exploits relations and models produced by data mining techniques and infers new information and knowledge.

1.2.1 Data Mining

The key point in the knowledge discovery process is the data mining, so much so that the "data mining" and "KDD" terms are often treated as synonyms [HK00].

Data mining tasks are classified into *predictive* and *descriptive* [FPSS96a]. Predictive tasks aim to build a model that is useful for predicting future behaviors or values of certain features. These comprise *association analysis*, i.e., discovering *association rules* that show attribute-value conditions occurring frequently together in a given set of data; *classification* and *prediction*, i.e., deriving some models (or functions) which describe data classes or concepts by a set of data objects whose class label is known (i.e., the *training set*); such models have the main goal of being used to predict the class of objects whose class label is unknown as accurately as possible; *deviation detection*, i.e., dealing with *deviations* in data, which are defined as differences between measured values and corresponding references such as previous values or normative values; *evolution analysis*, i.e., detecting and describing regular patterns in data whose behavior changes over time. On the other hand, in a descriptive data mining task, the built model has to describe the data in an understandable, effective, and efficient form. Relevant examples of descriptive tasks are *data characterization*, whose main goal is to summarize the general characteristics or features of a target class of data, *data discrimination*, i.e., a comparison of the general features of target class data objects with the general features of objects from one or a set of contrasting classes, and *clustering*.

Regardless of the particular task, data mining techniques have to cope with a twofold problem. The first aspect concerns the way the data are modeled, while the second one is related to the strategies and to the algorithms which are involved in the data mining phase and that have to produce models and extract information.

1.3 Challenges for Complex Data

Investigating and solving representation problems for complex data and defining proper algorithms and techniques to extract models, patterns and new information from such data in an effective and efficient way are the main challenges which this thesis aims to face.

In the first sections of this introduction, it has been discussed about issues related to complex data and how it is difficult to find a representation model that preserves distinctive aspects of such data. Nowadays, database systems are very powerful and versatile tools which can be suitable for storing every kind of data; however, they still depend on traditional relational model. Therefore, complex data representation needs the definition of ad-hoc methodologies for modeling data in such a way that it is possible to archive them in concise yet feature-rich models, i.e., performing summarization techniques which capture main characteristic features.

My studies and research activities have dealt with issues related to mining of complex data. In particular, two main aspects have been investigated, which are *data modeling* and *data clustering*. The application contexts objective of such studies are *time series data*, *uncertain data*, *text data*, and *biomedical data*.

In the following, an overall description of issues and challenges in the areas of interest discussed in this thesis will be drawn.

1.3.1 Trajectories and Time Series

The first type of complex data investigated in this thesis concerns data that come from objects that evolve in time and space. It is usual to refer to these data as *moving objects* or *trajectories* [LOW02, LHY04]. A trajectory consists in a set of observations, each of which is composed by data point measurements, and temporal and spatial references.

There are many fields in which spatio-temporal data are largely present, especially telecommunication and mobile systems, but data can be also found in other domains and, in general, in each application that requires to trace changes of a specific feature in the space and in the time. Relational databases are not suitable for storing spatio-temporal data, since they do not have enough capabilities to address all the requirements that are related to these data. For these reasons, spatio-temporal data need ad-hoc storing and managing systems, that are known as *spatial databases* [WZL00, PT98]. A spatial database has many special requirements to support complex operations performed on its data but, as a main one, it has to allow to associate spatio-temporal references to the data; in this way, it is possible to handle and analyze interactions between objects, which are expressed in terms of past, present or future states. In such a scenario, analyzing spatio-temporal data has the objective of finding interesting similarities and patterns that are able to underline main trends.

The simplest case of spatio-temporal data are *time series*, that are two-dimensional trajectories. A time series (or *time sequence*) is represented by means of a sequence of numerical values associated to a certain characteristic of a phenomenon, as shown in Figure 1.2.

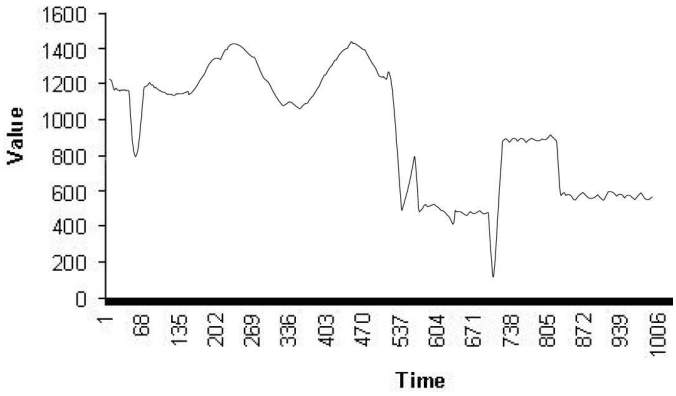


Fig. 1.2. An example of *time series*

This sequence is ordered according to the time reference associated to each observation. To cope with issues that regard time series data has a great relevance in many applications, such as indexing, querying, classification and clustering. The key point here concerns the way time series are compared, and it is crucial to define ad-hoc distance measures which take into account all the features present in these complex objects.

1.3.2 Uncertain data

Another type of complex data that deserve to be investigated are *uncertain data*. In the last years, there has been a grown of applications and modern technologies which are able to store and record data that present errors or that are partially complete. These data come from many application contexts, such as sensor networks and mobile applications, and arise from various aspects, e.g., implicit randomness in data acquisition/generation process, imprecision in physical measurements, and data staling. This makes uncertainty inherently present in several application domains. For instance, sensor measurements may be imprecise at a certain degree due to the presence of various noisy factors (e.g., signal noise, instrumental errors, wireless transmission) [CLL06, CKP03]. To address this issue, it is advisable to model sensor data as continuous probability distributions [FGB02, DGM$^+$05]. Another example are data representing moving objects, which continuously change their location so that the exact positional information at a given time instant may be unavailable [LHY04]. Moreover, some methods have recently been defined to handle

uncertainty in gene expression data [MFNL03a, HRC+05a, LMLR05a]. Further examples come from distributed applications, privacy preserving data mining, and forecasting or other statistical techniques used to generate data attributes [AY09]. Handling and analyzing these data are two of the most relevant problems that *uncertain data management* algorithms and applications aim to solve.

Depending on the application domain, it is possible to define several notions of uncertainty (e.g., [IL84, AKG87, Sad91, LLRS97, DS04, GT06, Agg07]). Uncertain data objects are usually represented in terms of an uncertainty region over which a *probability distribution* is defined [CKP03, KP05a, CCKN06]. This aspect leads to data that can be seen as located into a multidimensional region defined by the parameters used to describe their features. Dealing with such data has raised several issues in data management and knowledge discovery, mainly due to the intrinsic difficulty underlying the various notions of uncertainty. Substantially, it is possible to identify two main challenges related to the field of uncertain data: the first issue concerns the way these uncertain data are modeled and stored in database systems, while the second one regards data management and data analysis techniques performed on these particular data.

1.3.3 Text data

Computer-based systems allow to produce and store a huge amount of data and it has been largely discussed how these data are heterogeneous. However, if we think at the whole data types, we realize that the majority of them are *text documents*, and this consideration holds nowadays, since Internet and multimedia libraries have a lot of digital text data [HNP05]. Texts are used to define concepts by means of linguistic units, and these information described are very complex, with many features that can not be directly represented in standard database systems. Phrases and words of natural language have an enormous expressive power and are particularly suitable for providing qualitative descriptions, mental picture of something and event reports.

Although there are different levels of structuring, i.e., unstructured data (plain documents), semi-structured data (HTML and XML documents) [Tag05], it is very difficult to find a way to encode information described within text data. Computer systems can not use directly information stored in unstructured text for analysis purposes; it is necessary to perform a preliminary preprocessing phase, which consists in applying a set of specific methods and algorithms that are able to filter out text from useless elements and to extract statistical information from text units.

We can "mine" the text, that is dig out the hidden gold from textual information: the challenge here consists in overcoming the ambiguity of natural language interpretation to distill information buried inside the text, to discover interesting patterns or trends and extracting knowledge from them.

That is where *text mining* [FD95] comes in. The goal of text mining is to adapt and apply data mining techniques in such a way that algorithms are able to deal with text data, and these methods are suitable to extract interesting information and knowledge from this huge amount of unstructured masses of text.

1.3.4 Biomedical Data

Another interesting field in which computer-based methodologies are used to store and to analyze complex data is medicine. In particular, the focus of the research activities in this field is on *biomedicine*, since it does not concern with the practice of medicine but with the theory: biomedicine embraces all the methodologies and techniques that are used to research, to known molecular phenomenons and to produce diagnosis and medical treatments.

Biomedicine can be divided into two principal areas, that are *genomics* [LO009, BC09, UCCA09] and *proteomics* [AA98, Bla99]. Genomics studies the *genome*, which refers to the DNA sequence of organisms. A DNA corresponds to a transcription of nucleotide sequence that defines a structural scheme and a biological mapping of the characteristics of an organism. Proteomics, instead, studies the *proteome*, which is the entire set of proteins expressed from a genome. It studies protein expressions and functions, and all the mechanisms in which proteins are involved, in order to understand illness states and to identify protein alterations. While the genome of a cell does not vary during its life and is the same for all the cells of each organism, proteome is very dynamic and changes from cell to cell and from time to time, since it is very sensitive to external factors and to activities of other cells. In addition, proteomics is often seen as a next phase of analysis process of cells, that follows genomics and aims to study gene functions and protein relations.

Despite of the specific area of interest, biomedicine avails itself of computer science techniques to handle the huge amount of complex data produced during laboratory analysis. Computer-based systems interact directly with medical instruments to store data and, subsequently, to analyze genes and/or proteins to discover interesting patterns. Once again, data mining techniques reveal to be a very useful set of tools which help analyst in knowledge process applied in biomedical areas.

1.4 Contributions

The main research activities carried out during these years of my Ph.D. studies have been concerned with complex data management. In particular, my studies have been focused on two principal issues in the management of complex data, that are *data representation* and *knowledge extraction*. Facing the first issue means to find strategies for structuring data and defining a way of representing their characteristics in a compact yet feature-rich way. The

second issue is highly related with the first one and aims to extract information and hidden relations within the data; this step needs for the definition of proper data mining techniques and algorithms which are able to handle complex data and explore their features and aspects to extract patterns and knowledge.

It is possible to illustrate research contributions of this thesis by dividing them into three main parts, each of which concerns with one specific area and data type:

Time Series — A time series representation model has been developed, which is conceived to support accurate and fast similarity detection [GPTG07, GPTG09]. This model is called *Derivative time series Segment Approximation (DSA)*, as it achieves a concise yet feature-rich time series representation by combining the notions of *derivative estimation*, *segmentation* and *segment approximation*. DSA involves a segmentation scheme that employs the paradigm based on a piecewise discontinuous function. However, in contrast to any other technique of dimensionality reduction, the segmentation step is performed on the derivative version of the original time series, rather than directly on the raw time series, in order to underline features and trends of the original time series. The intuition underlying the DSA model works out very advantageously in supporting accurate and fast similarity detection.
DSA has been also involved in two real-case applications. In particular, this methodology has been applied to biomedicine, by modeling and analyzing *mass spectrometry (MS) data* [GPT$^+$07, GPT$^+$08b, GPT$^+$09c], and to electricity data, by modeling *load profiles* associated to low-voltage electricity customer consumption data [GPT$^+$09a].

Mass Spectrometry. A system for advanced analysis of MS data has been developed [GPT$^+$09c]. The general objective of this system, called *Mass Spectrometry Data Analysis (MaSDA)*, is to assist the user in discovering useful knowledge from MS data. The key idea consists in exploiting temporal information implicitly contained in MS data and model such data as time series [GPT$^+$07]. The major advantages of this solution are the dimensionality and the noise reduction. Another important aspect of MaSDA is that it offers a graphical tool for preprocessing the raw mass spectra called *Mass Spectra Preprocessing tool (MSPtool)* [GPT$^+$08b]. It provides the user with a wide set of MS preprocessing steps by means of an easy-to-use graphical interface. Besides the functionalities of time series based MS modeling and MS preprocessing, the MaSDA system is designed to perform various tasks of MS data analysis, by employing *Data Mining and Knowledge Discovery* techniques, and to evaluate and visualize the patterns of knowledge discovered from the input MS data.

Load Profiles. As regards electricity data, a clustering framework for electricity customer low-voltage (LV) load profiles has been pro-

posed [GPT+09a], which is supported by information on meta-data. A major emphasis of this study was on the most typical class of electricity customers, i.e., private, residential domestic customers. In this context, the goal of the research consisted in the characterization of LV customers based on their consumption data. In order to accomplish customer profiling task, a time series based model has been used to suitably represent load profiles and enable the detection of their characteristic trends. Besides this primary data, meta-data associated to the load profiles have been also exploited, which revealed to be useful to enrich a-priori knowledge on the customers. The clustering framework has been conceived for detecting groups of customers having similar consumption behavior.

Uncertain Data — Research in uncertain data mining went into two directions. In a first phase, a new proposal for partitional clustering has been defined by introducing the *Uncertain K-medoids (UK-medoids)* [GPT08a] algorithm. This approach provides a more accurate way to handle uncertain objects in a clustering task, since a cluster representative is an uncertain object itself (and not a deterministic one). In addition, efficiency issue has been also taken into account by defining a distance function between uncertain objects that can be calculated offline once per dataset.
In a second phase, our researches aimed to investigate issues related to hierarchical clustering of uncertain data. Therefore, a hierarchical clustering framework for uncertain data, called *U-AHC* [GPTG08] has been proposed. The key point lies in equipping such scheme with a more accurate distance measure for uncertain objects. Indeed, it has been resorted to *information theory* field to find a measure that is able to compare probability distributions of uncertain objects, which are used to model uncertainty. In addition, since the clustering scheme uses a centroid-based linkage, a cluster prototype is defined as a mixture model that summarizes the probability distributions of all the objects within that cluster. In this way, the distance measure is applied to cluster prototype (and not directly to uncertain objects), with consequent advantages in terms of efficiency. Theoretical studies have been provided to validate this proposal, while experimental results also assessed the high-quality clustering solution achieved by the proposed analysis framework.
This technique has also been employed to perform clustering of microarray data with probe-level uncertainty [GPT+09b]. A strategy to model probe-level uncertainty and a hierarchical clustering scheme for analyzing such data have been defined. This approach performs a gene-based clustering to discover clustering solutions that are well-suited to capture the underlying gene-based patterns of microarray data.

Text Data — Research results on *text data* can be summarized in two main contributions. The first one regards clustering of *multi-topic docu-*

ments [PT09], while the second one concerns *distributional clustering of XML documents* [GGPT09].

Clustering of Multi-Topic Documents. A framework for hard clustering of documents according to their mixtures of topics has been proposed [PT09]. Documents are assumed to be modeled by a generative process, which provides a mixture of probability mass functions (pmfs) to model the topics that are discussed within any specific document. The framework combines the expressiveness of generative models for document representation with a properly chosen information-theoretic distance measure to group documents, which computes the distance between any two probability distributions over a topic feature space. The proposed clustering method refers to a centroid-linkage-based agglomerative hierarchical scheme. Moreover, it is "hard" in that it produces a clustering solution for a given collection of documents in such a way that each document is assigned to a unique cluster and each cluster contains documents that share the same topic assignment.
The effectiveness of the proposed approach has been experimentally evaluated over large collections of documents. A major goal of this evaluation was to assess the advantages of combining the expressiveness of state-of-the-art generative models for document representation with information-theoretic distance; this strategy revealed to be a compelling solution for topic-based hard clustering of documents.

Distributional Clustering. The proposal focused on the development of a distributed framework for efficiently clustering XML documents [GGPT09]. The distributed environment consists of a peer-to-peer network where each node has access to a portion of the whole document collection and communicates with all the other nodes to perform a clustering task in a collaborative fashion. The proposed framework is based on an approach to model and cluster XML documents by structure and content [TG06, TG09]. Indeed, XML documents are transformed into *transactional data* according to the notion of *tree tuple*. XML tree tuples enable a flat, relational-like XML representation that is well-suited to meet the requirements for clustering XML documents according to structure and content information. The framework is based on the well-known paradigm of *centroid-based partitional clustering* to conceive the distributed, transactional clustering algorithm. It should be emphasized that such a clustering paradigm is particularly appealing to a distributed environment. Indeed, the availability of a summarizing description of the clustered data provided by the cluster centroids is highly desirable especially when the input data is spread across different peers. Cluster centroids are hence used to describe portions of the document collection and can conveniently be exchanged with other nodes on the network.

1.5 Outline of the Thesis

The outline of the thesis follows the scheme proposed in this introduction. It is organized in four parts. Part I — *The Basics* provides the basic background knowledge on the areas that are investigated in this thesis. Part II, Part III, and Part IV present research proposals for clustering complex data.

In particular, Part II — *Time Series Data Clustering*, illustrates a proposal for time series similarity search and detection (Chapter 3 and Chapter 4), and provides also research results of applying such technique in real domains (Chapter 5).

In Part III — *Uncertain Data Clustering*, it will be first described the state-of-the-art algorithms for this field (Chapter 6), and then the methodologies for handling uncertain data, which consist in a K-medoids based algorithm (Chapter 7) and in a centroid-based hierarchical agglomerative algorithm exploiting an information-theory based distance measure (Chapter 8). It will be also illustrated how to involve such methodology in clustering biomedical data (i.e., microarray) (Chapter 8, Section 8.6).

Part IV regards issues related to document clustering. In particular, two proposals will be presented, which are multi-topic text clustering exploiting generative models (Chapter 9), and a collaborative clustering of XML documents in a distributed environment (Chapter 10).

The thesis concludes by reviewing, in Chapter 10, the main contributions provided for mining complex data, and considering open problems and directions of future research.

2

Background

Summary. This chapter provides background notions and definitions about the main topics of this thesis. It starts with an insight into the problem of clustering. The basics of partitional, hierarchical, and density-based clustering approaches are given, along with a brief description of soft and model-based clustering algorithms. In addition, it will be shown the evaluation criteria used in the remainder of the thesis to assess the quality of clustering solutions. Afterward, the focus is putted on the main issues and requirements of the different types of complex data threated in this thesis. In particular, as regards time series, similarity search and dimensionality reduction problems are briefly reviewed. Concerning uncertain objects, the focus is on the various models used to represent uncertainty, and on the strategies to estimate the distance between uncertain data. The chapter ends by describing text data issues, with particular emphasis on text preprocessing and text representation models.

2.1 Data Mining and Clustering

In the first pages of this thesis, it has been illustrated the process of Knowledge Discovery in Databases (KDD) and its main steps. Among these steps, it has been underlined the relevance of *data mining*, since it provides a set of algorithms and methods to analyze data and to extract relevant *patterns*.

It results important to distinguish between two different data mining strategies, that are related to the type of data to analyze [JMF99]. In *supervised classification*, algorithms exploit *labeled* data, in the sense that each object in the dataset has been pre-classified and signed as belonging to a specific pattern or class. Within this view, a supervised algorithm aims to define a proper data model which is able to predict class label value by exploiting other features (i.e., attributes) of such data. The objective consists in assigning a class identifier to previously unseen records. Typically, a dataset of labeled objects is divided into two parts: the *training set* is used to build the model, while the *test set* is used to validate it. The first phase exploits learning algorithms and induction process to produce a model that fits with the

data belonging to the training set, and then, in the second phase, a deductive process validate the built model on the test set data. Supervised algorithms find application in many contexts, such as predictive tasks, classification and categorization problems [Mit97, GH88, NA08, VTM09].

It is also usual to deal with *unlabeled* data, i.e., data that do not have class labels. These data typically come from contexts in which patterns are unknown or not well-evident. In this case, it is not possible to use supervised algorithms; therefore, *unsupervised classification* field provides a set of algorithms that perform a *data driven analysis*, in the sense that unsupervised analysis employs solely data features, without considering any other external knowledge source and/or information.

Clustering is one of the most important unsupervised task. It aims to organize a given collection of unlabeled objects into meaningful groups, called *clusters* [JMF99, HK00, Har75, JD88, KR90, Ber06, GMW07]. The main objective of a clustering algorithm is to discover patterns and hidden structures within the data, and to highlight these new discovered information by means of a set of clusters, in such a way that objects belonging to the same cluster are highly "similar" each other, while objects belonging to different clusters are quite "dissimilar".

Attempting to draw a parallel between classification and clustering, it is possible to note that a classification algorithm is able to label unseen data according to a previous-built model, while a clustering task produces an "indirect labeling" of the data, since it can be derived from the clusters. In fact, clustering process finds relationships in the data and produces groups (i.e., clusters), each of which implies a specific pattern; for this reason, data within a clusters are "labeled", and the label is the specific pattern that they share.

There are several application domains that address clustering problem, especially the ones that need data exploration issues [BP03, JTJ04, WH09]. In fact, clusters underline hidden common features of a set of objects, providing both data summarization and data simplification.

Regardless of strategy and of specific algorithm used, it is possible to formalize clustering task as follows:

Definition 2.1 (Clustering). *Let $\mathcal{D} = \{o_1, \ldots, o_N\}$ be a dataset of N objects. A* clustering *task defines a grouping (i.e., clustering solution) $\mathcal{C} = \{C_1, \ldots, C_K\}, K \ll N$, of the objects in the dataset such that each cluster $C_i \subseteq \mathcal{D}, \forall u \in [1..K]$ and $C_i \cap C_j = \emptyset, \forall i, j \in [1..K], i \neq j$.*

Definition 2.1 refers to a generic clustering process that produces disjoint groups. However, it is possible to characterize clustering algorithms according to the adopted strategy. For this reason, there are different clustering approaches, which can be summarized in the taxonomy shown in Figure 2.1 [JD88]. From the figure, it is possible to note that there are many different clustering strategies. However, at the top level of the taxonomy, there are two principal clustering approaches: *partitional* and *hierarchical*. The main difference is that partitional methods produce as an output a single

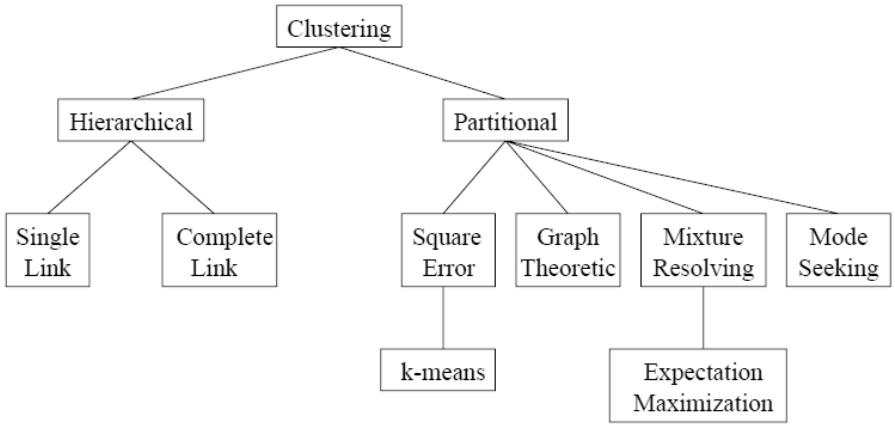


Fig. 2.1. The Knowledge Discovery in Databases (KDD) process

data partition, while hierarchical ones produce a collection of partitions, each of which is obtained at a different step of clustering process.

The following sections provide a detailed description of the most diffused clustering approaches, which have been involved in the next chapters of this thesis.

2.1.1 Partitional Clustering

Partitional clustering algorithms yield a flat grouping of the data, in the sense that the clustering process identifies a single data partitioning and, at each step of the algorithm, such partitioning is updated and refined. This strategy is simple and immediate, and it is particularly suitable for large datasets, since its time complexity is linear in the size of the dataset itself.

Partitional clustering algorithms can be classified according to the criterion function used during the partitional process. The most commonly used criterion in partitional clustering is the *squared error* [LC03]; within this view, algorithms aim to find a clustering solution that minimizes the sum of these errors calculated for each cluster. Formally, the squared error for a clustering solution $\mathcal{C}$ over a dataset $\mathcal{D}$ is:

$$e^2(\mathcal{C}, \mathcal{D}) = \sum_{i=1}^{K} \sum_{o_u \in C_i} |o_u - r_i|^2$$

where r_i denotes the *representative* of the cluster C_i.

Depending on the way the representative of a cluster is chosen, it can be distinguished between two other partitional clustering approaches. The most commonly used partitional clustering methods are the *centroid-based* ones. In such methods, each object o_u in the dataset to be partitioned is assigned to

a cluster according with its proximity from a particular point, that is called the *centroid* c_i of the cluster C_i.

The most common centroid-based clustering method is the K-means algorithm [McQ67]. The outline of the algorithm is shown in Algorithm 1.

Algorithm 1 K-means

Input: a dataset objects $\mathcal{D} = \{o_1, \ldots, o_N\}$; the number of output clusters K
Output: a set of clusters $\mathcal{C} = \{C_1, \ldots, C_K\}$

```
1: for i = 1 to K do
2:    c_i ←randomInitialize(D)
3: end for
4: repeat
5:    for all C_i ∈ C do
6:       C_i ← ∅
7:    end for
8:    for all o_u ∈ D do
9:       j ← argmin_{i∈[1..K]} dist(o_u, c_i)
10:      C_j ← C_j ∪ {o_u}
11:   end for
12:   for all C_i ∈ C do
13:      c_i ← updateCentroid(C_i)
14:   end for
15: until centroids do not change or a certain termination criterion is reached
```

The algorithm is substantially composed by three main phases. The first one is the *initialization step*, where initial centroids are randomly chosen in the hypervolume containing the objects in the dataset. The second phase is the *iteration step*, which consists in assigning each object to the cluster for which the distance between the object and the corresponding cluster centroid is minimum. This step requires the definition of a *distance function* used to calculate the proximity between objects and centroids. Although K-means algorithm has been adapted for handling several data types, it was originally proposed to deal with data objects with numerical features exploiting; for this reason, the *Euclidean distance* was originally employed as proximity measure in the algorithm. The third phase consists in the *updating step*, in which cluster centroids are updated according to the objects assigned to each cluster during the previous step. A typical strategy to compute a centroid of a cluster consists in averaging data features of the objects belonging to the cluster itself.

K-means follows a very intuitive clustering algorithm scheme. It is easy to implement and its temporal and spatial complexity is linear with respect to the dataset size ($\mathcal{O}(N)$); for this reason, it is particularly suitable for clustering large datasets. However, the centroid-based clustering paradigm has a number of drawbacks. In particular, it works well when clusters are well-separated and

compact, since it aims to minimize the intra-cluster variance but does not ensure to produce the global minimum variance clustering solution.

In order to overcome problems that affect centroid-based algorithms, *K-medoids* method has been proposed [KR87], which belongs to the family of the *medoid-based* clustering methods. It follows the same scheme and uses the same termination criteria of K-means. The only difference lies in a more appropriate choice for the cluster representatives, which are not fictitious objects (i.e., centroids) but correspond, for each cluster, to a real object within the cluster itself, i.e., medoids. This strategy has two advantages: it is suitable for representing objects with any type of features, and medoids are more robust to outliers, since the choice of each medoid is performed by considering the location of the predominant object fraction in the space edged by the cluster.

Independently from the way cluster representative are chosen, partitional clustering algorithms present some problems. The first drawback regards the choice of the number of desired clusters K; in fact, this is a user-defined parameter of the algorithm and its value may noticeably affect clustering quality. Although there are some strategies to consciously choose the cluster number [Dub87], this setting needs carefulness and requires a certain feeling with the data to be clustered. The second problem is related to the choice of the initial cluster representatives. This is another weakness point of partitional algorithms, which may lead to bad-quality clustering solutions. Also in this case, it is possible to resort to specific strategies for selecting initial cluster representatives [BF98, BM93].

2.1.2 Hierarchical Clustering

While partitional clustering methods provide a single partition of the objects, hierarchical clustering algorithms produce many partitions, that are nested into a hierarchical tree called *dendrogram* [Mur83]. Each level of the dendrogram stores a clustering solution derived from the one of the previous level and that is used to build the clustering solution of the next level. Figure 2.2 shows an example of a dendrogram.

Depending on the strategy used to build the dendrogram, hierarchical algorithms can be divided into two categories: *agglomerative* and *divisive* [JD88, KR90]. Algorithm 2 illustrates the scheme of a typical agglomerative hierarchical clustering (AHC) paradigm. They follow the *bottom-up* paradigm, since they start with single-object clusters. At each step, the two (or more) most appropriate clusters are merged, and the new clustering configuration represents the next level of the hierarchy. Divisive algorithms start from one single cluster containing all the dataset objects and iterate splitting the most appropriate cluster.

Both agglomerative and divisive approaches need to select, at each iteration of the algorithm, the most appropriate cluster(s) to be split or merged. Intuitively, this choice can be performed by analyzing the characteristics of the objects within each cluster; indeed, it is crucial to estimate the degree

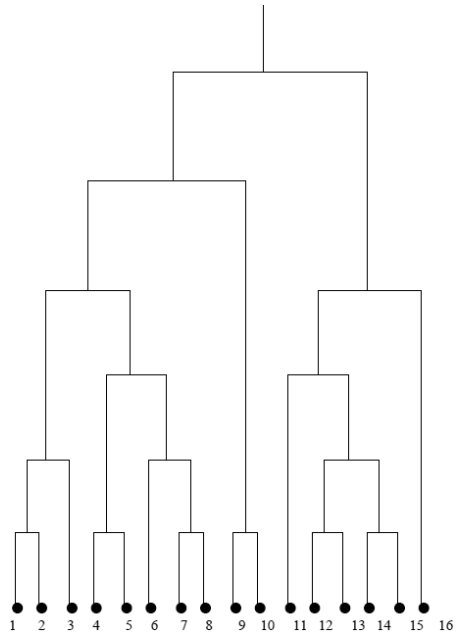


Fig. 2.2. An example of dendrogram for an cluster hierarchy built over a simple dataset of 16 elements

Algorithm 2 Agglomerative Hierarchical Clustering (AHC)

Input: a dataset objects $\mathcal{D} = \{o_1, \ldots, o_N\}$
Output: a clustering hierarchy $\mathbf{C}$ (i.e., a set of clustering partitions)
1: $\mathcal{C} \leftarrow \{C_1, \ldots, C_N\}$ such that $C_i = \{o_i\}, \forall i \in [1..N]$
2: $\mathbf{C} \leftarrow \{\mathcal{C}\}$
3: **repeat**
4: let C_i, C_j be the pair of most appropriate clusters in $\mathcal{C}$ to be merged
5: $\mathcal{C} \leftarrow \{C \mid C \in \mathcal{C}, C \neq C_i, C \neq C_j\} \cup \{C_i \cup C_j\}$
6: $\mathbf{C} \leftarrow \mathbf{C} \cup \{\mathcal{C}\}$
7: **until** $|\mathcal{C}| = 1$ or a certain termination criterion is reached

of similarity/dissimilarity among cluster objects. This proximity measure is called *linkage metric*. The type of linkage metric chosen significantly affects the entire clustering process.

The most used linkage metrics are based on inter-cluster measures [Mur83, Mur85, Ols95], which include *single-link*, *complete-link*, and *group-average*. These approaches differ each other in the strategy used to define the similarity between cluster pairs. Taking into account two clusters, single-link method calculates the minimum pairwise distance among data objects (one object taken form the first cluster and the other one taken from the second), while complete-link calculates the maximum one. Single-link strategy is able to de-

tect non-elliptical shape clusters, but suffers from a chaining effect and is to much sensitive to noise and outliers. On the contrary, complete-link strategy is more tolerant to noisy datasets, but is able to detect only compact tightly-bounded clusters. Group average linkage measures are a reasonable trade-off between single linkage and complete linkage methods. The most known group average methods are *average-link* and *centroid-based-link*. Average-link considers the mean value of the pairwise distances, while centroid-based-link (UPGMA — Unweighted Pair Group Method with Arithmetic mean) computes the distance between the cluster representatives.

All the above linkage measures can be derived from the Lance-Williams general formula [LW67], which produces several instances of linkage measures for agglomerative algorithms: given any two clusters C_i and C_j and a new cluster C_{ij} composed by merging C_i and C_j, the proximity measure between C_{ij} and any other existing cluster C_k can be expressed by a linear function of the proximities between C_k and the original clusters C_i and C_j. Formally,

$$p(C_{ij}, C_k) = \alpha_{C_i} p(C_i, C_k) + \alpha_{C_j} p(C_j, C_k) + \beta p(C_i, C_j) + \gamma |p(C_i, C_k) - p(C_j, C_k)|$$

This formula underlines the rigidity of the hierarchical clustering scheme, since a merging decision can not be revisited once it has been taken. Thus, optimal local merging decisions may lead to non-high-quality global clustering.

The major disadvantage of hierarchical clustering algorithms equipped with linkage metrics is the time complexity, which is quadratic in the number of the objects in the dataset ($\mathcal{O}(N^2)$). However, there are some cases in which ad-hoc strategies can be used to perform more efficient clustering executions; in particular, when the *connectivity matrix* (i.e., the $N \times N$ matrix that stores all the pairwise distances between objects in the dataset) is sparsified, hierarchical divisive clustering can exploit graph partitioning strategies [JD88], such as the Minimum Spanning Tree (MST) algorithm.

However, despite these well-known drawbacks, hierarchical clustering approaches are widely used; the two main advantages of these algorithms are the ability of dealing with any attribute type and the production of clustering solutions of different levels of granularity,

2.1.3 Density-based Clustering

Partitional clustering techniques suffer from an accuracy issue when data objects do not follow gaussian distribution. In particular, there are many cases in which clusters have irregular shapes. Figure 2.3 shows an example of dataset in which partitional algorithms are not able to identify a high-accuracy clustering solution.

In order to overcome these issues, *density-based clustering* algorithms have been proposed. They start from the fact that an open set in an Euclidean space can be divided in a number of connected components. Therefore, it is crucial

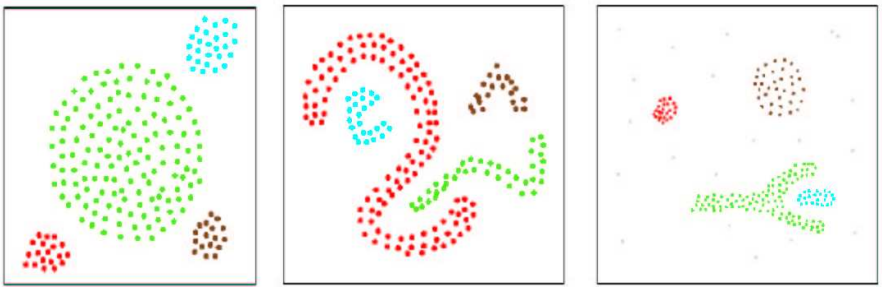


Fig. 2.3. Examples of datasets in which partitional algorithms are not able to identify high-quality clustering solutions

to determine base concepts for density-based clustering methods, which are *density*, *connectivity*, and *boundary*.

Density-based algorithms start form the idea that a cluster is determined by a *dense* set of objects in the dataset. It is built incrementally by starting from an initial point and including, at each step, a set of neighbor objects; in this way, each cluster grows in the direction of the density region. Since density-based algorithms need a metric space to be applied, they are particularly suitable for clustering spatial data [HK00, Kol01]

Density-based clustering algorithms perform well in particular contexts; indeed, it has been shown that these approaches are able to detect clusters of any shape. The strategy adopted to build clusters makes these algorithms very robust to outliers, allowing them to handle noisy data. Moreover, they scale very well.

Nevertheless, there are also several drawbacks for density-based clustering approaches. First, there may be clusters with not equally-distributed density; in particular, inside a single dense cluster it is possible to have two more data subregions with different densities. Second, clusters may be difficult to interpret, since they are composed by connected objects that carry a great variety of feature values within the cluster. This aspect may affect pattern identification and cluster characterization.

It is possible to distinguish between two families of density-based clustering algorithm, and this classification is dictated by the approach used in the definition of the density function. Indeed, the first one exploits *data point*, while the second one takes into account the *underlying attribute space*. The most representative algorithms for the first class are *DBSCAN*, *GDBSCAN*, and *OPTICS*, while *DENCLUE* belongs to the second algorithm class.

Density functions based on data points

DBSCAN (Density Based Spatial Clustering of Application with Noise) [EKSX96] is the most representative density-based clustering algorithm. Its outline is shown in Algorithm 3.

Algorithm 3 DBSCAN

Input: a dataset objects $\mathcal{D} = \{o_1, \ldots, o_N\}$;
a minimum distance ϵ;
a minimum number of neighbors $minPts$
Output: a set of clusters $\mathcal{C}$

```
1: for all o_u ∈ D such that o_u has not been visited do
2:    mark o_u as visited
3:    S_{o_u} ← getNeighbors(o_u,ε)
4:    if |S_{o_u}| ≥ minPts then
5:       C ← {o_u}
6:       expandCluster(C, S_{o_u}, ε, minPts)
7:       C ← C ∪ {C}
8:    else
9:       mark o_u as OUTLIER
10:   end if
11: end for
```

It requires two input parameters, which are the radius size (i.e., ϵ) and the minimum number of neighbors (i.e., $minPts$). The algorithm starts picking up an unvisited object o_u in the dataset and calculates its neighborhood (i.e., S_{o_u}), which depends on ϵ. If the neighborhood contains a number of objects equal or grater than $minPts$, then a new cluster is created. It is composed by the current object o_u and by all the other objects that are "connected" to it. To find these other objects, the function *expandCluster* performs, for each of the neighborhood objects, the same outline described above. The overall execution time complexity of the algorithm is $\mathcal{O}(N \times \log(N))$.

GDBSCAN [SEKX98] is a variant of DBSCAN that exploits other measures to define neighborhoods in spatial dataset. In fact, many spatial databases contain complex objects, which can be represented in a space by means of geometrical figures (e.g., polygons) instead of a single point. For this reason, other measures can be employed in the neighborhoods definition process.

Effectiveness and efficiency of DBSCAN are both highly-related to the two parameters of the algorithm, i.e., ϵ and $minPts$. Otherwise, there are no ways to determine them from the data to be clustered. This aspect is particularly relevant in presence of regions with different density, since each of them require a proper parameter setting to be correctly identified. To overcome this issue, OPTICS [ABKS99] algorithm orders dataset objects and varies ϵ parameter in order to cover a spectrum of radii $\epsilon' \leq \epsilon$. To achieve this objective, OPTICS stores additional information for each object, that are the *core-distances* and the *reachability-distances*.

There have been also developed density-based clustering algorithms that do not need user defined parameters. DBCLASD (Distribution Based Clustering of Large Spatial Databases) [XEKS98] uses the χ^2-test to discover object distributions in a dataset. Although this algorithm does not require input pa-

rameters, it suffers from an efficiency issue, since it is 2-3 time slower than DBSCAN.

Density functions based on underlying attribute space

The other family of density-based algorithms exploits the underlying attribute space to group objects. DENCLUE (DENsity-based CLUstEring) [HK98] uses a *density function*, which is a superposition of several *influence functions* over the underlying attribute space. The influences of data instances in data space are modeled via a simple *kernel function* (e.g., the Gaussian kernel). DENCLUE aims to discover local maxima of such density function, which are called *density-attractors*. A hill-climbing-like procedure is started over the objects in the dataset. It assigns each object to a local maximum and uses the gradient technique to estimate the maximum density growth. DENCLUE has a parameter ξ, which filters out points with local density value lower than a certain threshold. This aspect makes the algorithm robust in presence of noisy data.

2.1.4 Soft Clustering

Traditional clustering algorithms perform *hard* strategy, since each object in the dataset is assigned to only one cluster. Therefore, hard clustering produces disjoint groups. There are many contexts in which an object may belong to more than one cluster, maybe with different membership degrees. *Fuzzy clustering algorithms* produce *soft* object assignments, since each object in the dataset is assigned to more than one cluster. Fuzzy algorithms use a proper *membership function* [Zad65], which assigns a membership degree of objects to clusters.

There have been proposed many fuzzy versions of popular hard clustering algorithms. In particular, *fuzzy c-means* (FCM) and *fuzzy c-medoids* (FCMdd) [Bez81] are fuzzy variants of the two most popular partitional clustering algorithms. These fuzzy algorithms perform better than the corresponding hard clustering ones in terms of accuracy, since they are able to avoid most (but not all) of the local minima.

2.1.5 Model-based Clustering

Clustering methods can be divided into two categories [ZG03]: *discriminative* and *generative* approaches. Traditional clustering methods are discriminative (distance-based) approaches, since they require the computation of distance/similarity between objects to group them into clusters.

Recently, there has been a growing interest in model-based clustering methods [ZG03, CGS00], which attempt to discover and learn generative models within the data. These new perspective moves the focus of the clustering

process from objects to clusters. In fact, while in traditional clustering algorithms objects are grouped according to their similarity, model-based clustering methods assume each cluster to be defined by a proper *model*, and each object of the dataset is assigned to the best-fitting cluster. Typically, model structure is represented by means of a *Hidden Markov Model* (HMM) [Smy96, CGS00]. Model-based clustering strategy is based on the assumption that data objects are generated by a mixture of underlying probability distributions, and each cluster is expression of one of these mixtures by grouping together data objects that are responsible of generating such specific mixture component. Model parameters are typically estimated by using the *Expectation-Maximization* (EM) [DLR77] algorithm, which iteratively assigns values to probability distribution variables until a certain function is maximized (i.e., the *likelihood*).

There are two major families of model-based clustering algorithms [HK00]. *Statistical approaches* use probabilistic measurements for clustering objects and representing data groups by means of probabilistic descriptions. The most representative algorithms are COBWEB, CLASSIT and AutoClass [Fis87, GLF89, CS96]. The other family of algorithms is based on *neural networks*. The best known approach is *Self-Organizing Map* (SOM) [Koh82], which provides a projection of the M-dimensional input space onto a lower-order one. When involved in clustering task, the mapping produced by SOM identifies clusters, in the sense that it induces an object grouping according to their similarity (in an Euclidean sense) in the original space.

2.1.6 Evaluation Criteria

Clustering is an unsupervised task. Indeed, it is crucial to define measures which evaluate the quality of a clustering solution. It is possible to resort to two different strategies to evaluate the effectiveness of a clustering task. The first one exploits *internal criteria*, while the second one uses *external criteria* [HBV01, HBV02].

Internal criteria are the most intuitive way to evaluate clustering solutions. They start from the general purpose of clustering task, that is to achieve cluster cohesiveness (i.e., objects in the same cluster should be highly-similar each other) and cluster separation (i.e., lowly-similar objects should belong to different clusters).

External criteria can be used when further information are available for the data to be clustered. By means of these measures, it is possible to define how well a clustering solution agrees with a certain predefined scheme of known classes, which typically refer to *natural clusters*. It is possible to resort to IR field to find evaluation criteria suitable for clustering.

In the following, it will be provided the definitions of the most popular internal criteria (i.e., intra-distance and inter-distance, and Cophenetic correlation coefficient) and external criteria (i.e., F-measure, Entropy, and Error Rate).

Internal Criteria

Intra-distance and Inter-distance

The two major internal criteria used for clustering evaluation are *intra-distance* and *inter-distance*. Given a collection $\mathcal{D}$ of objects, let $\mathcal{C} = \{C_1, \ldots, C_K\}$ be the output partition yielded by a clustering algorithm. The intra-distance of any cluster $C \in \mathcal{C}$ is calculated as the mean of the pair-wise distances between the objects in C, whereas the inter-distance between any pair of clusters $<C_i, C_j> \in \mathcal{C}, \forall i, j \in [1..K], i < j$, is computed as the average of the pair-wise distances between the objects in C_i and the objects in C_j. The overall intra-cluster ($intra(\mathcal{C})$) and inter-cluster ($inter(\mathcal{C})$) distances of a clustering solution $\mathcal{C}$ are calculated by averaging the local values. Formally,

$$intra(\mathcal{C}) = \frac{1}{|\mathcal{C}|} \sum_{C \in \mathcal{C}} \frac{2}{|C|(|C|-1)} \sum_{o_u \in C} \sum_{o_v \in C, v>u} dist_E(o_u, o_v)$$

$$inter(\mathcal{C}) = \frac{2}{|\mathcal{C}|(|\mathcal{C}|-1)} \sum_{<C_i, C_j> \in \mathcal{C} \times \mathcal{C}, i<j} \frac{1}{|C_i||C_j|} \sum_{o_u \in C_i} \sum_{\o_v \in C_j} dist_E(o_u, o_v)$$

where $dist_E(o_u, o_v)$ computes the Euclidean distances between o_u and o_v. The overall quality $Q(\mathcal{C})$ of a clustering solution $\mathcal{C}$ can be defined by taking into account intra-distance and inter-distance values. Indeed,

$$Q(\mathcal{C}) = inter(\mathcal{C}) - intra(\mathcal{C})$$

It is easy to note that the lower $intra(\mathcal{C})$ and the higher $inter(\mathcal{C})$, the better the clustering quality $Q(\mathcal{C})$.

Cophenetic Correlation Coefficient

Another popular internal criterion is the *cophenetic correlation coefficient* [SR62]. It is mainly used in hierarchical clustering contexts to evaluate the compactness of the obtained clustering solution. Indeed, this measure (ranging within $[-1, 1]$) aims to evaluate a dendrogram produced by any hierarchical algorithm according to how it preserves the pairwise distances between the original data points. Intuitively, the higher the cophenetic correlation value for a dendrogram, the higher the compactness and the better the quality achieved by the hierarchical algorithm.

Formally, let $\mathcal{D} = \{o_1, \ldots, o_N\}$ be a dataset of N objects and let $\mathbf{C}$ be the dendrogram solution produced by a hierarchical clustering algorithm (i.e., a hierarchy of clusters). The *cophenetic correlation coefficient* ($CPCC(\mathcal{D}, \mathbf{C})$) is defined as

$$CPCC(\mathcal{D}, \mathbf{C}) = \frac{\sum_{u<v}(d_E(o_u, o_v) - \overline{d}_E)(t(o_u, o_v) - \overline{t})}{\sqrt{\left[\sum_{u<v}(d_E(o_u, o_v) - \overline{d}_E)^2\right]\left[\sum_{u<v}(t(o_u, o_v) - \overline{t})^2\right]}}$$

for all $u, v \in [1..N]$. In the formula above, $d_E(o_u, o_v)$ denotes the Euclidean distance between the objects o_u and o_v, while $t(o_u, o_v)$ is the dendrogrammatic distance of such objects, which indicates the level of the dendrogram at which the objects o_u and o_v are first joined together. The values $\overline{d}_E$ and $\overline{t}$ represent the average of the $d_E(o_u, o_v)$ and the average of the $t(o_u, o_v)$, respectively.

External Criteria

F-*measure*

As regards external criteria, the most popular one is F-*measure* [vR79, DB75], which is defined as the harmonic mean of the basic notions of *precision* and *recall*.

Given a collection $\mathcal{D}$ of objects, let $\Gamma = \{\Gamma_1, \ldots, \Gamma_H\}$ be the reference classification of the objects in $\mathcal{D}$, and $\mathcal{C} = \{C_1, \ldots, C_K\}$ be the output partition yielded by a clustering algorithm. *Precision* of cluster C_j with respect to class Γ_i is the fraction of the objects in C_j that has been correctly classified, whereas *Recall* of cluster C_j with respect to class Γ_i is the fraction of the objects in Γ_i that has been correctly classified. Formally,

$$\mathrm{P}_{ij} = \frac{|C_j \cap \Gamma_i|}{|C_j|} \qquad \mathrm{R}_{ij} = \frac{|C_j \cap \Gamma_i|}{|\Gamma_i|}$$

For each pair (C_j, Γ_i), the F-measure F_{ij} is computed as:

$$\mathrm{F}_{ij} = \frac{2\mathrm{P}_{ij}\mathrm{R}_{ij}}{\mathrm{P}_{ij} + \mathrm{R}_{ij}}$$

In order to score the quality of $\mathcal{C}$ with respect to Γ by means of a single value, the overall F-measure $\mathrm{F}(\mathcal{C}, \Gamma)$ is computed according with the *micro-averaging* strategy, which uses the weighted sum of the maximum F_{ij} score for each class Γ_i.

$$\mathrm{F}(\mathcal{C}, \Gamma) = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{H} |\Gamma_i| \max_{j \in [1..K]} \mathrm{F}_{ij}$$

Alternatively, overall F-measure can be computed according with the *macro-averaging* strategy, which computes first the overall values for precision and recall. Formally,

$$\mathrm{P} = \frac{\sum_{i=1}^{H} \max_{j \in [1..K]}\{\mathrm{P}_{ij}\}}{\max\{K, H\}} \qquad \mathrm{R} = \frac{\sum_{i=1}^{H} \max_{j \in [1..K]}\{\mathrm{R}_{ij}\}}{H}$$

$$\mathrm{F} = \frac{2\mathrm{PR}}{\mathrm{P} + \mathrm{R}}$$

Micro-averaging and macro-averaging strategies may lead to quite different results; this situation occurs when classes are not balanced in terms of object number. However, there is no a common agreement that establishes the most reliable strategy, since the choice depends on the application requirements. Anyway, micro-averaging is a good choice for non specific context, since it weights objects proportionally to the size of the cluster they belong to.

The above definitions for F-measure are derived from a most general formula. It is defined as:

$$\mathrm{F}_\beta = \frac{(1+\beta^2)\mathrm{PR}}{\beta^2\mathrm{P}+\mathrm{R}}$$

F_β has a non-negative real value β, which is used to bias the formula with respect to precision or recall. Indeed, the higher β, the more important recall than precision. Above formula consider equal-weighted precision and recall; for this reason, they define *balanced* F*-measure* or simply F_1*-measure*.

Entropy

Another common external validity measure is *entropy*. For each cluster $C_j \in \mathcal{C}$, the class distribution of data is computed as the probability $p(\Gamma_i|C_j)$ that an instance in C_j belongs to class Γ_i. Using this class distribution, the normalized entropy of C_j is computed as

$$\mathrm{E}_j = -\frac{1}{\log H}\sum_{i=1}^{h}[p(\Gamma_i|C_j)\log\big(p(\Gamma_i|C_j)\big)]$$

where $p(\Gamma_i|C_j) = |C_j \cap \Gamma_i|/|C_j|$ if $|C_j \cap \Gamma_i| \neq 0$, otherwise $p(\Gamma_i|C_j) = 0$. The overall entropy ($\mathrm{E} \in [0..1]$) is defined as the sum of the individual cluster entropies weighted by the size of each cluster:

$$\mathrm{E} = \frac{1}{|\mathcal{D}|}\sum_{j=1}^{k}(|C_j| \times \mathrm{E}_j).$$

Error Rate

The *Error Rate* ($\mathcal{E}$) is another popular external validity measure used to compare two classifiers. It takes into account both errors of commission and errors of omission:

$$\mathcal{E} = \frac{1}{2\times|\mathcal{D}|}\sum_{i=1}^{H}(|C_i \setminus \Gamma_i| + |\Gamma_i \setminus C_i|)$$

where $\mathcal{D}$ is a set of objects, $\Gamma = \{\Gamma_1, \ldots, \Gamma_H\}$ is the expected organization of the objects in $\mathcal{D}$, and $\mathcal{C} = \{C_1, \ldots, C_K\}$ is the output of a classification or clustering algorithm.

The *accuracy* of a classifier is defined in terms of the error rate as

$$\mathcal{A} = 1 - \mathcal{E}$$

Note that $\mathcal{A}$ ranges between zero and one; in particular, higher value of $\mathcal{A}$ indicates better quality.

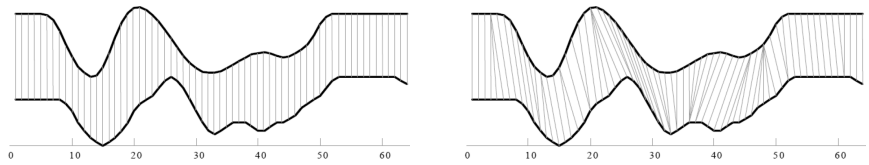
2.2 Time Series

A *time series* is the monodimensional case of spatio-temporal data. It consists in a sequence of (real) numeric values upon which a total order based on timestamps is defined. Time series are generally used to represent the temporal evolution of objects, hence enormous amounts of such data are naturally available from several sources of different domains, including speech recognition, medicine and biology measurement, financial and market data analysis, telecommunication and telemetry, sensor networking, motion tracking, meteorology, and so on.

Most research on time series data management and knowledge discovery has been devoted to the similarity search and detection problem, which arises in many tasks such as indexing and query processing, change detection, frequent pattern mining, classification, and clustering. In this thesis, we refer to clustering and classification as evaluation frameworks for similarity detection. In particular, we focus on the clustering task as it is necessary when the data being organized are not associated with predefined categories, which is a very frequent context in real-world application domains. Clustering of time series data has been attracting a growing interest in several scenarios. For instance, in the biomedical domain, frequently posed problems include finding groups of genes with similar expression profiles across a number of experiments, organizing patients according to different healthy/disease conditions, and finding groups of similar functional activities of the human brain in response to a given stimulus. In socio-economics domain, clustering energy/power consumption patterns can support applications of fraud detection. Other challenging scenarios involve, for instance, seasonality patterns of retail data, personal income data, models of ecological dynamics, multimedia data streams. A more exhaustive list of applications which demand for time series clustering can be found in [War05].

2.2.1 Similarity Search

The goal of clustering analysis in time series data domain is to discover similarities to group together series with similar trends. A first simple approach to compare two time series objects is a mere application of the Euclidean distance, which computes the distance between two time series by averaging the point-to-point distance of every data point in the first series with the



(a) (b)

Fig. 2.4. Two time series aligned according to (a) Euclidean norm and (b) DTW

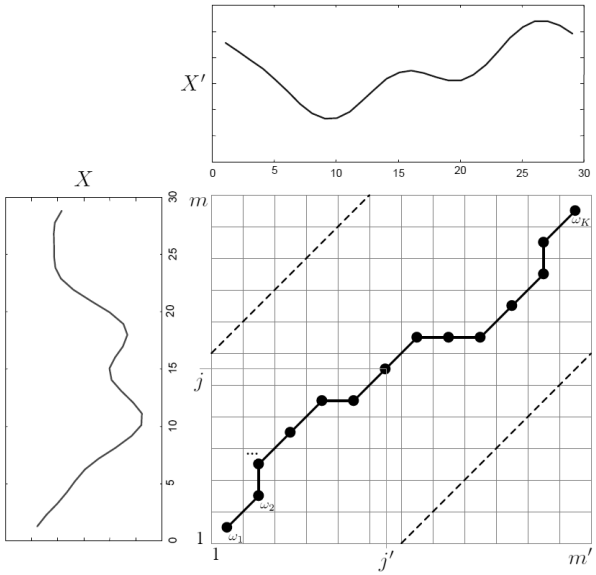


Fig. 2.5. An example warping path

corresponding one in the second. However, despite of its simplicity, this measure does not work well in several situations, such as amplitude scaling time shifting.

A better way to compare two time series is "warping" the time axis in order to achieve an alignment between the data points of the series. The Dynamic Time Warping (DTW) algorithm has long been known in speech recognition [RJ93], and shown to be an effective solution for measuring the distance between time series [BC94].

Indeed, unlike the Euclidean norm, DTW allows elastic shifting of a sequence to provide a better alignment, thus it can handle time series with local time shifting and different lengths. Figure 2.4 shows an example of alignments provided by Euclidean norm and DTW, whereas Figure 2.5 illustrates how a warping path is built.[1]

[1] Figures 2.5 and 2.4 are borrowed from [KP01].

Although DTW represents a valid technique which is able to handle the most common cases related to the time series data similarity detection domain, it may suffer from the production of "singularities", i.e., alignments of a single point of a series with multiple points of another series. To avoid these situation, DDTW has been proposed [KP01], which is the derivative version of DTW.

There are also several alternative proposals to DTW, which are based on *edit distance*, but they are too less efficient and less accurate than DTW based approaches [VGK02, CN04b, CÖO05].

2.2.2 Dimensionality Reduction

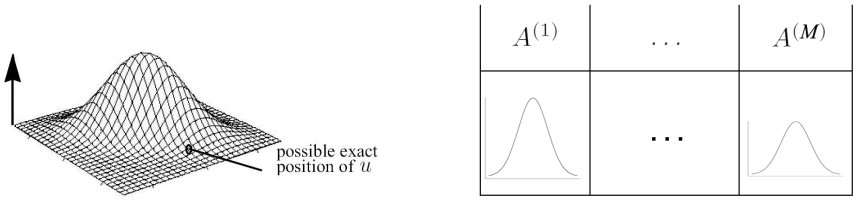
Besides the similarity problem in time series, another issue concerns the high dimensionality characterizing time series data in many application domains. Dimensionality reduction methods are useful for modeling time series into a more compact form. However, while this can help to compare time series efficiently, dimensionality reduction methods may lose significant information about the main trends in a time series, which are essential to effective similarity detection. Within this view, it is desirable to have methods which can perform high data compression while preserving main features in a series and underlying relevant trends.

There are two main approaches to perform dimensionality reduction task over a time series. The first way consists in approximating a continuous time series with piecewise discrete function (i.e., *segmentation*) [CF99, WAA00, Bri90, PH74, KP98, KCPM01, KP00, YF00, CKMP02, LKLC03], the second approach aims to perform dimensionality reduction via low-order continuous functions [KJF97, KAS98, RM98, RM97, CN04a, MH03].

2.3 Uncertain Objects

In recent years there has been a growing interest in clustering *uncertain data*. In contrast to traditional, "sharp" data representation models, uncertain data objects can be represented in terms of an uncertainty region over which a probability density function (pdf) is defined. Handling uncertainty in data management has been requiring more and more importance in a wide range of application contexts. Indeed, data uncertainty naturally arises from, e.g., implicit randomness in a process of data generation/acquisition, imprecision in physical measurements, and data staling.

Dealing with such data has raised several issues in data management and knowledge discovery, mainly due to the intrinsic difficulty underlying the various notions of uncertainty. In particular, traditional data management and mining approaches are designed to work on "deterministic" data, which are assumed to describe object properties in a precise way. However, such data usually come from real-world domains, and in most cases the data values are obtained by measurement operations that are typically affected by errors to a



(a) (b)

Fig. 2.6. Graphical representation of (a) a multivariate uncertain object and (b) a univariate uncertain object

certain degree (e.g., approximation, sampling). Therefore, there is a real demand for the development of methodologies and techniques which are able to suitably model uncertainty in data objects in order to make them amenable to further processing and analysis tasks.

2.3.1 Modeling Uncertainty

Various notions of uncertainty have been defined depending on the application domain (e.g., [IL84, AKG87, Sad91, LLRS97, DS04, GT06, Agg07]). In general, uncertainty can be considered at table, tuple or attribute level [TXC07], and it is usually specified by fuzzy models [GUP06], evidence-oriented models [Lee92, LSS96], or probabilistic models [SBHW06].

The focus here is on data containing *attribute-level* uncertainty, which is generally modeled according to a probabilistic model. We hereinafter refer to this data as *uncertain objects*. An uncertain object is usually represented by means of *probability distributions*, which describe the likelihood that the object appears at each position in a multidimensional space [CKP03, KP05a, CCKN06], rather than by a traditional vectorial form of deterministic values. Uncertainty information can also be present in data according to other forms, such as statistical properties (e.g., error percentage or deviation from an expected value); however, while these properties provide a concise information about an uncertain set of values, probability distributions offer a more-accurate solution in uncertainty modeling.

Attribute-level uncertainty expressed by means of probabilistic models is present in several application domains, such as sensor measurements, moving objects representation, and gene expression data [CKP03, CLL06, FGB02, DGM$^+$05, LHY04, MFNL03b, HRC$^+$05b, LMLR05b]. In order to manage uncertain objects with attribute-level uncertainty, in the following there will be introduced the two basic models generally used for representing uncertain objects, namely *multivariate uncertainty* and *univariate uncertainty* models.

Figure 2.6 illustrates the idea behind multivariate and univariate modeling of uncertain objects. In a multivariate uncertainty model, an M-dimensional uncertain object is defined in terms of an M-dimensional region and a multivariate probability density function, which stores the probability according

to which the exact representation of the object coincides with each point in the region. In a univariate uncertainty model, an M-dimensional uncertain object has, for each attribute, an interval and a univariate probability density function that assigns a probability value to each point within the interval. Depending on the specific application context, an uncertain object can be modeled according to one of the two models.

It should be noted that, in general, it is not straightforward to map multivariate objects into univariate objects, and vice versa. Indeed, deriving a multivariate representation from a univariate one cannot be performed without knowing in advance the conditional pdfs or making specific statistical assumptions (e.g., statistical independence). On the other hand, transforming a multivariate model into a univariate one requires the computation of the marginal pdfs from the joint distribution; this can be a very complex and inefficient operation depending on the form and/or the dimensionality of the distribution.

Definition 2.2 (multivariate uncertain object). *A* multivariate uncertain object *o is a pair* (R, p)*, where* $R = [L^{(1)}, U^{(1)}] \times \cdots \times [L^{(M)}, U^{(M)}]$ *is the* m*-dimensional region in which o is defined and* $p : \Re^M \rightarrow \Re_0^+$ *is the probability density function of o at each point* $\mathbf{x} \in \Re^M$*, such that:*

$$\int_{\mathbf{x} \in \Re^M \setminus R} p(\mathbf{x}) \mathrm{d}\mathbf{x} = 0 \tag{2.1}$$

$$p(\mathbf{x}) > 0, \quad \forall \mathbf{x} \in R \tag{2.2}$$

Definition 2.3 (univariate uncertain object). *A* univariate uncertain object *o is a tuple* $(A^{(1)}, \ldots, A^{(M)})$*. Each attribute* $A^{(h)}$ *(with* $h \in [1..M]$*) is a pair* $(I^{(h)}, p^{(h)})$*, where* $I^{(h)} = [L^{(h)}, U^{(h)}]$ *is the interval of definition of* $A^{(h)}$*, and* $p^{(h)} : \Re \rightarrow \Re_0^+$ *is the probability density function that assigns a probability value to each* $x \in \Re$*, such that:*

$$\int_{x \in \Re \setminus I^{(h)}} p^{(h)}(x) \mathrm{d}x = 0 \tag{2.3}$$

$$p^{(h)}(x) > 0, \quad \forall x \in I^{(h)} \tag{2.4}$$

The definitions above involve the region/intervals of definition for the pdf(s) associated to any uncertain object. This represents a key aspect, since in many real applications these objects are defined according to limited region/intervals. Moreover, this allows for defining an uncertain object in a more general way, since the case of uncertain object with unlimited region/intervals is also implicitly taken into account in these models.

It should also be noted that Definitions 2.2–2.3 refer to the most general case of uncertain objects modeled as continuous random variables and

described by probability density functions. Nevertheless, without loss of generality, we can state that these definitions include the discrete case. Indeed, for any discrete multivariate uncertain object $o = (R, p)$, the M-dimensional region $R = [L^{(1)}, U^{(1)}] \times \cdots \times [L^{(M)}, U^{M)}]$ bounds a discrete set S of M-dimensional points, and the probability distribution function p is a multivariate discrete pdf, also known as multivariate *probability mass function*, defined as $p : S \rightarrow \Re_0^+$. Analogously, for any discrete univariate uncertain object $o = (A^{(1)}, \ldots, A^{(M)})$, the h-th attribute ($h \in [1..M]$) is defined over an interval $I^{(h)}$ which bounds a discrete set $S^{(h)}$, and has a function $p^{(h)} : S^{(h)} \rightarrow \Re_0^+$ which is a univariate probability mass function.

In the following, we assume statistical independence between any actual location $\mathbf{x}_u, \mathbf{x}_v \in \Re^M$ of any two multivariate/univariate uncertain objects o_u, o_v belonging to the same dataset $\mathcal{D}$, i.e., we assume that:

$$\Pr(o_u \equiv \mathbf{x}_u \cap o_v \equiv \mathbf{x}_v) = \Pr(o_u \equiv \mathbf{x}_u)\ \Pr(o_v \equiv \mathbf{x}_v),\ \ \forall o_u, o_v \in \mathcal{D},\ \forall \mathbf{x}_u, \mathbf{x}_v \in \Re^M$$

where $o \equiv \mathbf{x}$ denotes the event "*the actual location of uncertain object o is in the point* $\mathbf{x} \in \Re^M$".

Also, for any univariate object $o = ((I^{(1)}, p^{(1)}), \ldots, (I^{(M)}, p^{(M)}))$, we assume statistical independence among its M dimensions:

$$\Pr(o \equiv \mathbf{x}) = \prod_{h=1}^{M} p^{(h)}(x_h)$$

for each point $\mathbf{x} = [x_1, \ldots, x_M] \in \Re^M$.

2.3.2 Distance Measures

Dealing with uncertain objects has raised several issues in data management and knowledge discovery; in particular, organizing uncertain objects is challenging due to the intrinsic difficulty underlying the various notions of uncertainty. As a consequence to this challenge, *clustering uncertain objects* has been attracting increasing interest in recent years (e.g., [KP05b, KP05a, CCKN06, NKC+06, LKC07]).

While the proposed clustering algorithms mainly differ on the clustering strategy and the cluster model, the adopted notions of distance between uncertain objects come into two main approaches. The first approach consists in computing the distance between aggregated values extracted from the probability distributions of the uncertain objects (e.g., expected values); the second approach instead requires to somehow compare the whole distributions. However, both these approaches have some drawbacks in their own: as stated in, e.g., [KKPR05], the first approach has an accuracy issue, whereas the second one may suffer from expensive operations for approximating the probability distributions of the uncertain objects.

Information-theory represents a fruitful research area to devise distance measures for comparing probability distributions. However, most of the existing information-theoretic measures, such as the popular ones falling into the Ali-Silvey class of distance measures [AS66], cannot be used to directly define distances for uncertain objects. Indeed, such measures may have a number of shortcomings (e.g., do not satisfy the symmetric property, do not range within a bounded interval, etc.) that limit their applicability. Most importantly, information-theoretic measures commonly require that the probability distributions being compared hold for random variables defined over a common event space (i.e., common domain region); unfortunately, the domain regions of the probability distributions associated to uncertain objects may not have wide intersections.

2.4 Text Data

In recent years, there has been a large diffusion of digital information, especially in the contexts of World Wide Web (WWW) and Internet. This process leads to a massive proliferation of many communication channels, such as e-mail, forum, and digital libraries. Typically, data generated in these scenarios are *textual*, since they are composed by a set of words (or terms), phrases, and paragraphs. Therefore, term *text* is often used to refer to *document*, which denotes a single unit of textual information [HNP05, Tag05].

The process of extracting new information and patterns from text data is known as *Knowledge Discovery form Text* (KDT) [FD95]. It is possible to easily draw a parallel between KDD and KDT, since KDT follows the same tasks of KDD, but it is applied to text data. It is straightforward that KDT has to face with several specific domain issues of handling text data. The main problems are related to the absence (or the poor presence) of any kind of structure within these data. The main difference between KDT and KDD regards the way data are represented. Indeed, preprocessing phase plays a crucial role in KDT, and it usually relies on experiences and results obtained by other related research areas, such as *Information Retrieval (IR)* [SWY75, SJW97, Hea99], *Natural Language Processing (NLP)* [Abn91, MS99, Kod99], and *Information Extraction (IE)* [Wil97].

The way text data are represented is crucial in KDT. However, it has been shown how this step is challenging and how many aspects have to be considered to produce a structured while high-informative representation. In fact, since various levels of text structuring can be identified, i.e., *non-structured* (plain) texts and *semi-structured* texts (e.g., HTML and XML), there are many techniques which try to equip document modeling with structural and semantic aspects. Nevertheless, traditional approaches use only syntactics, since they are based on the *bag-of-words* model, which represents a text document by means of a set of words.

2.4.1 Text Preprocessing

The bag-of-words model needs a proper preprocessing phase to filter out from the text elements that are not relevant and to properly transform tokens to exploit all their informative content.

The first phase of text preprocessing is the *tokenization* step, which aims to transform a text into a set of "textual units". Tokenization takes the text document as an input and produces a *dictionary* of basic textual units by removing non-informative contents (e.g., punctuation marks) and replacing other unknown symbols and characters with single white spaces. More formally, tokenization step takes the dataset $\mathcal{D} = \{d_1, \ldots, d_N\}$ of composed by N documents (i.e., a *corpora*) and produces a vocabulary $\mathcal{W} = \{w_1, \ldots, w_M\}$ of M terms over $\mathcal{D}$.

Typically, the size of the vocabulary after tokenization is very large. In order to reduce this size, a set of further preprocessing operations can be applied, in such a way that the vocabulary contains only the most representative terms in the document collection.

Filtering operations can be performed to reduce vocabulary size. The most known approach is the *stop words removal*. This operation filters out from the vocabulary all the terms that do not provide further content information, such as conjunctions, articles, prepositions, etc.

Lemmization techniques aim to uniform term occurrences by mapping all the terms in their canonical form, i.e., verbs to the infinite tense and nouns to the singular form. Since all terms must be known and identified, it is should be performed a preliminary step, which consists in tagging the part of speech of each term.

Stemming operation consists in representing a set of terms by their common radix (i.e., the *stem*). This technique allows to map terms with similar/equal meaning into a unique stem. The most known stemming algorithm has been defined by Porter, which uses a set of production rules to iteratively transform terms into their stemmed version.

Linguistic Preprocessing

Preprocessing techniques described above perform basic operations in order to achieve standard text preprocessing. However, these techniques define preprocessing strategies that exploits only syntactic operations over terms. In some cases, it can be useful to enrich text preprocessing by executing additional linguistic tasks, which are able to enhance information about terms and to produce a higher-quality preprocessing. The most-frequently used linguistic preprocessing operations are *Part-Of-Speech tagging (POS)*, *text chunking*, *Word Sense Disambiguation (WSD)*, and *parsing* (an exhaustive discussion about linguistic preprocessing operations can be found in [MS99]). However, in many text mining tasks, bag-of-words and basic preprocessing produce good results, especially if enhanced with semantic facilities [HSS03].

2.4.2 Text Modeling

To analyze documents, it is crucial to define a way in which they are represented. In the last years, many text representation models have been developed in the Information Retrieval (IR) context. The basic assumption for text data is that such models have to exploit terms, since they are content descriptors used to go up to knowledge and topics. Thus, the most known text representation models are based on *index terms*, which use bag-of-words. In the following, we provide a description of the two major text models based on term indexes, that are the *Boolean* and the *Vector space* models.

Boolean model

Boolean model [Sal89, BYRN99] is one of the most simple text representation models. It exploits notions related to set theory and boolean algebra, and represents a text document as vector of term indexes. The vector is binary, since a value equals to 0 identifies the absence of that term in the document, while 1 identifies term presence. Boolean model is a rigid way of representing a document, since text characterization is only based on presence/absence of terms.

Vector space model

Another text representation model is the *Vector space model* [SL68, SWY75]. It aims to solve the principal issue related to the Boolean model, which is the inability of taking into account and handling different term relevance degrees in each document. Formally, a document $d_u \in \mathcal{D}$ can be expressed by means of an M-dimensional vector $\mathbf{d_u} = [r_{u1}, \ldots, r_{uM}]$, where r_{ul} expresses the "relevance" of the term $w_l \in \mathcal{W}$ in the document d_u.

This model provides a more accurate way to represent a document. This modeling is able to define various levels of document fitting by measuring the proximity between document vectors. A proximity measure should obey to specific requirements, in such a way that a document can be threated as a traditional database object. A proximity measure between two document vectors should be *symmetric*, *transitive*, and *reflexive*. However, it is not a reasonable choice to directly apply simple metrics (e.g., the length of the difference vector) in text data domain. For this reason, the proximity between two text documents represented according to the Vector space model can be seen as a *similarity* function. The most known similarity function in this context is the *cosine similarity* [DM01], which measures the cosine of the angle between two document vectors.

Feature Weighting and Dimensionality Reduction

It has been shown that the Vector space model represents terms in the vocabulary according to their relevance in each document. However, the concept of

term "relevance" is quite generic; to assign a proper meaning to this concept, there have been defined many techniques which analyze term occurrences and other statistics in documents and provide an estimation of term relevance. Such techniques are known as *feature weighting*.

The most general weighting technique consists in applying the *Zipf's law* [Zip49]. It is based on the assumption found by Luhn [Luh58], according to which both very frequent and very unfrequent terms are low-informative for representing the text content. Therefore, Zipf defines empirical models to estimate term frequency values for English text case words.

Starting from Zipf's law, many other term weighting schemes have been defined, which are specific for each corpora. *Term frequency* (*tf*) [Rob81] counts the number of occurrences of each terms in the vocabulary. It is based on the assumption that most frequent terms in a document represent better its content rather than unfrequent ones. However, common terms are typically highly frequent in a document; while *tf* considers them as relevant, they do not provide valuable information. This is the major drawback of this weighting scheme. To overcome such issue, *inverse document frequency* (*idf*) [Spa73] has been proposed. It is substantially based on the assumption that the higher the number of documents in which a term appears, the lower its discriminant power.

The above drawn considerations underline an important aspect: it is desirable to have a weighting strategy that takes into account both local document statistics (i.e., *tf*) and global collection statistics. This leads to the definition of a new weighting function, the *term frequency — inverse document frequency* (*tf.idf*) [Sal89] function. It is a combination of *tf* and *idf*, since it assigns high-relevance values to terms that appears frequently in a single document but rarely in the remaining documents of the corpora. Since *tf.idf* is one of the most known weighting functions, in the following we provide a formal definition:

$$tf.idf(w_l, d_u) = tf(w_l, d_u) \times idf(w_l) = freq(w_l, d_u) \times \log\left(\frac{|\mathcal{D}|}{|D(w_l)|}\right)$$

where $freq(w_l, d_u)$ denotes the number of occurrences of w_l in d_u, and $D(w_l) = \{d \in \mathcal{D} \mid freq(w_l, d_u) \geq 1\}$.

Another important aspect concerns *document normalization techniques*. Such methods aim to avoid the bias due to the presence of documents with different sizes in a collection. There have been proposed several normalization schemes based on *tf* and *tf.idf* [Sal89, RS76, SYY75].

Although *tf.idf* is able to underline features in a collection of documents, its performances in dimensionality reduction are quite low. In addition, it does not provide relevant information on statistical structure of the documents. To address these issues, IR research community has developed many other techniques. *Latent Semantic Indexing* (LSI) [DDF$^+$90] is the most representative dimensionality reduction approach. It starts from the *tf.idf* matrix representation of documents and identifies a linear subspace containing the most relevant features in the collection. Authors state that LSI performs high

data compression while preserving relevant features, and it can capture also some basic linguistic notions (e.g., synonymy and polysemy).

Generative models

In recent years there has been a growing interest in approaches to model documents based on the idea that any document can be represented as a *mixture* of probability distributions over its terms, and each component of the mixture refers to a *topic*. Within this new perspective, Hofmann made an outstanding step forward by introducing *probabilistic LSI* (pLSI) [Hof99]. After Hofmann, many other researchers investigate probabilistic model issues [Hof01, BNJ03, US02, SN07, KPAG08, ZG03, ZG05].

The assumption of considering a document as a mixture of probability distributions that underline topics is effective for a broad variety of text data, such as interdisciplinary documents. The document representation is hence obtained by a *generative process*, which expresses document features with a mixture model. Generative models are able to provide a more refined document modeling, which can also involve semantic aspects to better represent multi-topic documents. This last observation represents an important advantage in terms of expressiveness with respect to discriminative approaches, which typically focus only on syntactic aspects of documents.

Semi-structured text

The classical idea of text document refers to *plain* (or *non-structured*) text. Nevertheless, there has been a great diffusion of document collections that come from the Web. It is clear that Web documents have a higher level of structuring with respect to plain text. Therefore, it results to be interest to analyze also this other type of text data. For this purpose, the first step consists in defining a classification of text data according with their degree of structure. A plain text is usually considered as a *structure-free text*, or *non-structured text*, since analysis activities can only exploit syntactic and semantic relationships between terms in the text. By the contrast, when textual information is stored in a database system or in a file with a strict, predefined format, it is usual to refer to such data as *structured text*.

Between non-structured and structured texts there are *semi-structured* documents. They represent an evolution of structured to represent and describe complex real-life objects and their relationships. Since they do not have a rigid format, their structure may change rapidly and unpredictably. Semi-structured texts are equipped with *markup* tags that determine the logical and layout structure; this is the case of most Web pages encoded as *HTML* documents.

When text format is well-defined but flexible, markups can be used to encode *semantics*. In such cases, data can be viewed as a graph, whose vertices are objects, identified by a unique identifier and an atomic or compound value

(a set of object references). This general representation was initially proposed in the Object Exchange Model (OEM) within the Lore project at Stanford university. The goal of OEM was to build a complete management system for semistructured data [MAG+97]. Despite of the fact that OEM defines a highly-flexible modeling strategy, it does not take into account many relevant aspects, such as the *extensibility* of markup tags and structural validation with respect to a *schema* of object type definition.

XML Documents

XML (eXtensible Markup Language) [BPSM98, BPSM+04] has been defined by the World Wide Web Consortium (W3C) to overcome the limitations of OEM. It is a metalanguage for markup languages, which provides a *"standardized, extensible means of coupling semantic information within documents describing semistructured data"* [CRZ03]. All these relevant aspects have led to a great diffusion of XML, especially for representing and exchanging data on the Web.

XML allows the definition of semantic markups emphasizing both structuring and modeling aspects for the data. This is in contrast with HTML, which takes into account only the presentation and the layout issues for the data. XML documents have physical and logical structure. The first one is exploited by *entities*, while the second one consists of a hierarchy of nested *elements*. For this reason, the most natural way for representing an XML document is a labeled rooted tree, in which elements and attributes are mapped into inner nodes, and textual data information are assigned to leaf nodes.

Although XML has been defined as a flexible markup language, a set of structural constraints can be induced, which allow flexible structuring without violating *well-formedness* conditions. Indeed, an XML document can be validated with respect to its conformance to a desired structure specified by a *Document Type Definition (DTD)* [BPSM98] or an *XML Schema* [FW04, TBMM04, BM04].

Mining XML Documents

Since in XML documents implicitly lie both content and structure information, it is desirable to exploit such aspects during data analysis process. Within this view, XML data management needs proper analysis techniques, which are known as knowledge discovery from XML data. *XML mining* consists in the definition of data mining techniques that can deal with XML documents to extract relevant information from such data. Even if they are mainly text documents, it is necessary to develop ad-hoc mining strategies to take advantage of structure and content information.

There are several strategies to compare XML documents. The first one consists in exploiting only structural information within XML data to be analyzed. In this context, the most known approaches are based on *string matching* [WF74, MP80, AG97, Gus97], *tree matching* [Lu79, Tai79, HO82, Kil92],

and *schema matching* [GSSC95, KS96, CAV01, DDH01, JH01]. The second approach is more refined and takes into account both content and structure information within XML documents [GM00, DM02, YRCK01]; although, it is more difficult to define so designed algorithms.

Part II

Time Series Data Clustering

3

Time Series Data Management: State of the Art

Summary. This chapter provides a description on the state-of-the-art on time series data management. In particular, the focus is on similarity search and detection, since these aspects are the central problems in time series data processing. For this reason, it results to be important to investigate similarity measure approaches, which define a way to calculate the similarity between time series, whereas dimensionality reduction techniques have been proposed to improve the efficiency of similarity search.

3.1 Similarity Measures

DTW is widely used to perform similarity search and detection in time series. Given two sequences T_1 and T_2, DTW performs a non-linear mapping of one sequence to another by minimizing the total distance between them. To achieve such objective, a $(|T_1| \times |T_2|)$-matrix storing the squared Euclidean distances between the two sequences is used to find an optimal warping path (i.e., a sequence of matrix elements) via a dynamic programming algorithm. Moreover, a number of pruning techniques and computationally cheap lower bounds (e.g., [Keo02, KPC01, YJF98]) have been proposed to make DTW able to support fast and tight indexing and query processing.

However, a major weakness of DTW is that it tends to produce "singularities", i.e., alignments of a single point of a series with multiple points of another series. This phenomenon becomes undesirable when unexpected singularities are produced. An effective variant of DTW, called Derivative Dynamic Time Warping (DDTW) [KP01], has been proposed to reduce the phenomenon of singularities. Basically, DDTW considers new features in the sequences while maintaining a computational complexity equal to DTW. The novelty of DDTW is that local derivatives of the data points are estimated to capture information on the trends in the sequences and to find a warping more robust to singularities. For instance, two data points having identical values, one with a negative slope (i.e., part of a falling trend) and the other one with

a positive slope (i.e., part of a rising trend), are correctly not mapped each other when DDTW is used. In a sense, DDTW can be seen as DTW equipped with a preliminary preprocessing step, in which the original data points are replaced with their derivatives.

An alternative, although not computationally more convenient approach to similarity search and detection in time series is based on edit distance-like string matching measures. The Longest Common SubSequence (LCSS) algorithm [VGK02] is a variant of the edit distance that uses the length of the longest common subsequence of two sequences to define the distance between them. LCSS can deal with noisy time series by performing approximate matching rather than exact matching of time series, although it suffers from large-grained similarity. Edit Distance with Real sequences (EDR) [CÖO05] performs the same distance quantization of LCSS (which is parametric with respect to a tolerance threshold) in order to remove noisy effects. Unlike LCSS and EDR, Edit distance with Real Penalty (ERP) [CN04b] is a metric and still supports local time shifting. ERP can be seen as a variant of EDR and DTW, although it does not require a noise-tolerance threshold like EDR, and does not replicate previous data points to add a gap like DTW.

3.2 Dimensionality Reduction

To address the high dimensionality issue in time series, there are mainly two basic approaches as it has been mentioned above: approximating a time series by a piecewise discontinuous function or applying a low-order continuous function to a time series.

The first approach includes Discrete Wavelet Transform (DWT) [CF99, WAA00], Swinging Door (SD) [Bri90], Piecewise Linear Approximation (PLA) [PH74, KP98], Piecewise Aggregate Approximation (PAA) [KCPM01, KP00, YF00], Adaptive Piecewise Constant Approximation (APCA) [CKMP02], and Symbolic Aggregate approXimation (SAX) [LKLC03]. Using DWT, a time series is represented in terms of a finite length, fast decaying, oscillating and discretely sampled waveform (mother wavelet), which is scaled and translated in order to create an orthonormal wavelet basis. Each function in the wavelet basis is related to a real coefficient: the original series is reconstructed by computing the weighted sum of all the functions in the basis, using the corresponding coefficient as weight. The Haar basis [BGG97] is the most widely used in wavelet transformation. The DWT representation of a time series of length n consists in identifying n wavelet coefficients, whereas a dimensionality reduction is achieved by maintaining only the first p coefficients (with $p \ll n$).

SD is a data compression technique that belongs to the family of piecewise linear trending functions. Recently, SD has been adopted in several PI data analysis scenarios (e.g., [TSS04]). Also, in [ISS07], SD has been compared to wavelet compression. The SD algorithm employs a heuristic to decide whether

a value is to be stored within the segment being grown or it is to be the beginning of a new segment. Given a pivot point, which indicates the beginning of a segment, two lines (the "doors") are drawn from it to envelop all the points up to the next one to be considered. The envelop has the form of a triangle according to a parameter that specifies the initial amplitude of the lines. The setup of this parameter has impact on the data compression level.

In the PLA method, a time series is represented by a piecewise linear function, i.e., a set of line segments. Several approaches have been proposed to recognize PLA segments (e.g., [PH74, KP98]); among these methods, the most efficient ones are able to produce a PLA representation with computational complexity linear with the length of the time series.

PAA transforms a time series of n points in a new one composed by p segments (with $p \ll n$), each of which is of size equal to n/p and is represented by the mean value of the data points falling within the segment. Like PAA, APCA approximates a time series by a sequence of segments, each one represented by the mean value of its data points. A major difference from PAA is that APCA can identify segments of variable length. Also, the APCA algorithm is able to produce high-quality approximations of a time series by resorting to solutions adopted in the wavelet domain.

The SAX representation of a time series involves three steps. Initially, the PAA version of a time series is computed, then the PAA coefficients are quantized, and finally each quantization level is represented by a single character, called SAX symbol.

Note that representing a time series of n points according to DWT, SD, (the fastest versions of) PLA, PAA and SAX can be performed in $\mathcal{O}(n)$, whereas the complexity of APCA is $\mathcal{O}(n \log(n))$.

Dimensionality reduction techniques based on piecewise discontinuous approximations can be combined with existing similarity measures, in order to improve the computational cost in similarity searches. In particular, the use of DTW on the coefficients obtained by segmenting a time series has been investigated in the literature (e.g., [KP00]), and several lower bounding measures operating on segmented versions of a time series have been defined. Among these methods, the Fast search method for dynamic Time Warping (FTW) [SYF05] has been recently proposed as one of the most effective methods that use the time warping distance on a coarse version of the original sequences.

The other approach to dimensionality reduction, which approximates a time series with a continuous polynomial, includes Singular Value Decomposition (SVD) [KJF97, KAS98], Discrete Fourier Transforms (DFT) [RM98, RM97], splines, non-linear regression and Chebyshev polynomials [CN04a, MH03]. SVD consists of space rotation and truncation applied on a data matrix and is computationally more expensive than all the other discussed methods for dimensionality reduction. DFT and Chebyshev approaches are quite close to DWT: they are based on the use of a set of orthonormal functions, whose contributions to the whole representation are given by the relating co-

efficients. Major differences among these representations regard the functions that compose the orthonormal basis (i.e., sine waves for DFT, and Chebyshev polynomials for Chebyshev) and the computational cost (i.e., $\mathcal{O}(n \log n)$ for DFT, and $\mathcal{O}(n)$ for Chebyshev). Also, Chebyshev approximation is very close to the optimal minimax polynomial, which represents an approximation able to minimize the maximum deviation from the original data points.

It has been recently observed from an empirical viewpoint that there is no absolute winner among the dimensionality reduction methods in every application domain.[1] Nevertheless, it is interesting to compare the above methods with respect to the desiderata discussed in the Introduction. The requirement for time warping-aware representation is not satisfied by methods based on orthonormal functions such as DFT and Chebyshev, while the requirement for low complexity is satisfied by a few methods (e.g., DWT, Chebyshev, PAA, SAX and our DSA). The sensitivity to features can be considered according to three main sub-requirements (which are satisfied by our DSA) for the segments detected in an individual time series: *i)* segments may have different lengths, *ii)* any segment represents different slopes of a subsequence of data points, *iii)* segments capture the series trends. Point *i)* is addressed by APCA and SD, but not by SAX, PAA and PLA; SD and PLA satisfy point *ii)*, unlike PAA, SAX, and APCA; finally, no one of such methods satisfies point *iii)*. Also, most dimensionality reduction methods are not parameter-free, which is instead an advantageous feature of DSA—indeed, the threshold used to control the segmentation step is automatically determined in DSA, consequently the user is not required to specify it.

[1] http://www.cs.ubc.ca/∼rng/psdepository/chebyReport2.pdf

4

A Time Series Representation Model for Accurate and Fast Similarity Detection

Summary. Since the continuous increasing of sources of time series data and the cruciality of real-world applications that use such data, we believe there is a challenging demand for supporting similarity detection in time series in a both accurate and fast way. This chapter presents the *Derivative time series Segment Approximation (DSA)* representation model. This model has been conceived to support the principal issue regarding time series data management, that is accurate and fast similarity detection. The three main phases of DSA are shown, that are *derivative estimation*, *segmentation*, and *segment approximation*. The chapter concludes with an exhaustive experimental analysis, which has been conceived to compare DSA with state-of-the-art similarity methods and dimensionality reduction techniques in clustering and classification tasks. Experimental evidence from effectiveness and efficiency tests on various datasets shows that DSA is well-suited to support both accurate and fast similarity detection.

4.1 Motivation and Contributions

In many real-world applications, there is a growing interest to develop methods that are able to fit an emerging demand for both *accurate and fast similarity detection*. In this respect, we believe there is a number of special requirements that should be satisfied by any representation model to support accurate and fast similarity detection in time series, which are summarized as follows:

- *Time warping-aware.* Time series should be modeled into a form that can be naturally mapped to the time domain. This will make it feasible to benefit from using dynamic time warping for similarity detection.
- *Low complexity.* Since the high dimensionality of time series data, modeling time series should be performed maintaining a reasonably low complexity, which is possibly linear with the series length.
- *Sensitivity to features.* It is clearly desirable that time series approximation is able to preserve as much information in the original series as

possible. For this purpose, approximating a time series should be accomplished in such a way that it tailors itself to the local features of the series, in order to capture the important trends of the series.
- *Parameter-free.* Most representation models and dimensionality reduction methods require the user to specify some input parameters, such as, e.g., the number of coefficients or symbols. However, prior domain knowledge is often unavailable, and the sensitivity to input parameters can seriously affect the accuracy of the representation model or dimensionality reduction method.

For these reasons, we define a time series representation model which is conceived to support accurate and fast similarity detection. This model is called *Derivative time series Segment Approximation (DSA)*, as it achieves a concise yet feature-rich time series representation by combining the notions of *derivative estimation*, *segmentation* and *segment approximation*.

Our DSA involves a segmentation scheme that employs the paradigm based on a piecewise discontinuous function. However, in contrast to any other technique of dimensionality reduction, the segmentation step is performed on the derivative version of the original time series, rather than directly on the raw time series. The derivative estimates represent a new feature space that enables the identification of the trends of the original series. Moreover, the final segment modeling step allows for concisely fitting the detected trends in a low-dimensional, time warping-aware representation of the original time series. As we proved experimentally, the intuition underlying the DSA model works out very advantageously in supporting accurate and fast similarity detection, indeed DSA is able to fulfill all of the desiderata mentioned above:

- DSA sequences can be compared by using DTW directly;
- the derivative-based feature generation allows for representing a time series by focusing on the trends that are characteristic of the series;
- the segmentation step in DSA has a computational complexity which is linear with the series length, and it is adaptive with respect to the identified trends of the series;
- the absence of mandatory input parameters in DSA addresses the unavailability of prior domain knowledge.

We conducted an extensive experimental evaluation of DSA within clustering and classification frameworks, by considering aspects of effectiveness as well as efficiency. This evaluation necessarily involved the prominent state-of-the-art methods for time series representation and dimensionality reduction. Experimental evidence has shown that DSA supports accurate and fast similarity detection, in terms of a number of results that are summarized in Section 4.3.2.

4.2 Derivative time series Segment Approximation (DSA)

In this section we describe our *Derivative time series Segment Approximation (DSA)* model to represent time series into a concise form which is designed to capture the significant variations in the time series profile. More precisely, a DSA sequence is the result of a transformation that applies to a time series and yields a shorter sequence of values approximating the segments identified in the derivative version of the original series. DSA entails derivative estimation, segmentation and segment modeling to map a time series into a different value domain which allows for maintaining information on the significant features of the original series in a dense and concise way.

We hereinafter denote a time series with $T = [(x_1, z_1), \ldots, (x_n, z_n)]$, where each couple (x_h, z_h) is composed by a real numeric value (x_h) and a timestamp (z_h); as is often the case by assuming a fixed sampling period, T can be simply rewritten as $T = [x_1, \ldots, x_n]$. Also, we can assume that the timestamp associated with the first point x_1 is set to be zero.

Given a time series T of length n, DSA computes a new sequence τ of p values, with $p \ll n$, by three main steps:

1. Derivative estimation — the original time series is transformed into a new one in which each point is replaced with its first derivative estimate.
2. Segmentation — the derivative time series is decomposed into variable-length segments, each of which is comprised of a subsequence of points having close slopes.
3. Segment approximation — the individual segments are substantially map-ped to angular values, which represent synthetic information on the average slopes within the segments.

4.2.1 Derivative Estimation

Given a time series $T = [x_1, \ldots, x_n]$, the derivative estimation step yields a sequence $\dot{T} = [\dot{x}_1, \ldots, \dot{x}_n]$, whose elements are first derivative estimates of the points in T.

A simple yet effective derivative estimation model is that exploited in [KP01]—we hereinafter refer to it as DDTW estimation model—which computes, for each point (except the first and the last one in the series), the mean value between the slope of the line from the left neighbor to the point and the slope of the line from the left neighbor to the right neighbor. Formally:

$$\dot{x}_h = \begin{cases} \dot{x}_{h+1} & \text{if } h = 1 \\ \frac{1}{2}[(x_h - x_{h-1}) + \frac{1}{2}(x_{h+1} - x_{h-1})] & \text{if } h \in [2..n\text{-}1] \\ \dot{x}_{h-1} & \text{if } h = n \end{cases} \tag{4.1}$$

We slightly modify the DDTW estimation model by also considering the slope of the line from the point to the right neighbor; actually, this modification leads to an algebraic simplification producing an expression that is

equivalent to consider only the slope of the line from the left neighbor to the right neighbor. The derivatives of the first and the last point in the series are computed by taking into account their respective neighbors as well. Formally:

$$\dot{x}_h = \begin{cases} x_{h+1} - x_h & \text{if } h = 1 \\ \frac{1}{2}(x_{h+1} - x_{h-1}) & \text{if } h \in [2..n\text{-}1] \\ x_h - x_{h-1} & \text{if } h = n \end{cases} \tag{4.2}$$

We investigated how the performances of DSA and DDTW may vary depending on the derivative estimation model. As we describe in Appendix A, the DSA derivative estimation model (Eq. 4.2) leads to a better derivative-based feature space than the DDTW derivative estimation model (Eq. 4.1).

4.2.2 Segmentation

Segmenting a time series of length n consists in identifying $p-1$ delimiter points ($p \ll n$) to partition the series into p contiguous subsequences of points (segments) having similar features.

In our approach, segmentation is computed on the derivative version of a time series. Precisely, a derivative time series $\dot{T} = [\dot{x}_1, \ldots, \dot{x}_n]$ is transformed into a sequence $S_{\dot{T}} = [s_1, \ldots, s_p]$ of variable-length segments of the form $s_j = [s_{j,1}, \ldots, s_{j,k_j}] = [\dot{x}_{j_1}, \ldots, \dot{x}_{j_{k_j}}]$, such that:

- $s_{1,1} = \dot{x}_1$ and $s_{p,k_p} = \dot{x}_n$, and
- for each $j \in [1..p\text{-}1]$, s_{j,k_j} immediately precedes $s_{j+1,1}$ in the time axis.

A critical aspect in segmentation is how to determine the segment delimiters. For this purpose, we follow a sliding windows approach, i.e., a segment is grown until it exceeds an error threshold, and the process continues from the next point not yet considered. Although more refined segmentation schemes could be devised (e.g., top-down or bottom-up schemes), in this chapter we chose to pursue the above idea for the sake of its simplicity.

The key idea in our segmentation method is to break a series according to the first point such that the absolute difference between it and the mean of the previous points is above a certain threshold ϵ; this point becomes the anchor for the next segment to be identified in the rest of the series. Formally, let $\mu(s_j)$ denote the average of the points in a sequence s_j of $S_{\dot{T}}$, i.e., $\mu(s_j) = (\sum_{h=1}^{k_j} \dot{x}_{j_h})/k_j$, for each $j \in [1..p\text{-}1]$. The sequence s_j is identified as a segment if and only if

$$|\mu([s_{j,1}, \ldots, s_{j,h}]) - s_{j,h+1}| \leq \epsilon, \ \forall h \in [1..k_j - 1]$$

and

$$|\mu([s_{j,1}, \ldots, s_{j,k_j}]) - s_{j+1,1}| > \epsilon$$

Intuitively, this condition allows for aggregating subsequent data points having very close derivatives; in such a way, the growth segment s_j represents a subsequence of points with a specific trend.

To estimate the threshold ϵ, we resort to an index of spread of the (derivative) data points within a sequence. The objective is to produce a number of segments that is large enough to capture the "characteristic" trends in the original series (i.e., subsequences of points having close derivative estimates), but small enough to guarantee a reasonably good degree of compression. Precisely, we devise three definitions of ϵ, namely *dataset-oriented*, *series-oriented*, or *segment-oriented*.

The dataset-oriented definition of ϵ aims to express this threshold by globally referring to a given collection of time series. Given a dataset $\mathcal{D}$ of N time series, ϵ can be defined as:

$$\epsilon(\mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} \frac{|\dot{T}_i|}{\max\{|\dot{T}| \mid T \in \mathcal{D}\}} \sigma(\dot{T}_i)$$

where $\sigma(\dot{T}_i)$ denotes the standard deviation over the points in the i-th derivative series, and normalization of the series lengths is provided to deal with variable-length series.

The above definition is reasonably adequate when most time series in the collection show similar shapes; however, this may not necessarily hold in several real domains (e.g., sensor network measurements). Therefore, an estimation of ϵ might be provided by a series-oriented definition, i.e., globally to each individual time series T:

$$\epsilon(\dot{T}) = \sigma(\dot{T})$$

Another definition of ϵ may involve the individual segments being identified in each series. We can hence define a segment-oriented ϵ for each segment s_j as

$$\epsilon(s_j) = \sigma(s_j)$$

It is easy to observe that, regardless the definition of ϵ, the segmentation step on a dataset of N series can be performed in $\mathcal{O}(N \times n_{max})$, where n_{max} is the maximum of the series lengths. Since the segment-oriented definition is tailored to the local features of an individual series, we chose to adopt it in the segmentation step of our DSA model.

It is worth noting that the segmentation step in DSA does not require any user-specified parameter, since the threshold ϵ is automatically chosen by analyzing the information of each series. By contrast, this step is not automatic for other methods of dimensionality reduction (i.e., APCA, SAX, SD, PAA, PLA, Chebyshev, DWT and DFT), where the user is required to specify an input, such as the number of segments or coefficients being computed. We believe this represents an important advantage of our method.

4.2.3 Segment Approximation

The individual segments of a derivative time series are represented with a synthetic information capturing their respective main features. More precisely,

each segment s_j is mapped to a pair formed by the value z_j+1, where z_j is the timestamp of the last point ($\dot{x}_{j_{k_j}}$) in s_j, and an angle that explains the average slope of the portion of time series bounded by s_j. This is mathematically expressed by the notion of arctangent applied to the mean of the (derivative) points in each segment.

Given a segmented derivative time series $S_{\dot{T}} = [s_1, \ldots, s_p]$, the final step of segment approximation yields a sequence $\tau = [(\alpha_1, t_1), \ldots, (\alpha_p, t_p)]$ such that

$$\begin{aligned} \alpha_j &= \arctan(\mu(s_j)), \; j \in [1..p] \\ t_j &= t_{j-1} + k_j, \qquad j \in [1..p] \end{aligned}$$

where we assume $t_0 = 0$ for any DSA sequence.

Modeling a given time series by means of the DSA representation hence leads to a new sequence whose elements (pairs angle-timestamp) still maintain a direct association to the original time domain, while concisely representing the features of original points. This makes the DSA model able to fully support dynamic time warping, i.e., (dis)similarities between DSA sequences can be computed by using DTW-based measures.

As a final remark, it is easy to observe that the time complexity of computing a DSA sequence from a time series of length n has a total cost $\mathcal{O}(n)$, since the three steps, namely derivation, segmentation and segment approximation, cost $\mathcal{O}(n)$, $\mathcal{O}(n)$, and $\mathcal{O}(p)$ ($p \ll n$), respectively.

4.3 Experimental Evaluation

We devised an experimental evaluation to assess the ability of our DSA in supporting effective and efficient similarity detection within clustering and classification frameworks. We compared DSA against state-of-the-art methods for modeling and comparing time series data, which include LCSS, EDR, ERP, DTW, DDTW and FTW as distance measures, and APCA, SAX, PAA, PLA, SD, Chebyshev, DWT and DFT as dimensionality reduction methods. Since our DSA and the competing dimensionality reduction methods are not similarity/distance measures, we chose to apply DTW over the segments/coefficients computed by each particular representation scheme in the time domain (i.e., APCA, SAX, PAA, PLA, SD and DSA), whereas we used the Euclidean distance (L_2) to compare the sequences obtained by Chebyshev, DWT and DFT, as suggested in their respective works.

Before going into the details of the obtained results, in this section we first describe experimental settings by introducing the clustering and classification algorithms and the validity criteria used. We also discuss the preliminary task of preprocessing of the raw time series and the setup of the various methods that compete with our DSA.

4.3.1 Settings

Algorithms

Finding the best strategy of time series clustering or classification is not an objective of this research; rather, we are interested in assessing the impact of the proposed time series representation model in similarity detection, and hence we conceived standard clustering and classification frameworks for time series data. Specifically, we resorted to well-known paradigms, namely partitional clustering, agglomerative hierarchical clustering [JD88] and nearest neighbor classification [Mit97]. As stated in [War05], partitional and hierarchical clustering methods have been extensively used in the context of time series clustering. Analogously, using nearest neighbor learning has been widely encouraged for time series classification (e.g., [KXWR]).

Partitional clustering. The partitional clustering paradigm is characterized by simplicity, and low computational and memory requirements. In this chapter, we mainly use the popular K-means algorithm [McQ67], which is a centroid-based partitional clustering method; in Appendix C we also discuss a more general partitional clustering method. It is worth noting that choosing the number of output clusters does not represent a drawback in our evaluation context, since we selected datasets for which reference classifications are available, and hence we were able to fix the number of clusters equal to the actual number of classes in each clustering experiment. Also, we address the random selection of the initial cluster centroids by performing multiple runs of the K-means algorithm to avoid that the quality results were due to random chance. In order to define the cluster centroids, we adopt two strategies depending on whether or not the representation model produces variable-length segments. Concerning SAX, PAA, PLA, SD, Chebyshev, DWT and DFT, we compute the cluster representatives by simply averaging the corresponding coefficients over the time series in any specific cluster. In the following, we present a method for computing cluster representatives of DSA sequences; we remark that although this method has been originally conceived for DSA clusters, it can be easily adapted to any representation model that is able to produce variable-length segments (coefficients), such as APCA.

Computing cluster representatives in K-means. Let us denote with $\mathcal{T}_C = \{\tau_1, \ldots, \tau_M\}$ a cluster of DSA sequences, where each τ_i has the form $[(\alpha_{i_1}, t_{i_1}), \ldots, (\alpha_{i_{p_i}}, t_{i_{p_i}})]$, and with $C = \{T_1, \ldots, T_M\}$ the cluster of original time series such that each $T_i \in C$ is associated with a unique $\tau_i \in \mathcal{T}_C$. The objective is to compute a DSA sequence prototype $rep(\mathcal{T}_C)$ as the representative of cluster C.

We identify a fixed number v of segments over which the average series is defined. We can reasonably define the number of segments v as the closest integer to the mean $(\sum_{i=1}^{M} p_i)/M$ over all the series $\tau_i \in \mathcal{T}_C$. The timestamps associated to the new v segments are defined as $\hat{t}_j = t_{max} \times j/v$, for each $j \in [1..v]$, where $t_{max} = \max\{t_{u_{p_u}} \mid \tau_u \in \mathcal{T}_C\}$ $(i \in [1..M])$ and $\hat{t}_0 = 0$. For

each τ_i, the angle α'_{i_j} corresponding to the timestamp $\hat{t}_j$ (with $\hat{t}_j \leq t_{i_{p_i}}$) is computed to be equal to the angle α_{i_u} of the u-th segment including the point sampled at time $\hat{t}_j$. Formally, α'_{i_j} is equal to the angle α_{i_u} such that $(\alpha_{i_u}, t_{i_u}) \in \tau_i$ and $t_{i_{u-1}} < \hat{t}_j \leq t_{i_u}$, for all $i \in [1..M], j \in [1..v]$. Note that any pair $(\alpha'_{i_j}, \hat{t}_j)$ is introduced only if the condition $\hat{t}_j \leq t_{i_{p_i}}$ holds, i.e., if the i-th time series is defined in the timestamp $\hat{t}_j$.

For each τ_i the new v_i pairs $(\alpha'_{i_j}, \hat{t}_j)$ are then included in the rewritten DSA sequence $\tau''_i = [(\alpha''_{i_1}, t''_{i_1}), \ldots, (\alpha''_{i_{q_i}}, t''_{i_{q_i}})]$ which is computed as

$$\textit{time-sort}\{(\alpha_{i_1}, t_{i_1}), \ldots, (\alpha_{i_{p_i}}, t_{i_{p_i}}), (\alpha'_{i_1}, \hat{t}_1), \ldots, (\alpha'_{i_{v_i}}, \hat{t}_{v_i})\}$$

where $\hat{t}_{v_i}$ is the new timestamp with maximum value defined over the i-th sequence.

Formally, for each $\tau_i \in \mathcal{T}_C$, the sequence $\hat{\tau}_i = [(\hat{\alpha}_{i_1}, \hat{t}_1), \ldots, (\hat{\alpha}_{i_v}, \hat{t}_{v_i})]$ is computed, where

$$\hat{\alpha}_{i_j} = \frac{\sum_{(\alpha''_{i_u}, t''_{i_u}) \in \tau''_i \ \wedge \ t''_{i_u} \in (\hat{t}_{j-1}, \hat{t}_j]} [\ \alpha''_{i_u} \times (t''_{i_u} - t''_{i_{u-1}})\ ]}{\hat{t}_j - \hat{t}_{j-1}}, \quad j \in [1..v_i]$$

The DSA representative $rep(\mathcal{T}_C)$ is finally computed as:

$$rep(\mathcal{T}_C) = [(\overline{\alpha}_1, \hat{t}_1), \ldots, (\overline{\alpha}_v, \hat{t}_v)], \quad \text{where} \quad \overline{\alpha}_j = \frac{\sum_{\hat{t}_j \leq \hat{t}_{v_i} \wedge i \in [1..M]} \hat{\alpha}_{i_j}}{|\{\hat{t}_j | \hat{t}_j \leq \hat{t}_{v_i}\}|}$$

for each $j \in [1..v]$. Note that $\Delta t = \hat{t}_j - \hat{t}_{j-1}$ is a constant for each $j \in [1..v]$.

Hierarchical clustering. The agglomerative hierarchical clustering paradigm allows us to test the competing methods in a clustering framework which does not rely on a notion of cluster prototype and on the cluster initialization. For this purpose, we use the UPGMA algorithm (Unweighted Pair Group Method using arithmetic Averages), which is based on group-average-linkage to compute the distance between any two clusters [JD88].

K-Nearest Neighbor classification. As the most basic instance-based learning method, the *K-Nearest-Neighbor (K-NN)* algorithm [Mit97] is a straightforward approach to assess the various representation and similarity detection methods in our context. Indeed, according to the K-NN algorithm, the classification of an instance (time series) will be most similar to the classification of instances that are similar to each other.

Assessment criteria

To assess the effectiveness of classification algorithms, we directly compare the result of the automatic assignment with a reference (expected) organization of the data. Analogously, in clustering frameworks, we assess how well a clustering solution fits a given scheme of known classes, thanks to the availability of reference classifications for all the test datasets. For this reason, we resort to F_1-*measure* and *Error Rate* ($\mathcal{E}$) (cf. Section 2.1.6).

Preprocessing time series

Raw time series are usually preprocessed by *smoothing* data points in order to reduce the noise in the data. Moving average represents the simplest family of smoothing models, as it is a compromise between the mean and the random walk model. Given a raw time series $T = [x_1, \ldots, x_n]$ and a smoothing degree δ (i.e., the maximum width of the moving average), the *centered δ-point moving average* recomputes the data points by considering both the previous and next observations around a center:

$$x_h^{smoothed} = \begin{cases} \mu([x_1, \ldots, x_{h+r}]) & \text{if } h-r \leq 0 \\ \mu([x_{h-r}, \ldots, x_{h+r}]) & \text{if } h-r > 0 \text{ and } h+r \leq n \\ \mu([x_{h-r}, \ldots, x_n]) & \text{if } h+r > n \end{cases}$$

, where $r = (\delta - 1)/2$ denotes the maximum number of back and forward points that are taken into account for smoothing the h-th point.

More refined models, such as exponential smoothing models, compute the weighted average of past observations on the basis of previously smoothed observations. Given a smoothing factor $\omega \in [0..1]$, the simple *exponential smoothing* is computed as:

$$x_h^{smoothed} = \begin{cases} x_h & \text{if } h = 1 \\ \omega x_h + (1-\omega) x_{h-1}^{smoothed} & \text{if } h > 1 \end{cases}$$

It should be emphasized that denoising is essential to make the data amenable to the further analysis tasks, regardless of the specific representation method or distance measure used. In particular, in derivative-based feature spaces, denoising time series data (e.g., via a smoothing function) before differentiating them is necessary to avoid that the approximation of derivatives by finite differences will amplify the noise present in the data. In this respect, the combination of smoothing prior to the step of derivative estimation in our DSA approach (as well as in DDTW) can be seen as somehow similar to the regularization of a differentiation process [TA77], although potentially less accurate and general.

Setup of the competing methods

Unlike our DSA, most of the competing methods require one or more parameters to be set. In some cases, which include LCSS, EDR, ERP and Chebyshev, typical settings are suggested in their respective works; specifically, the matching thresholds for LCSS and EDR are assumed to be equal to $(\max \sigma(T_i))/4$ and $\min \sigma(T_i)$ respectively (for all the series T_i in a dataset), the constant gap for ERP is set to 0, and the number of coefficients for Chebyshev is set to 20. Such parameter settings revealed to be good enough to enable the respective methods to achieve their best performances in accuracy. In particular, we took care in monitoring the behavior of the Chebyshev method, and finally

found no significant improvement in accuracy by increasing the number of Chebyshev coefficients.

In other cases, to make a comparative evaluation possible in terms of accuracy and efficiency, we aimed to prepare the various methods to perform at levels of data compression which were as close as possible. We tried several values for the parameters of APCA, SAX, PAA, PLA, DWT, DFT and FTW. More precisely, for each dataset and algorithm, we varied the setting of each of these methods in such a way that it achieved the same compression (i.e., number of segments) obtained by DSA, and ±5%, ±10%, and ±20% of the DSA compression; then, we measured the relative clustering/classification quality (F_1-measure) scores and finally chosen the setting corresponding to the best score. Analogously, the alphabet length W (i.e., the number of symbols) required by SAX was chosen, for each dataset and clustering/classification algorithm, as the value that led to the best trade-off between clustering/classification quality and time performance.

A final remark concerns SD, which requires a deviation threshold (i.e., the "doors" amplitude); however, setting this parameter is even more difficult, since the compression factor (i.e., the number of segments) cannot be specified directly in swinging door compression. In [TSS04, XCCH02], many calibration trials are conducted to find the deviation thresholds corresponding to a given compression factor, for each dataset. We followed this approach and set the deviation threshold in such a way that the number of segments produced by SD was as near as possible to the number of segments produced by DSA, finally choosing the value that led to best clustering/classification quality.

Data description

We selected seven datasets coming from various application domains, and characterized by different series profiles and dimensionality. Table 4.1 summarizes the characteristics of the main datasets used for the experiments.

Table 4.1. Datasets used in the experiments

dataset	*size*	*classes*	*time steps*
GunX	200	2	150
Tracedata	200	4	275
ControlChart	600	6	60
CBF	300	3	128
Twopat	800	4	128
Mixed-BagSh.	160	9	1,614
OvarianCancer	49	2	28,000

GunX comes from the video surveillance domain, whereas Tracedata simulates signals representing instrumentation failures. In CBF (Cylinder-Bell-Funnel), each class is characterized by a specific pattern, namely a plateau

(C), an increasing ramp followed by a sharp decrease (B), a sharp increase followed by a decreasing ramp (F). ControlChart contains synthetically generated control charts which are classified into one of the following: normal, cyclic, increasing trend, decreasing trend, upward shift, and downward shift. In Twopat, two different patterns (upward step and downward step) are used to define the classes down-down, up-down, down-up, and up-up. Mixed-BagShapes contains time series derived from shapes belonging to nine classes (bone, cup, device, fork, glass, hand, pencil, rabbit and tool). The first five datasets are available at http://www.cis.temple.edu/∼latecki/TestData/TS_Koegh/, whereas Mixed-BagShapes can be found at http://www.cs.ucr. edu/∼eamonn/shape/.

Besides the benchmark datasets above, we also used OvarianCancer [PAH+02], which contains proteomic spectra generated by Surface-Enhanced Laser Desorption and Ionization - Time Of Flight Mass Spectrometry (SELDI-TOF MS). The spectra (i.e., MS data) are derived from an analysis of serum samples of a female population belonging to two classes (ovarian cancer diseased and healthy). It should be emphasized that OvarianCancer data, like most of MS datasets, are huge-dimensional and largely affected by noisy factors. Noise is typically due to a number of reasons, such as sample preparation, insertion of the samples into the mass spectrometer, and instrumental and measurement errors. For this purpose, OvarianCancer spectra were subject to a preliminary preprocessing phase specific for MS data [MCK+05, WNP03]. MS preprocessing has been recognized as a crucial step for tasks of MS data management and knowledge discovery, and mainly consists of operations such as noise reduction, baseline subtraction and peak detection. The interested reader can find details about the preprocessing steps carried out in [GPT+08b].

Table 4.2. Segmentation and compression rate performed by DSA modeling

dataset	*avg. no. DSA segments*	*compr.*
GunX	68	55%
Tracedata	118	57%
ControlChart	35	42%
CBF	77	40%
Twopat	38	70%
Mixed-BagSh.	816	49%
OvarianCancer	943	97%

It is interesting to have a look at the impact on the time series dimensionality by using DSA. Table 4.2 shows that DSA achieves a 59% compression of the original series lengths on average, with a maximum compression percentage of 97% in the OvarianCancer dataset. As we shall discuss later in this section, the reasonably good rate of compression achieved by DSA does not have a negative impact on the accuracy in detecting similarities.

4.3.2 Results

Effectiveness evaluation

We measured the ability of DSA and the competing methods in supporting time series clustering and classification effectively. We investigated how clustering/classification results can be influenced by choosing different alternatives for data preprocessing and setting the parameters therein involved; then, we assessed the performance of DSA and the other methods according to their respective best settings.

Tuning preprocessing parameters. We used the smoothing functions previously described in Section 4.3.1 to preprocess the time series in each dataset. In the case of centered moving average, the smoothing degree $\delta(= 2r + 1)$ was varied within a typical range, namely [5..9], whereas ω in exponential smoothing settings was varied from 0 to 1 by a 0.1 step. Moreover, we tried to perform zero, one or more iterations of smoothing (up to 5), on the various datasets and for each preprocessing scheme; the rationale here is that smoothing should be avoided to prevent loss of information for low-noise series and, conversely, excessive noise might be treated with multiple smoothing. It should be noted that, in the case of K-means evaluation, we performed multiple runs (100) of the K-means algorithm and finally averaged the quality results over the runs to obtain a single value of F_1-measure.

We observed that smoothing helped to improve the performance of the various methods on all the datasets, except OvarianCancer; OvarianCancer represented an exception since, in this case, data was preliminarily subject to domain-specific preprocessing steps (cf. Section 5.1.1), hence further preprocessing via smoothing would have tended to cause loss of information on potentially significant data features.

The exponential smoothing revealed to be more effective than moving average, as it was selected 74 out of 90 times as the best preprocessing way. The parameter ω was set to low values in most cases, thus suggesting the need for a greater smoothing effect (which is indeed achieved by values of ω closer to zero). Also, the number of smoothing iterations appeared to be not relevant in practice; three iterations of smoothing were enough in most cases, except for SD which always required four or five iterations. In Appendix B, we report further details about the preprocessing stage, including the best settings and an evaluation of the impact of smoothing on the various datasets.

Accuracy in time series clustering. We evaluated DSA and the other methods in two clustering frameworks, namely K-means and UPGMA. In relation to the selected clustering algorithm, each method was used with the best preprocessing setup for the specific dataset. In any case, the quality of the obtained clustering solutions was calculated in terms of F_1-measure.

Table 4.3 refers to K-means clustering and shows the quality results averaged over 100 runs of this algorithm. Looking at the table we can observe that DTW on DSA sequences (for short, DTW on DSA) was the first ranked

Table 4.3. Summary of average quality results (F_1-measure) for K-means clustering

	GunX	Trace data	Control Chart	CBF	Twopat	Mixed BagSh.	Ovarian Cancer
LCSS	0.59	0.30	0.50	0.79	0.36	0.32	0.34
EDR	0.54	0.74	0.88	0.86	0.42	0.70	0.58
ERP	0.72	0.62	0.76	0.58	0.39	0.48	0.34
DTW	0.66	0.78	0.87	0.89	*0.95*	*0.77*	0.60
DDTW	*0.89*	*1*	*0.89*	*0.97*	*0.95*	0.76	*0.62*
FTW	0.74	0.90	0.81	0.67	0.55	0.73	0.58
L_2 on DFT	0.63	0.77	0.78	0.67	0.39	0.70	0.36
L_2 on DWT	0.61	0.67	0.76	0.74	0.36	0.68	0.36
L_2 on CHEBY	0.57	0.72	0.70	0.69	0.38	0.72	0.34
DTW on SD	0.67	0.95	0.85	0.87	0.89	0.76	0.69
DTW on PLA	0.73	0.77	*0.89*	0.87	0.75	0.74	0.60
DTW on PAA	0.68	0.78	0.87	0.86	0.73	0.75	0.59
DTW on SAX	0.73	0.77	0.83	0.87	0.69	0.71	0.58
DTW on APCA	0.77	0.81	*0.89*	0.83	0.91	0.74	0.55
DTW on DSA	**0.92**	**1**	**0.90**	**0.96**	**0.97**	**0.78**	**0.75**

method in all the datasets except CBF; however, in this dataset, DTW on DSA was only 1% below the performance of DDTW, which turned out to be the best method among the competing ones. Also, DTW on DSA always led to better results than DTW alone. It should be emphasized that the comparison with DDTW is particularly important in order to gain an insight into the role of derivative-based features in time series representation and the impact of combining derivative estimation and segmentation in the accuracy of similarity detection.

DTW on DSA performed as good as or better than the remaining methods, and in some cases the performance difference was quite evident. In particular, DTW on DSA led to quality improvements up to about 59% with respect to DWT, DFT and Chebyshev, and up to 25% with respect to SAX, PAA, PLA, SD, and APCA. Among these competing methods, it can be noted that DTW on APCA and SD obtained the best results; in general, DTW on APCA, SD, SAX, PAA and PLA achieved higher quality clustering than Chebyshev, DWT and DFT.

Table 4.4. Summary of quality results (F_1-measure) for UPGMA clustering

	GunX	Trace data	Control Chart	CBF	Twopat	Mixed BagSh.	Ovarian Cancer
LCSS	0.47	0.23	0.38	0.39	0.24	0.24	0.33
EDR	0.49	0.38	0.41	0.47	0.28	0.28	0.56
ERP	0.49	0.37	0.40	0.40	0.33	0.28	0.54
DTW	0.61	0.48	0.48	0.51	0.61	0.38	0.61
DDTW	*0.72*	*0.76*	*0.54*	0.49	*0.64*	0.41	*0.63*
FTW	0.60	0.52	0.44	0.44	0.50	0.40	*0.63*
L_2 on DFT	0.60	0.40	0.29	0.47	0.39	0.28	0.62
L_2 on DWT	0.60	0.40	0.29	0.47	0.39	0.28	0.62
L_2 on CHEBY	0.60	0.40	0.29	0.47	0.39	0.28	0.62
DTW on SD	0.51	0.55	0.40	0.49	0.43	*0.42*	0.60
DTW on PLA	0.61	0.63	0.40	0.51	0.43	0.29	0.61
DTW on PAA	0.61	0.61	0.36	0.51	0.56	0.41	0.61
DTW on SAX	0.61	0.60	0.48	*0.56*	0.44	0.31	0.61
DTW on APCA	0.61	0.63	0.40	0.51	0.43	0.29	0.61
DTW on DSA	**0.73**	**0.82**	**0.54**	**0.60**	**0.67**	**0.51**	**0.73**

Table 4.4 reports on the quality results obtained by the UPGMA algorithm. A first remark on these results is that the F-measure scores for the various methods were generally much lower than the corresponding results obtained by the K-means algorithm on all the datasets, except OvarianCancer; in particular, for DTW on DSA, there was a quality decrease from 19% (in GunX) to 36% (in CBF and ControlChart). However, like in K-means clustering, DTW on DSA revealed to behave as good as or better than the other methods and, in some cases (i.e., CBF and Mixed-BagShapes) we observed increased advantages (quality improvements) with respect to the corresponding results by K-means.

Accuracy in time series classification. Analogously to the evaluation by clustering frameworks, we assessed the performance of DSA and the competing methods using the K-NN classification algorithm. Table 4.5 shows the best results obtained by the various methods, where each triple of values refers to the parameter K, F_1-measure and accuracy, in that order. We split each dataset 70% for training, the remainder for testing. Concerning the choice of K (the

Table 4.5. Summary of quality results (F_1-measure, accuracy ($\mathcal{A}$), and the corresponding K in parentheses) for K-NN classification

	GunX	Trace data	Control Chart	CBF	Twopat	Mixed-BagSh.	Ovarian Cancer
LCSS	F_1=.63 (3) $\mathcal{A}$=.63 (3)	F_1=.42 (3) $\mathcal{A}$=.75 (1)	F_1=.65 (1) $\mathcal{A}$=.85 (1)	F_1=.36 (5) $\mathcal{A}$=.37 (1)	F_1=.51 (1) $\mathcal{A}$=.68 (1)	F_1=.80 (1) $\mathcal{A}$=.92 (1)	F_1=.29 (1) $\mathcal{A}$=.63 (3)
EDR	F_1=.83 (5) $\mathcal{A}$=.80 (5)	F_1=.95 (1) $\mathcal{A}$=.96 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.76 (3) $\mathcal{A}$=.94 (2)	F_1=.62 (1) $\mathcal{A}$=.80 (1)
ERP	F_1=.99 (1) $\mathcal{A}$=.99 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.79 (5) $\mathcal{A}$=.81 (5)	F_1=.64 (1) $\mathcal{A}$=.75 (1)	F_1=.78 (1) $\mathcal{A}$=.96 (1)	F_1=.58 (1) $\mathcal{A}$=.63 (1)
DTW	F_1=.95 (3) $\mathcal{A}$=.95 (3)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.84 (1) $\mathcal{A}$=.97 (1)	F_1=.68 (5) $\mathcal{A}$=.80 (2)
DDTW	F_1=1 (2) $\mathcal{A}$=1 (2)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.92 (1) $\mathcal{A}$=.98 (1)	F_1=.70 (1) $\mathcal{A}$=.87 (2)
FTW	F_1=.89 (4) $\mathcal{A}$=.88 (4)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.93 (1) $\mathcal{A}$=.93 (1)	F_1=.88 (1) $\mathcal{A}$=.90 (1)	F_1=.83 (2) $\mathcal{A}$=.96 (2)	F_1=.68 (3) $\mathcal{A}$=.80 (1)
L_2 on DFT	F_1=.96 (1) $\mathcal{A}$=.96 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.98 (1) $\mathcal{A}$=.98 (1)	F_1=.77 (3) $\mathcal{A}$=.77 (3)	F_1=.91 (3) $\mathcal{A}$=.98 (3)	F_1=.65 (1) $\mathcal{A}$=.70 (1)
L_2 on DWT	F_1=.90 (5) $\mathcal{A}$=.89 (5)	F_1=.95 (2) $\mathcal{A}$=.96 (2)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.90 (1) $\mathcal{A}$=.93 (1)	F_1=.88 (1) $\mathcal{A}$=.97 (1)	F_1=.65 (1) $\mathcal{A}$=.80 (2)
L_2 on CHEBY	F_1=.71 (1) $\mathcal{A}$=.71 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.91 (1) $\mathcal{A}$=.93 (1)	F_1=.85 (1) $\mathcal{A}$=.97 (1)	F_1=.61 (1) $\mathcal{A}$=.77 (2)
DTW on SD	F_1=.99 (5) $\mathcal{A}$=.99 (5)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.87 (1) $\mathcal{A}$=.97 (1)	F_1=.68 (4) $\mathcal{A}$=.73 (1)
DTW on PLA	F_1=.95 (3) $\mathcal{A}$=.95 (3)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.86 (1) $\mathcal{A}$=.97 (1)	F_1=.68 (4) $\mathcal{A}$=.77 (1)
DTW on PAA	F_1=.99 (1) $\mathcal{A}$=.99 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.87 (1) $\mathcal{A}$=.97 (1)	F_1=.67 (2) $\mathcal{A}$=.73 (2)
DTW on SAX	F_1=.78 (1) $\mathcal{A}$=.74 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.98 (5) $\mathcal{A}$=.99 (5)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.99 (1) $\mathcal{A}$=.99 (1)	F_1=.80 (2) $\mathcal{A}$=.95 (1)	F_1=.63 (3) $\mathcal{A}$=.80 (1)
DTW on APCA	F_1=.95 (3) $\mathcal{A}$=.95 (3)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=1 (1) $\mathcal{A}$=1 (1)	F_1=.86 (1) $\mathcal{A}$=.97 (1)	F_1=.68 (4) $\mathcal{A}$=.77 (1)
DTW on DSA	**F_1=1 (1) $\mathcal{A}$=1 (1)**	**F_1=1 (1) $\mathcal{A}$=1 (1)**	**F_1=1 (1) $\mathcal{A}$=1 (1)**	**F_1=1 (1) $\mathcal{A}$=1 (1)**	**F_1=1 (1) $\mathcal{A}$=1 (1)**	**F_1=.95 (1) $\mathcal{A}$=.98 (1)**	**F=.75 (1) $\mathcal{A}$=.90 (1)**

neighborhood size), it should be emphasized that [KXWR] recommends the use of the "simple yet very competitive" 1-NN algorithm (i.e., K-NN with K equal to 1); we followed the lead of this authoritative reference, however we also considered more values of K, up to $K = 5$.

Using DTW on DSA led to both F_1-measure and accuracy equal to 1 for all the selected K on GunX, Tracedata, CBF and Twopat. Moreover, DTW on DSA performed better than the other methods on Mixed-BagShapes and GunX (in this dataset, DDTW achieved optimal accuracy like DSA, although requiring a neighborhood size greater than 1). It should also be noted that DDTW confirmed to be the best method among the competing ones.

We observed that both F_1-measure and accuracy tended to decrease for higher values of K. In particular, the use of $K = 1$ revealed to be the best choice, in terms of F_1-measure or accuracy, 87 out of 105 times over all the methods and datasets—which advocates the aforementioned recommendation by [KXWR].

Efficiency evaluation

We measured the time performances of DSA and the other methods in accomplishing the tasks of modeling and clustering time series. For each dataset, we randomly selected samples of sizes equal to 25%, 50%, 75% and 100% of the size of the entire dataset and, for each of the such samples, we performed a preprocessing stage for the various methods, as we did for the entire datasets (see Section 4.3.2). At the end, for each sample and method, we used the respective best preprocessing setup. We left the string matching based approaches (i.e., LCSS, EDR and ERP) out of this presentation since they revealed to be drastically slower than all the other methods.[1]

Table 4.6. Summary of best time performances (msecs) in time series modeling

	GunX	Trace data	Control Chart	CBF	Twopat	Mixed BagSh.	Ovarian Cancer
DFT	2,835	8,276	1,652	3,679	4,701	286,829	570,056
DWT	8	13	9	5	17	48	181
CHEBY	58	58	173	87	232	46	*14*
SD	7	15	14	17	24	65	262
PLA	4	7	*2*	3	*4*	21	46
PAA	*2*	*3*	*2*	*2*	*4*	*14*	41
SAX	8	13	11	11	19	56	67
APCA	1,758	7,151	412	657	2,358	68,739	135,982
DSA	**15**	**40**	**27**	**31**	**52**	**143**	**391**

Performances in time series modeling. Table 4.6 summarizes the best performances (in milliseconds) in modeling time series on the various datasets. PAA, PLA, SAX, SD and DWT performed as the fastest methods; actually, this result was not surprising since simpler models obviously lead to higher efficiency and, at the same time, lower accuracy. DFT and APCA were always by far slower than the other methods. Our DSA was close to the fastest methods in most cases; in particular, compared to Chebyshev, the larger the dataset the faster was modeling by DSA with respect to modeling by Chebyshev polynomials. It is important to note that, since DSA shares with PAA, PLA, SD and SAX the same asymptotic time complexity (i.e., linear with

[1] Experiments were conducted on a platform Intel Pentium IV 3GHz with 2GB memory and running Microsoft WinXP Pro.

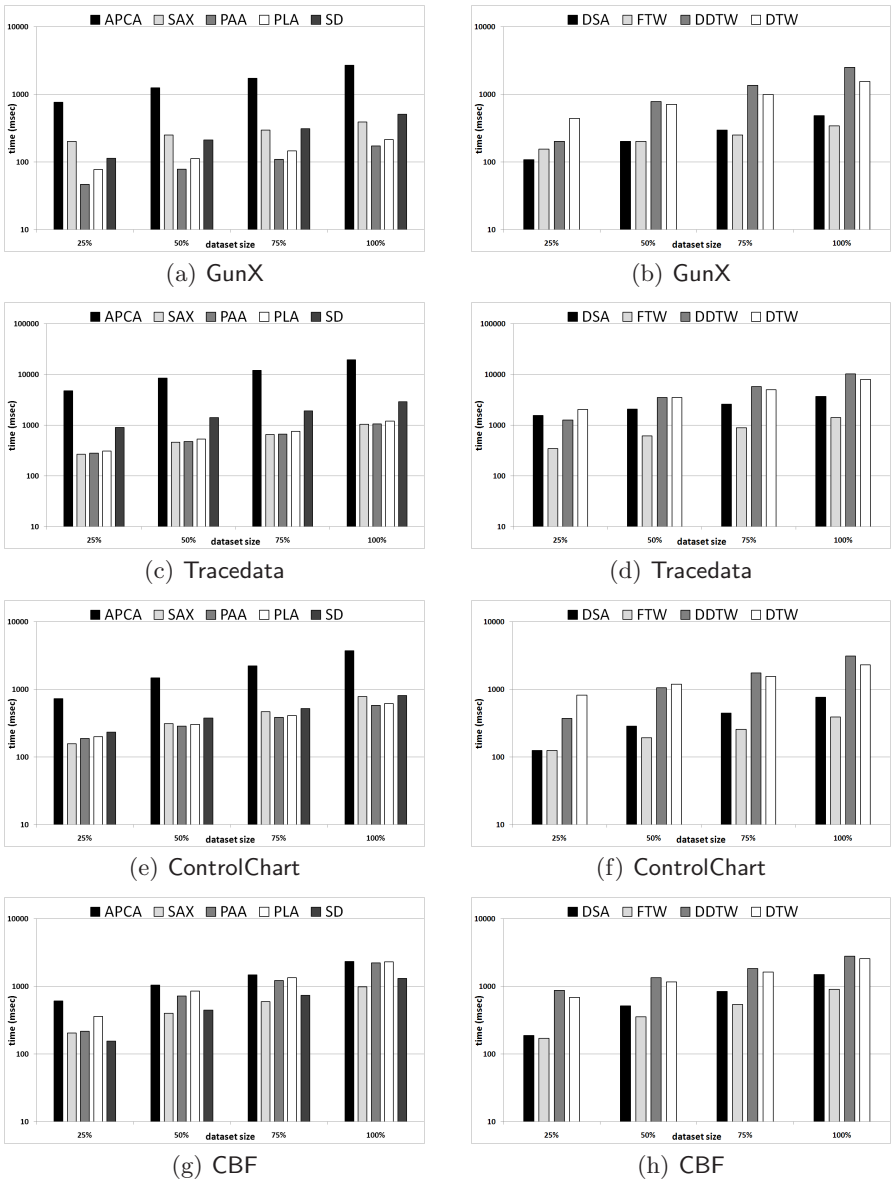
Table 4.7. Summary of best time performances (msecs) in time series modeling and clustering

	GunX	Trace data	Control Chart	CBF	Twopat	Mixed BagSh.	Ovarian Cancer
DTW	1,548	8,018	2,298	2,564	3,548	594,446	4,071,815
DDTW	2,518	10,345	3,117	2,789	4,971	664,885	4,098,329
FTW	368	1,491	*405*	*926*	5,254	237,823	5,252,045
DTW on SD	511	2,912	814	1,317	870	240,744	22,655
DTW on PLA	253	1,286	652	2,342	1,103	218,272	20,588
DTW on PAA	*197*	*1,103*	601	2,262	*812*	*216,196*	18,685
DTW on SAX	413	1,113	805	1,132	1,941	282,312	*18,403*
DTW on APCA	2,865	19,503	3,793	2,563	4,864	749,003	9,282,532
DTW on DSA	**521**	**3,733**	**792**	**1,587**	**377**	**121,402**	**23,821**

the series length), the time differences between DSA and these fast methods should be not really relevant in practical contexts.

Performances in time series clustering. We also evaluated the time performances for the clustering task, including in this stage the time required by the series modeling task as well; for the sake of brevity, we present here results obtained by the K-means algorithm, and we focus on time warping-aware representations.

Figures 4.1–4.2 and the summary reported in Table 4.7 show that DTW on DSA drastically improved the clustering performances of basic DTW and DDTW; clearly, this was a consequence of the dimensionality reduction due to the segmentation performed by DSA. More surprisingly, DSA behaved very close to the fastest competing methods: indeed, it is interesting to note that the performance difference between DSA and PAA, SAX, PLA and SD was not as evident as in the modeling performances previously observed; in particular, DSA-based clustering was even faster than PAA, SAX, PLA and SD on Twopat (the largest dataset) and Mixed-BagShapes (the dataset with the highest number of classes). This suggests that our DSA is able to yield a time series representation that might require more time to be computed, but generally is more accurate yet convenient to fit the whole task of clustering.



(a) GunX (b) GunX

(c) Tracedata (d) Tracedata

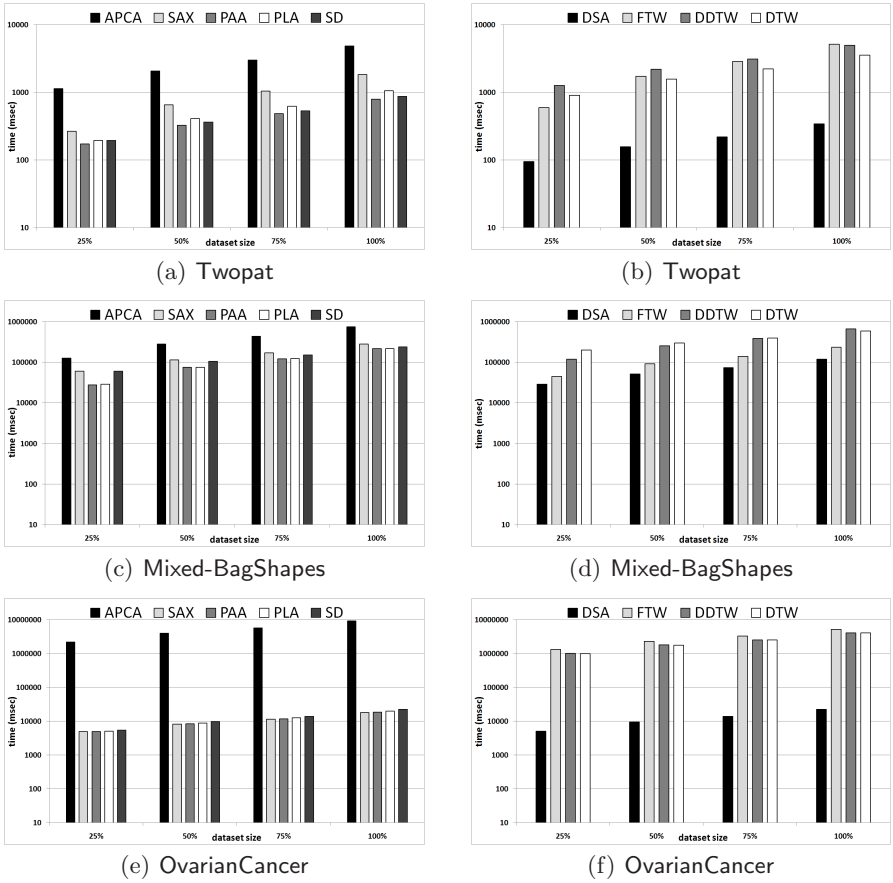
(e) ControlChart (f) ControlChart

(g) CBF (h) CBF

Fig. 4.1. Time performances in time series modeling and clustering: GunX, Tracedata, ControlChart, CBF

Summary of results and discussion

We evaluated the capabilities of our DSA as well as the competing methods in supporting similarity detection within clustering and classification frame-



(a) Twopat (b) Twopat

(c) Mixed-BagShapes (d) Mixed-BagShapes

(e) OvarianCancer (f) OvarianCancer

Fig. 4.2. Time performances in time series modeling and clustering: Twopat, Mixed-BagShapes, OvarianCancer

works. Of course, the extent to which such frameworks actually led to good solutions depend on how each of the following critical aspects was devised: (i) the preprocessing scheme, (ii) the similarity method and its application in relation to a representation model, and (iii) the clustering/classification algorithms.

Since the focus of this research is on a compact representation of derivative-based features of time series and its impact on similarity detection, it should be emphasized that the evaluation frameworks are indeed "parametric" with respect to the algorithms. In order to provide a complete specification of our evaluation frameworks, we conducted experiments using standard clustering and classification algorithms mainly because of their simplicity, applicability, and relatively less dependence on algorithmic parameters.

Facing with the experimental results presented in the above sections and having the focus on point *ii)*, we can summarize the main remarks of our study as follows:

- Applying the dynamic time warping (DTW) to DSA sequences leads to clustering and classification solutions that are more accurate than those obtained by using DTW on the original time series. The advantage taken by DSA is essentially due to the combination of a derivative-based feature generation with dimensionality reduction by segmentation. In this way, DTW on DSA performs as good as or better than the major methods for time series representation and dimensionality reduction (i.e., SAX, APCA, PLA, SD, PAA, Chebyshev polynomials, etc.) and even than DDTW alone.
- Modeling a time series into a DSA sequence is reasonably fast if compared to other methods of dimensionality reduction, which is supported by a computational complexity that is linear with the series length; most importantly, the trade-off between accuracy and compactness of DSA sequences makes performing similarity detection more advantageous.

5

DSA in Real Case Applications

Summary. This chapter shows how to involve DSA representation model in real case applications, in such a way that it is possible to exploit its facilities to manage and analyze data of other contexts. In particular, the focus is on *Mass Spectrometry data* and *load profile electricity data*. For both scenarios, it will be discussed how to model data according to a time series based representation, i.e., DSA. Experimental evaluation assessed the validity of DSA in highlighting main data features in both domains, and how such modeling leads to achieve high accuracy results in clustering task.

5.1 Analyzing Mass Spectrometry Data

Mass Spectrometry (MS) is a powerful analytical technique aimed to extract interesting biological information from tissue or serum samples [GV03, AM03]. Due to its ability in separating ions of different masses from a sample, a mass spectrometer generates a vector of measurements representing the number of ions that hit the spectrometer detector during small, fixed intervals of time. A mass spectrum is thus represented as a plot of ion abundance (*intensity*) versus the mass-to-charge ratio (m/z). Techniques for spectra generation usually depend on sample preparation and ionization, used instrument and ions/macromolecules detections. Mostly used methods are Matrix-Assisted Laser Desorption/Ionization - Time Of Flight Mass Spectrometry (MALDI-TOF MS) [KH88] and Surface-Enhanced Laser Desorption/Ionization - Time Of Flight Mass Spectrometry (SELDI-TOF MS) [HY93]. By analyzing mass spectra it is possible to identify macromolecules contained in the original compounds by associating (portions of) proteins to their peak expressions in a spectrum.

The large amount of spectra generated by any MS process makes it mandatory automatic analysis and knowledge discovery from such data. Indeed, the size of raw spectra usually span from a few kilobytes to a few gigabytes per spectrum. Recently, there has been a lot of research concerning advanced

analysis on MS data in order to extract significant, previously unknown information or "knowledge" from such data. This usually involves various tasks aimed to identify biological patterns and organize them at different degrees of automation. Consequently, several approaches for MS data management and mining have been recently developed. For instance, in [PAH+02], data mining techniques have been used to identify discriminants in a female population, distinguishing ovarian cancer diseased from healthy ones. Similarly, data mining techniques for spectra data have been applied in [SGF+07] for surface-enhanced laser desorption/ionization mass spectrometry (SELDI MS) data, to identify discriminants in rectal cancer disease. In [PKS+04] machine learning algorithms have been used to identify biomarkers in SELDI MS data generated on tens of patients to figure out cerebral accident discriminants.

MS data preprocessing has been recognized as a mandatory phase in mass spectra data analysis. The need for preprocessing mass spectra arises since the data obtained from a mass spectrometer (i) have very large dimensionality and (ii) are naturally corrupted by various noisy factors. Several research studies have been proposed on the development of preprocessing steps for MS data (e.g., [CBM07, WNP03]), and in some cases they have focused on specific steps, such as baseline subtraction [WCD+05, SS04, RJFD99], peak identification [WKG04, YMA+03], and peak alignment [WCC05, Jef05, SS04]. Also, there has been recently a growing interest for developing MS data preprocessing systems that are able to fulfill the following main requirements: filtering data and highlighting relevant spectra portions with respect to non-relevant ones (e.g., noise), and allowing the user to perform the various preprocessing stages iteratively and interactively.

5.1.1 The Mass Spectrometry Data Analysis (MaSDA) System

In the next sections, we present a system for advanced analysis of MS data. The general objective of this system, called *Mass Spectrometry Data Analysis (MaSDA)* [GPT+09c], is to assist the user in discovering useful knowledge from MS data. The discovered patterns of knowledge might eventually support the user (i.e., the clinician) to take critical decisions. For instance, if some interesting relationships on certain biological conditions referring to a given disease have been found out by analyzing MS data, then one might use this new information to better plan early detecting of cancers.

The key idea underlying our approach to MS data analysis, which is implemented in the MaSDA system, is to exploit the temporal information implicitly contained in mass spectra and model such data as compact *time series* (cf. Chapter 4). The proposed MS data representation model is aimed to take some advantages with respect to the traditional count-vector-based approaches to MS data representation, in particular:

- The problem of high dimensionality of MS data is addressed by identifying variable-length segments in the time series representing mass spectra. Each

one of these segments is conceived to be comprised of locally tight points, and is finally mapped to a synthetic information. This enables us to drastically reduce the number of noisy dimensions while preserving relevant features (i.e., trends in the series profile).
- The critical task of preprocessing MS data is relatively simplified by leveraging major existing techniques for similarity detection in time series, which allow for dealing with mass spectra in a way more robust to noise and suited to different profiles of the spectra.

Another important aspect of our MS data analysis system is that it offers a graphical tool for preprocessing the raw mass spectra [GPT+08b], with the following main features:

- *Wide set of supported preprocessing operations* — it is designed to cover most of the MS data preprocessing steps that have been recognized as relevant in the literature.
- *Efficiency* — it guarantees high performance in MS preprocessing, by adopting fast algorithms for each step. This allows for efficiently dealing with high dimensional data.
- *Support for user interaction* — it enables the user to monitor and control the whole preprocessing task; in particular, the user can choose which preprocessing steps have to be performed and their execution order, and she/he can properly set the parameters involved into each step.
- *Ease-to-use* — it provides a user-friendly graphical interface and a simple wizard which guides the user in each preprocessing step.
- *Web-based access* — it makes use of the Java Web Start technology (JWS), which allows for launching the tool directly from the Web.[1]

Besides the functionalities of MS preprocessing and time series based MS modeling, our MaSDA system is designed to perform various tasks of MS data analysis, by employing *Data Mining and Knowledge Discovery* techniques, and to evaluate and visualize the patterns of knowledge discovered from the input MS data. As experimentally proved on some publicly available datasets, our system has been shown as a valid support for the user interested in effectively and efficiently analyzing MS data.

Our MaSDA system consists of five main modules, which are sketched in Figure 5.1.

1. *MS Data Preprocessing*: it performs one or more preprocessing steps on the raw spectra in order to make them amenable to the further analysis stage. In particular, this module includes at least the following preprocessing operations: range cut, peak smoothing, detection of valid peaks, baseline correction, quantization, and normalization.

[1] A beta version of the MS data preprocessing tool is available at the following Web address: http://polifemo.deis.unical.it/∼gtradigo/jnlp/msptool/

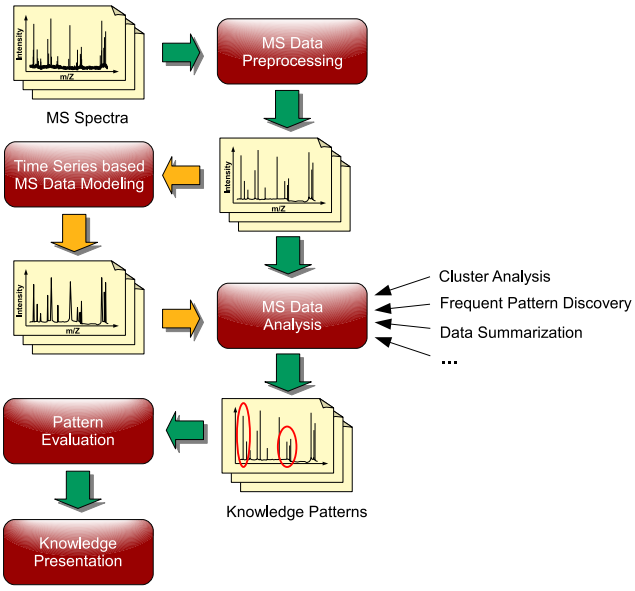


Fig. 5.1. The overall conceptual architecture of the MaSDA system

2. *Time Series based MS Data Modeling*: this module transforms the preprocessed MS data into time series, using a model conceived to maintain the significant trends (peak profiles) while reducing the data dimensions.
3. *MS Data Analysis*: it includes a number of submodules each performing a certain task of knowledge discovery, such as cluster analysis, frequent pattern discovery, data summarization, and so on. The input for this module is the preprocessed spectra, which is of the form either original (output of module 1) or based on time series (output of module 2).
4. *Pattern Evaluation*: it is in charge of assessing the validity of the knowledge patterns discovered.
5. *Knowledge Presentation*: this module finally presents the knowledge discovered by using visualization tools.

Preprocessing Mass Spectrometry Data

A raw spectrum outputted from a mass spectrometer is substantially a combination of three components: the true signal, a baseline signal, and noise [CBM07]; in particular, the true signal contains biological information, whereas the base intensity level (baseline) varies from point to point across the m/z axis, so that intensity values that are under the baseline represent ground noise and should be hence filtered out. Separating and reconstructing

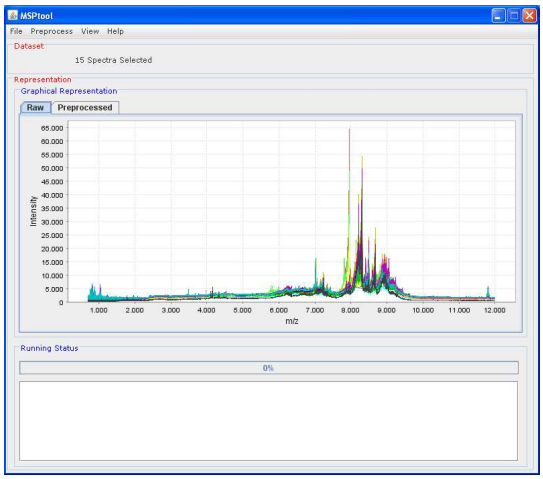


Fig. 5.2. A sample screenshot of the MSPtool, the MaSDA module for MS preprocessing

such individual components from a raw spectrum is a hard task, since their analytical forms are not known. Thus, spectra usually need to be subject to one or more preprocessing operations, in order to make them amenable to further analysis phases.

Since the variety of spectrometry platforms, experimental conditions and clinical studies, there exists a number of preprocessing operations (see, e.g., [CBM07, WNP03]). While there has not been shared agreement about a general-purpose preprocessing scheme, a reasonable list of preprocessing steps on mass spectra can be given as follows:

- *calibration*, which is used to map the observed time of flight into the inferred mass-to-charge ratio;
- *filtering or denoising*, which aims to reduce random noise generated by electronic or chemical causes;
- *baseline correction*, which is in order to recognize and filter out the baseline signal of mass spectra;
- *normalization*, which makes peak intensities understandable over a uniform range;
- *peak detection*, which is in charge of locating specific proteins or peptides on the identified locations on the m/z axis and typically involves an assessment of the spectra local maxima and their signal-to-noise ratio (S/N);
- *peak quantification*, which represents each detected peak by means of a concise information (e.g., peaks heights or areas);
- *peak matching/alignment*, which aims to recognize which peaks in different samples correspond to the same biological molecule.

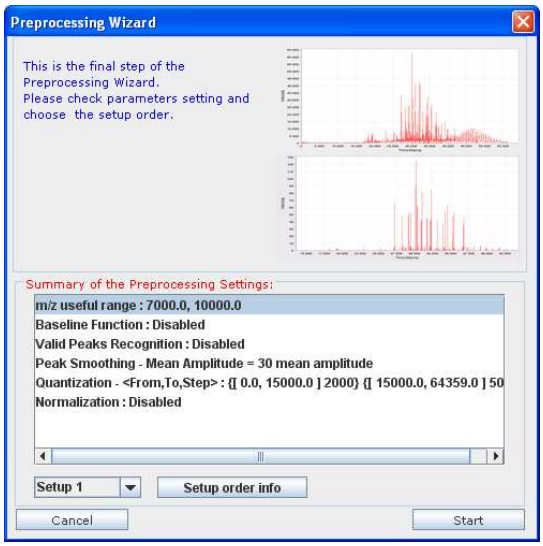


Fig. 5.3. Preprocessing Wizard - Summary of the preprocessing settings

In the following, we describe the capabilities of the module in MaSDA for MS preprocessing, we called *MSPtool*. MSPtool is a Java™ developed tool that implements most of the MS preprocessing operations discussed above (Figure 5.2). This tool offers its features visually in order to assist the user in performing an MS preprocessing task, i.e., observing the raw spectra, selecting an appropriate sequence of preprocessing steps, and choosing the parameter setting for each of the selected preprocessing steps.

MSPtool is able to deal with various formats storing the raw spectrum/spectra to be preprocessed, including plain-text files, comma separated values files (CSV), XML data. Also, the tool allows the user to graphically represent preprocessed spectra, which is useful to visually explore (and compare) the spectra profiles before and after the preprocessing step. Figure 5.3 shows a screenshot of the last step of the preprocessing wizard, which reports a summary of the preprocessing setting; in this step, it is also possible to change the order of the selected preprocessing operations.

We provide an overall description of the main MS data preprocessing steps involved into MSPtool.

Range cut

This step provides a cut of the m/z range of the spectra, in order to filter out those portions of the spectra that do not contain relevant biological information.

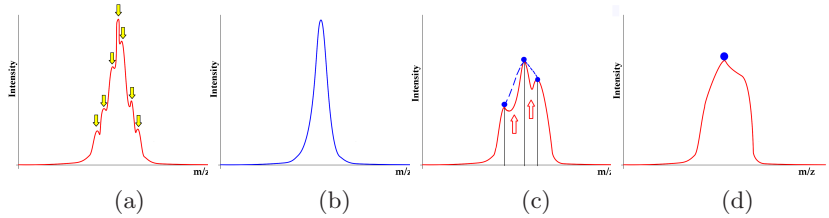


Fig. 5.4. Peak smoothing: (a) example M-peaks and (b) the corresponding ideal peak; (c) three local M-peaks and (d) the resulting profile after smoothing

Peak smoothing

Peak smoothing falls into the category of peak detection/quantification steps. This step deals with smoothing peak profiles in the spectra and is accomplished to reconstruct the theoretical Gaussian profile of the peaks.

An ideal peak profile is comprised of two parts: a monotonic ascending side and a monotonic descending side. We call *M-peak* a peak in a spectrum having its intensity higher than both the previous and the next point, i.e., a local maximum in the spectrum (Figure 5.4 (a)–(b)).

The peak smoothing algorithm has a parameter w_p (*peak amplitude*). w_p is a function of the mass spectrometer resolution, and can be initially set to the average width of peaks in the spectrum. Nevertheless, the user can change the value of w_p according to the data features. Basically, the algorithm works as follows: first, it detects all the M-peaks in the spectrum; each M-peak (except the last one) is compared with the next M-peak. If the distance between these two M-peaks is lower than $w_p/2$ then either a descending phase or an ascending phase can occur, and the spectrum is modified such that the resulting peak has the expected pseudo-Gaussian shape for both the ascending and the descending sides (Figure 5.4 (c)–(d)).

Valid peaks recognition

Valid peaks recognition is a further step of peak detection/quantification. This step aims to recognize as valid peaks the local maxima into a mass spectrum that satisfy specific requirements. In particular, the algorithm for valid peaks recognition implemented into MSPtool takes into account the signal-to-noise ratio (S/N) and works as follows: for each spectrum, the S/N at each local maximum of the spectrum is firstly computed by taking the ratio of the intensity at the maximum to the local noise estimate; then, only the local maxima having S/N greater than a user-defined threshold (*multiplicative factor*) are recognized as valid peaks. The non-valid peaks in a spectrum are discarded from the further analysis.

Baseline correction

This step aims to identify the baseline signal in the spectra and filter out all spectra intensity values below the baseline. In this module, the user can choose the function suitable for approximating the baseline (i.e., the baseline function) and setting the parameters for each function. In particular, in MSPtool the baseline functions available are four: *linear function*, *logarithmic function*, *exponential function* and *piecewise linear function*. The first three functions approximate the baseline as a linear, logarithmic and exponential function, respectively, whereas the definition of the piecewise linear function is as follows. The m/z range of each spectrum is divided into a user-defined number of equally-sized windows. The final piecewise linear function is composed by a number of linear functions, each of them properly defined according to the associated window. In particular, for each window, the corresponding linear function is computed by solving a line fitting problem to the local minima in the window.

Quantization

This step performs a quantization of the spectra, that is a discretization of the original intensity values according to specific quantization levels. A non-uniform quantization model is used in the MPStool in such a way that two or more ranges in the intensity axis are identified and subject to different fine-grained quantization.

Normalization

Spectra normalization changes spectra shapes by transforming original intensity values into new ones proportionally calculated according to a certain fixed range.

MSPtool implements several normalization techniques, including z-normalization and min-max normalization. The former subtracts the mean over all the spectra intensities from each intensity value and then divides this difference by the standard deviation over all the spectra intensities; the latter scales the intensity values such that, for each m/z and over all the spectra, the smallest intensity value becomes zero and the largest intensity becomes one.

Time series based modeling of MS data

A (preprocessed) mass spectrum is a sequence of paired values $S = [((m/z)_1, I_1), \ldots, ((m/z)_n, I_n)]$, where each pair is comprised of a mass-to-charge-ratio value and the associated intensity value. A mass spectrum so defined can be trivially modeled as a time series $T = [(x_1, t_1), \ldots, (x_n, t_n)]$ whose x_i correspond to the spectrum intensity values I_i, and the time steps t_i correspond to the

values $(m/z)_i$. Indeed, the notion of time implicitly lies in the sequence of mass-to-charge values.

Time series representing mass spectra are typically high dimensional data. Thus, it is desirable to model such time series into a compact representation that synthesizes the significant variations in the time series profile. For this purpose, we exploit a DSA-based representation scheme, which has been described in Section 4.2.

5.1.2 Experimental Evaluation

In this section we describe the experimental evaluation aimed to assess effectiveness of the proposed framework. We first present the data used in the main experiments and our evaluation methodology. Then we discuss the experimental results from both a quantitative and qualitative point of view.

Settings

Data description

Typical datasets in real-world MS application domains contain just tens or hundreds of spectra [CBM07]. We used datasets available from the NCI's Center for Cancer Research.[2] All these datasets contain SELDI-TOF spectra and were obtained using different clinical studies under different mass spectrometry platforms and experimental conditions. Below is a brief description of the datasets, whereas Table 5.1 summarizes their main characteristics.

Table 5.1. Main characteristics of test datasets

	#instances	*#attributes*	*#classes*	*size (MB)*
Cardiotoxicity	115	7,105	4	12.7
Pancreatic	181	6,771	2	18.2
Prostate	322	15,154	4	101

- Cardiotoxicity — high resolution, binned spectra used in a toxiproteomic analysis of anthracycline-induced cardiotoxicity [PRH$^+$04]. Data is labeled according to four classes: definite negative (24), probable negative (43), probable positive (10), definite positive (34).
- Pancreatic — high resolution, binned spectra used in a study on premalignant pancreatic cancer detection [HPM$^+$03]. There are here two classes: control (101), pancreatic intraepithelial neoplasias (80).

[2] http://home.ccr.cancer.gov/ncifdaproteomics/ppatterns.asp

- Prostate — low resolution spectra used in a study on prostate cancer, which have been already provided with the baseline subtracted [POP$^+$02]. Data is assigned with four classes: cancer and PSA level > 10 ng/ml (43), cancer and PSA level within [4..10] ng/ml (26), benign and PSA level > 4 ng/ml (190), no evidence of disease and PSA level < 1 ng/ml (63).

Evaluation Methodology

The objective of our evaluation methodology is to automatically measure the quality of a clustering solution. This requires to define the following elements for each experimental test: the data, the way(s) to preprocess the data, the clustering algorithm along with the distance/similarity measure and the choice of number of clusters, and the quality measure(s).

We resorted to K-means as the clustering algorithm equipped with either the Euclidean distance (L_2) or the Dynamic Time Warping (DTW); the latter has been used on the spectra modeled as time series by means of our DSA, whereas the Euclidean distance is referred to as the baseline method for computing distance among MS data.

As far as the data preprocessing, we did not fix a unique way to preprocess data but rather we identified the following sequences of operations as different *preprocessing setups*: *(S1)* baseline subtraction, peak detection, normalization; *(S2)* peak detection, baseline subtraction, normalization; *(S3)* baseline subtraction, normalization; *(S4)* peak detection, normalization; *(S5)* peak detection, baseline subtraction; *(S6)* baseline subtraction, peak detection; *(S7)* baseline subtraction; *(S8)* peak detection; *(S9)* normalization.

Clustering evaluation

To evaluate clustering quality, we exploit the availability of a reference classification for the data and use measures that allow for computing how well a clustering solution fits a predefined scheme of known classes (natural clusters). In oder to evaluate clustering results, we resort to two well-known external validity measure, that are F_1-measure and Entropy (E) (cf. Section 2.1.6).

Results

A major task of MS knowledge discovery consists in classifying spectra in order to discriminate them on the basis of their biological states (e.g., healthy or diseased individuals). To cope with huge dimensionality and frequently occurring noise in MS data, this task requires careful preprocessing and modeling of the data. However, the organization task is intrinsically difficult when no a-priori knowledge on the predefined set of categories or a training set of positive/negative examples from data is poorly or not available at all. In this case, the goal is to infer an organization of a collection of MS data into meaningful groups (i.e., *clusters*), based on interesting relationships discovered in the data. *Clustering* of MS data finds natural application to many real MS

scenarios, since the various pathologic states from clinical studies might require to be discovered in an unsupervised way. In the following we present the MaSDA functionality for organizing MS data which is accomplished by means of a task of cluster analysis [GPT+09c, GPTG09].

Figure 5.5 shows a screenshot of the MaSDA tool for clustering MS data. On the left of this figure, we can observe a number of component panels devoted to the configuration of a clustering experiment, which involves the choice of the preprocessing (smoothing) function, the model of cluster representative, the method of representation and similarity between series, and the algorithm of clustering. On the right of the figure, each of the output clusters and relating representative can be explored using different choices of visualization; the clustering results can be also saved into a file for further reloading.

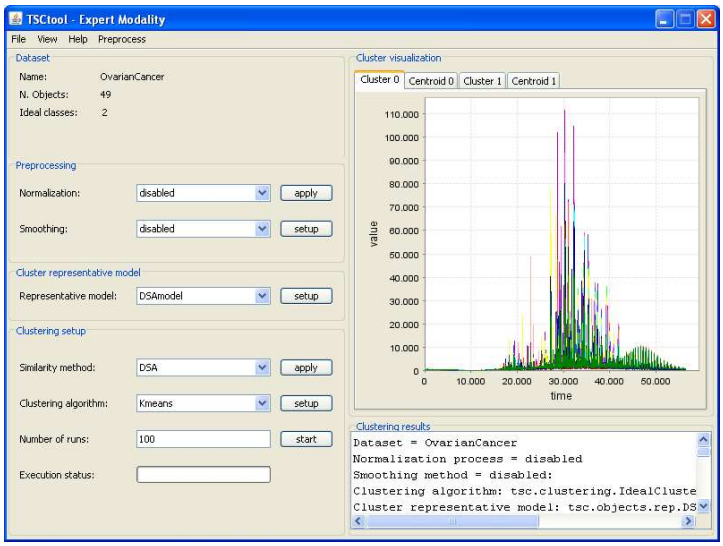
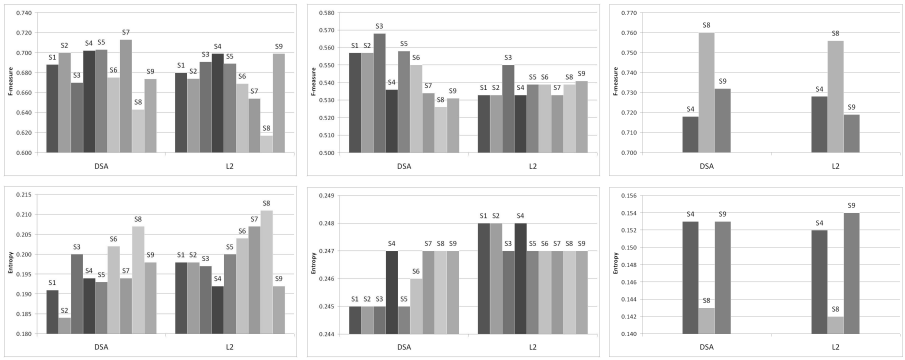


Fig. 5.5. A screenshot of the MaSDA tool for clustering MS data

Another important task allowed by MaSDA is that of data summarization. Given any set of MS time series, the objective here is to generate a summary, or prototype, sequence that is able to capture the most relevant features of the series in the given set.

A major goal of the experimentation conducted on MS data by using MaSDA was to identify groups of subjects that show similar characteristics according to the expected pathological states (e.g., in the Prostate dataset [POP+02] different cancer or benign conditions at various levels of PSA). Moreover, in this context a challenge is represented by the discovery of the proteomic profiles that distinguish disease-related or cancer conditions



(a) (b) (c)

Fig. 5.6. Clustering quality results (F_1-measure on top, Entropy on bottom): (a) Cardiotoxicity, (b) Pancreatic, and (c) Prostate

from the healthy ones. For instance, some discriminatory patters might be found out around early m/z values, other ones might be detected according to sequences of peaks at a certain intensity level. Intuitively, this issue can be more easily addressed by exploiting our time-series-based modeling of MS data: indeed, the compact representation that is substantially comprised of relevant features in the data (while discarding various noisy factors) favors the identification of significant patterns in the spectra.

For each test dataset, we conducted various runs of the K-means algorithm, by varying the preprocessing setup, and compared the results obtained by our time series based approach with the standard Euclidean approach.

Quantitative evaluation

We initially focused on testing the ability of the framework to detect and distinguish all the meaningful groups in the data, that is 4 for Cardiotoxicity and Prostate and 2 for Pancreatic. Thus, for all experiments on a specific dataset, we fixed the number of clusters exactly to the number of classes associated with that dataset.

Figure 5.6 shows the quality results obtained on the various datasets, and compares our DSA-based approach to the standard Euclidean distance.[3] Clustering performances were generally affected by the selected preprocessing setup, while baseline subtraction revealed to be essential for improving the clustering quality in most cases. Notice that only setups $S4$, $S8$ and $S9$ were considered for Pancreatic, since this data has been already subject to baseline subtraction.

Our DSA-based approach achieved reasonably good clustering results at least in Cardiotoxicity and Prostate; in Pancreatic, clustering inevitably resulted

[3] Reported results were averaged over 100 runs of the K-means algorithm with a very low standard deviation (ranging between 0.001 and 0.008).

in lower performances mainly due to the dominant presence of m/z values with very low intensity values and just a few characteristic trends in the spectra. Anyway, for all datasets the DSA-based approach was able to achieve higher F_1-measure and lower Entropy scores than the standard Euclidean approach.

Table 5.2. Evaluating MaSDA system: summary of clustering results on the various test datasets

		DSA			L_2		
	# clust. (K)	F_1 (*prec./rec.*)	E	*preproc. setting*	F_1 (*prec./rec.*)	E	*preproc. setting*
Cardiotoxicity	4	.72 (.75 / .68)	.19	$S7$	.69 (.70 / .69)	.19	$S4/9$
Cardiotoxicity	2	.76 (.78 / .73)	.38	$S2$	.67 (.67 / .67)	.45	$S4/9$
Pancreatic	2	.57 (.58 / .56)	.48	$S3$	.55 (.57 / .53)	.49	$S3$
Prostate	4	.76 (.84 / .69)	.14	$S8$	.75 (.84 / .68)	.14	$S8$
Prostate	2	.78 (.79 / .78)	.34	$S8$	.77 (.79 / .76)	.34	$S8$

Table 5.2 summarizes the quality results (i.e. F_1-measure along with corresponding precision and recall, and Entropy (E)) referring to the best preprocessing setups; for each dataset and method, the best preprocessing setup is that leading to the highest quality in terms of F_1-measure. This table also includes results obtained by a two-class task of clustering; precisely, for Cardiotoxicity and Prostate, we also tried to select only the data assigned with the definite cancer (diseased) or the definite non-cancer (healthy) classes, and then we performed clustering on this subsets. The objective here was to give emphasis on distinguishing solely the extreme classes.

Performing K-means with dynamic time warping on DSA sequences behaved as good as or better than standard Euclidean distance on the original spectra, up to a 10% improvement (Cardiotoxicity, 2 classes). It should be noted that the advantage of using the dynamic time warping on DSA sequences becomes important since the DSA model yields compact yet dense representations of the original spectra. The lower dimensionality of the spectra-series achieved by DSA is beneficial for the efficiency of the clustering task (and further post-processing analysis), while not affecting negatively the clustering effectiveness. To report some details, the following data compression ratios were achieved (on the preprocessed spectra): 64% on Cardiotoxicity, 65% on Pancreatic, and 97% on Prostate. The latter result is particularly significant since it shows that a very high compression was obtained for a dataset on which DSA still performs very closely to (slightly better than) the standard approach based on L_2.

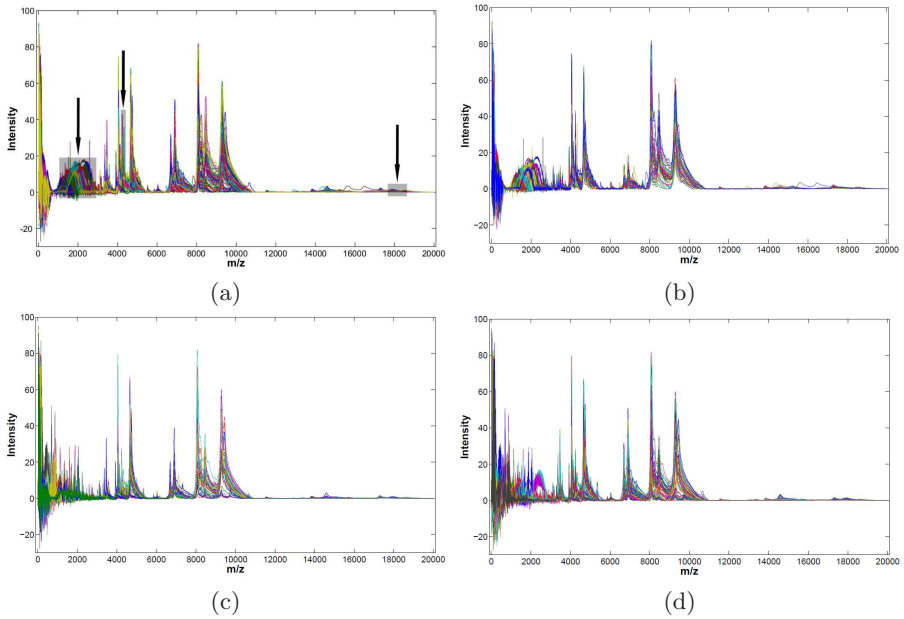


Fig. 5.7. Clusters vs. natural classes from Prostate: (a) cluster and (b) class of cancer with PSA>10 ng/ml; (c) cluster and (d) class of no evidence of disease

Qualitative evaluation

It is useful to gain an insight into the output clusters with respect to the real classes. Here we concentrate on Prostate and compare the definite cancer and no-evidence-of-disease clusters to their respective groupings in the reference classification (Figure 5.7)—for the sake of comparison, original spectra have been displayed for both clusters and classes. In Prostate, as discovered by authors of the study in [POP+02], there is a number of main discriminatory patterns, mostly distributed in the early m/z values. Figures 5.7(a) and (b) plot spectra belonging to the cluster and the class of definite cancer, respectively. At a first glance, the two plots look quite similar, suggesting that most cancer spectra have been correctly recognized. Also, we have highlighted on Figures 5.7(a) some of the most evident trends that distinguish the cancer conditions from the healthy ones. Analogously, we can observe similar graphs for the healthy cluster/class (Figures 5.7(c)/(d)).

5.2 Profiling Electricity Company Customers

Today energy markets are characterized by a growing insecurity in the wake of their liberalization. Due to an increasing customer volatility, it is becoming more difficult for utilities to plan their investments through the next decades. In addition, the problem of characterizing and predicting their customers' behavior and fitting a proper tariff policy accordingly has been recognized as relevant in this context. Designing new tariff structures allows the energy utilities to encourage competition, efficiency, and economical use of the resources. Defining proper tariffs makes it possible to support customers' interests and, at the same time, to recover the cost of electricity in a reasonable time. Enel, a large international power utility, has recently completed an Italian project called the Telegestore project [BCR05, Rog07]. By using up-to-date smart meters, Enel is able to measure and store load profiles of their mass-market LV customers in a flexible and effective way.

In recent years, technological improvement in electricity utility devices has leveraged various issues in load profile data management. In this respect, a significant research effort has been focused on load profile classification, especially regarding clustering of medium-voltage customers and short-term load forecasting of anomalous days. Customer classification puts the basis for properly designing tariff structures. The use of load pattern-based features has been identified as a key factor for classifying customers on the basis of their electrical consumption behavior. Classification allows utilities to promote collective tariffs rather than individual ones for each customer.

All the proposed techniques for load profile classification generally belong to pattern recognition and data mining approaches [FRVG05, CNP+05, CNP06, NDJR06, ALB+07, THD07, TSTK08]. Load profiles are usually represented as time sequences and the notions of proximity used for comparing them are typically based on the Euclidean distance. In the context of load profile clustering, the most used approaches refer to partitional clustering and hierarchical clustering [JD88].

In the following sections, we present a clustering framework for electricity customer load profiles, which is supported by information on meta-data (e.g., customer type, meter type, day, contract, location). Enel supported this research by providing data about 30,000 LV load profiles of anonymous Italian customers. The key intuition here consists in thinking at a load profile as a *time series*, and exploiting the advantages provided by our DSA model (cf. Section 4.2) in preserving main trends while achieving high compression level.

A major emphasis of our study was on the most typical class of electricity customers, i.e., private, residential domestic customers. Each customer load profile was segmented with respect to the type of day, which enabled a characterization of the customers' profiles on a per day basis.

We performed experiments by varying the algorithm and the distance measure in the proposed clustering framework. More precisely, we used the standard K-means and the Euclidean distance as baseline method. However, we

also resort to the Dynamic Time Warping distance, which is widely known to provide a better way to compare time series. Moreover, we introduce a simple top-down partitional algorithm, named TS-Part, which does not require the user to specify a desired number of output clusters, unlike the K-means algorithm.

Experimental results have shown that the Dynamic Time Warping supports higher-quality clustering than the Euclidean distance, in terms of both cluster separation and compactness. The best performance corresponded to setting TS-Part with the Dynamic Time Warping, which resulted in more clusters than those obtained by using the Euclidean distance. However, we observed that most of the data tend to group together in a relatively small number of clusters. This scenario enables the identification of relevant aspects which allow for supporting the design of tariff policies.

5.2.1 Low-voltage Electricity Customer Data

The Enel expertize in measuring and storing customer load profiles can be summarized in the Telegestore project [BCR05, Rog07]. Communication between meters and concentrator is accomplished by a PLC (Power Line Carrier) channel, whereas the public GPRS/GSM Network is responsible for the communication between the concentrator and the central system. All energy related data are first collected from the smart meters by the concentrator. Then, such data are uploaded by the central system. The Enel Telegestore network devices consist of more than 31 millions of smart meters and more than 350,000 concentrators installed and remotely managed [Rog07].

Enel smart meter is able to record and store active and reactive load profiles for all the four energy quadrants. A load profile represents the shape of the customer consumption chronologically ordered. Given a sampling period time, a smart meter logs the consumption corresponding to the associated location in a circular buffer. The sampling period is programmable and ranges from 1 to 60 minutes; as default, this is set to 15 minutes which allows for storing 38 days of load data, where each day is 96-sample long.

The smart meter also stores a flag register of "sample validity" in the stream of load data. This flag indicates a critical fault occurred during the sample measurement (e.g., a voltage interruption). In the experimental evaluation, we used this register to identify wrong samples and to correct each of them by using linear interpolation between the previous and the next valid sample.

5.2.2 Clustering Load Profile Data

Algorithms

According to most of research works on clustering load profiles, we resort to the well-known paradigm of *centroid-based partitional clustering* [JD88]. In

this context, we assume that the cluster centroids are computed as simple averages of the data (load profiles) in any specific cluster, since all the data have the same length in our setting. Of course, this assumption does not hold in general, and more refined methods for computing cluster centroids in time series data might be used [GPTG09].

The exemplary centroid-based partitional method is the popular K-means algorithm [McQ67]. In the experimental evaluation, we used the K-means algorithm as baseline method. We also developed a top-down partitional algorithm, named *TS-Part*. A major feature of TS-Part is that the number of output clusters is not required as a parameter, rather it is determined during the clustering task. This represents an advantage in many real application contexts, like ours, in which there is no a priori information which guides the user to properly set the number of output clusters.

TS-Part receives, as an input, a dataset $\mathcal{D} = \{T_1, \ldots, T_N\}$ of N load profiles modeled according to a time series based representation. It starts by considering the input dataset as a single cluster, then two main steps are iteratively repeated until the convergence is reached. The first step consists in finding the best split for each cluster in the current clustering. The second step recomputes the cluster centroids and reassigns all data according to the current clustering, similarly to the K-means algorithm. The convergence of the algorithm is reached when the split procedure does not perform any split.

Algorithm 4 TS-Part

Input: A set $\mathcal{D}$ of data objects
Output: A partition $\mathcal{C}$ of $\mathcal{D}$
1: $\mathcal{C} \leftarrow \{\mathcal{D}\}$
2: $\hat{\mathcal{C}} \leftarrow split(\mathcal{C})$ {Function 5}
3: **repeat**
4: $\quad \mathcal{C} \leftarrow relocation(\hat{\mathcal{C}})$
5: $\quad \hat{\mathcal{C}} \leftarrow split(\mathcal{C})$
6: **until** $\hat{\mathcal{C}} = \mathcal{C}$

In the splitting step, the quality of a given clustering solution is computed as the difference between the inter-cluster distance (i.e., the average pair-wise distance between all the cluster centroids) and the intra-cluster distance (i.e., the average distance between all the individual data within the cluster and the corresponding centroid). The split operation hence depends on a threshold of minimum quality, which is initially set as the quality of the input clustering.

Also, an *outlier* in a cluster is identified as an object whose distance from the associated centroid is maximum.

Assessment criteria

We evaluated compactness and separation of the solutions obtained by the clustering algorithms. More precisely, we employed two of the most used va-

Function 5 split

Parameters: A set $\mathcal{C} = \{C_1, \ldots, C_K\}$ of clusters
Returned data: A set $\hat{\mathcal{C}}$ of H clusters, where $H \geq K$

1: $\hat{\mathcal{C}} \leftarrow \emptyset$
2: $q_0 \leftarrow quality(\mathcal{C})$
3: **for all** $C \in \mathcal{C}$ **do**
4: $T^* \leftarrow outlier(C)$
5: $\mathcal{C}' \leftarrow C \setminus \{T^*\}, \quad q'_{max} \leftarrow quality(\mathcal{C}')$
6: $\mathcal{C}'' \leftarrow \{T^*\}, \quad q''_{max} \leftarrow quality(\mathcal{C}'')$
7: $splittable \leftarrow false$
8: **for all** $T \in C,\ T \neq T^*$ **do**
9: $q' \leftarrow quality(\mathcal{C}' \setminus \{T\})$
10: $q'' \leftarrow quality(\mathcal{C}'' \cup \{T\})$
11: $gain' \leftarrow (q' - q'_{max})/q'_{max}$
12: $gain'' \leftarrow (q'' - q''_{max})/q''_{max}$
13: **if** $gain' > q_0 \ \vee \ gain'' > q_0$ **then**
14: $q'_{max} \leftarrow q', \quad q''_{max} \leftarrow q''$
15: $\mathcal{C}' \leftarrow \mathcal{C}' \setminus \{T\}, \quad \mathcal{C}'' \leftarrow \mathcal{C}'' \cup \{T\}$
16: $splittable \leftarrow true$
17: **end if**
18: **end for**
19: **if** *splittable* **then**
20: $\hat{\mathcal{C}} \leftarrow \hat{\mathcal{C}} \cup \{\mathcal{C}'\} \cup \{\mathcal{C}''\}$
21: **else**
22: $\hat{\mathcal{C}} \leftarrow \hat{\mathcal{C}} \cup \{C\}$
23: **end if**
24: **end for**
25: **return** $\hat{\mathcal{C}}$

lidity criteria in load profile clustering, namely Mean Index Adequacy (MIA) and Clustering Dispersion Indicator (CDI) (e.g., [CNP+05, CNP06, THD07, TSTK08]). Both criteria are based on information on the data to be clustered, the centroids of the clustering solution, and the number of desired clusters.

Let $\mathcal{D} = \{T_1, \ldots, T_N\}$ be a dataset of load profiles modeled according to a time series based representation and $\mathcal{C} = \{C_1, \ldots, C_K\}$ be a clustering solution for $\mathcal{D}$. We denote with $\mathcal{R} = \{c_1, \ldots, c_K\}$ the set of centroids such that c_i is the centroid of the cluster $C_i, \forall i \in [1..K]$. We define the following distance measures:

$$d(c_i, C_i) = \sqrt{\frac{1}{|C_i|} \sum_{T \in C_i} dist(c_i, T)^2}$$

$$\hat{d}(\mathcal{R}) = \sqrt{\frac{1}{2N} \sum_{T \in \mathcal{D}} d(T, \mathcal{R})^2}$$

where $dist(\cdot, \cdot)$ is a distance measure for comparing time series, e.g., Euclidean or Dynamic Time Warping distance.

Based on the above formulas, the Mean Index Adequacy (MIA) and Clustering Dispersion Indicator (CDI) validity criteria are defined as follows:

$$MIA(\mathcal{C}) = \sqrt{\frac{1}{K}\sum_{i=1}^{K} d(c_i, C_i)^2}$$

$$CDI(\mathcal{C}) = \frac{1}{\hat{d}(\mathcal{R})}\sqrt{\frac{1}{K}\sum_{k=1}^{K} \hat{d}(C_k)^2}$$

MIA measures the compactness of a clustering solution by averaging the distances between each object within a cluster and its centroid. CDI expresses the degree of cluster separation as directly proportional to the average of the intra-cluster distance between the objects within the same cluster and inversely proportional to the pair-wise distances between the cluster centroids. For both criteria, lower values correspond to higher quality clustering solutions.

5.2.3 Experimental Evaluation

Settings

Data description and preparation

We were granted access to about 30,000 Enel Italian LV customer load profiles, measured during the period between the first week of February 2009 and the last week of March 2009. All the load profiles have been provided in anonymous form.

The load profile set was preliminarily partitioned according to meta-data associated to each individual customer. Such meta-data represents commercial and technical extra attributes that Enel provided with each load profiles. Specifically, customer meta-data includes the following attributes:

- Meter type: specifies the power capacity and the number of phases (i.e, single-phase, multi-phase) of the meter associated to the customer;
- Contractual power: the maximum contractual power allowed to the customer;
- Contract date: the start date of the customer's contract;
- Commercial category: identifies the type of customer, including residential domestic, non-residential domestic, public lightning, etc.;
- Product category: identifies a particular (private or public) usage of the energy contract;
- Zone: refers to the geographical location of the customer.

According to the above information, we filtered in the available load profiles which correspond to the most common customer type, namely the "private", "residential domestic" customer. We considered only the active energy part of each load profile. The resulting 5,000 load profiles were segmented in order to extract *daily* profiles. Since each daily profile is comprised of 96 samples, we obtained 30 daily profiles of 96 samples from each customer profile. Moreover, daily profiles were further partitioned depending on the type of day; precisely, we distinguished "weekdays" profiles from "saturdays" profiles and "sundays/holidays" profiles. According to this classification, we found 132,041 "weekdays", 24,527 "saturdays", and 24,541 "sundays/holidays" daily profiles.

The Rialto suite for data mining

Experiments and analysis described in this section were conducted using *Exeura Rialto™* [Ria]. Rialto is a graphical environment for performing data mining and knowledge discovery tasks. In contrast to other similar data mining tools, Rialto contains most of the functionalities required by one user-friendly tool that allows users to design, create, explore, analyze, and execute data mining tasks, as well as to deploy predictive and descriptive models into other tools, applications, and systems. Thanks to the possibility of extending the capabilities of Rialto, it was possible to generate a set of ad-hoc plug-ins for managing the data from the Enel legacy repositories.

Results

We present here main results from clustering experiments on the three types of daily load profile sets, namely "weekdays", "saturdays", and "sundays/holidays". For each of the three cases, we performed multiple runs of both clustering algorithms (i.e., K-means and TS-Part) and finally averaged the quality results, in terms of MIA and CDI, obtained over the runs. Each algorithm was equipped with Euclidean distance or DTW as distance measure. For each setting, the number of clusters was determined by TS-Part and then used to set the parameter (i.e., initial value of the number of output clusters) for the K-means.

Tables 5.3–5.5 summarize the best (average) performance of the clustering algorithms obtained on the three cases. Using DTW as distance measure enabled both clustering algorithms to produce higher quality clustering solutions than the corresponding ones obtained by using the Euclidean distance. This always holds in terms of CDI as well as MIA for all the cases. The better separation (CDI) and compactness (MIA) obtained by using DTW can be explained since DTW is more sensitive than the Euclidean distance to time shifts and, consequently, it is capable of detecting clusters that are more homogeneous (i.e., each cluster contains load profiles that have similar consumption trends).

clustering algorithm	*distance measure*	*# of clusters*	*MIA*	*CDI*
K-means	Euclidean	5	37.24	2.82
TS-Part	Euclidean	5	23.86	1.58
K-means	DTW	5	13.85	0.42
TS-Part	DTW	5	**12.36**	**0.13**

Table 5.3. Best (average) performance of clustering: Weekdays load profiles

clustering algorithm	*distance measure*	*# of clusters*	*MIA*	*CDI*
K-means	Euclidean	10	20.62	0.41
TS-Part	Euclidean	10	18.93	0.32
K-means	DTW	10	11.75	0.04
TS-Part	DTW	10	**9.12**	**0.02**

Table 5.4. Best (average) performance of clustering: Saturdays load profiles

clustering algorithm	*distance measure*	*# of clusters*	*MIA*	*CDI*
K-means	Euclidean	10	25.36	0.46
TS-part	Euclidean	10	17.52	0.04
K-means	DTW	10	11.84	0.29
TS-part	DTW	10	**9.62**	**0.02**

Table 5.5. Best (average) performance of clustering: Sundays/holidays load profiles

The best results in each table correspond to the use of our TS-Part algorithm equipped with DTW. It should be noted that TS-Part performed better than K-means in terms of both MIA and CDI. Also, the number of clusters detected in "weekdays" was generally lower than the "saturdays" and "sundays/holidays" cases, which prompted us to explore a more variegate scenario of consumption trends in non-weekdays.

Figure 5.8 shows active energy profiles of the centroids belonging to the most representative clusters obtained by TS-Part equipped with DTW.

From a qualitative viewpoint, we observed that most of the energy consumption is concentrated on the lunch and dinner hours, in typical behaviors of residential domestic customers. We also found in some clusters a few anoma-

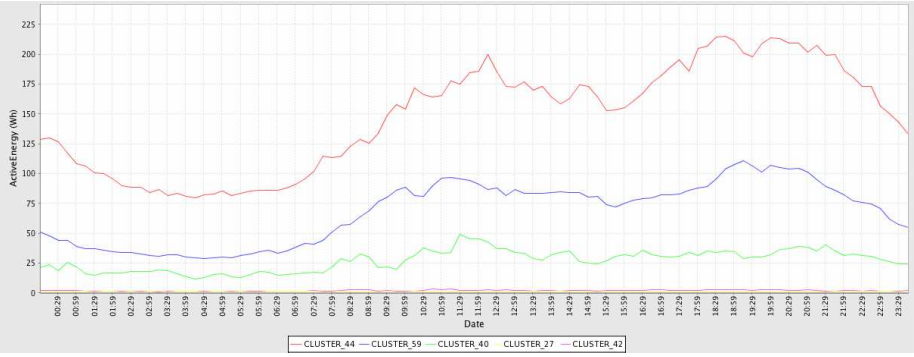


Fig. 5.8. Active energy profiles of sample cluster centroids for the most relevant cluster

lies which likely correspond to untypical habits of certain residential domestic customers (e.g., customers spending most of their time in residences which are located in places different from those declared in the contract).

Part III

Uncertain Data Clustering

6

Uncertain Data Clustering: State of the Art

Summary. This chapter provides the state-of-the-art in uncertain data clustering. In this context, researches have gone in several directions, which aim to analyze different problems related to uncertain data. The focus here is on data mining applications, and the various proposals can be divided into partitional methods and density-based methods.

6.1 Uncertain Data Clustering

In the context of uncertain data management, a lot of research has been mainly focused on data representation and modeling, indexing, query processing, and data mining (e.g., [DS07, AY09, ZLPZ08]). In particular, data mining applications have involved various tasks, such as classification [BZ04], outlier detection [AY08], association analysis [CKH07], and clustering [KP05b, KP05a, CCKN06, NKC+06, LKC07, KLC+08]. As regards clustering, it is possible divide algorithms into two families: *partitional* and *density-based* methods

6.1.1 Partitional Methods

One of the earliest attempts to solve the problem of clustering uncertain data is the partitional algorithm *UK-means* [CCKN06], which is an adaptation of the popular K-means [McQ67] designed for handling uncertain objects. UK-means suffers from a major weakness, that is the expensive computation of the *expected distance* (ED) between uncertain objects and cluster centroids (which are defined as deterministic objects), at each iteration of the algorithm.

In order to improve the UK-means efficiency, [NKC+06] proposes some pruning techniques to avoid the computation of redundant EDs. Such techniques make use of lower- and upper-bounds ad-hoc defined for each ED to be calculated, in order to define a specific bounding-box for each uncertain object; these boxes allow for eliminating some candidate assignments of objects to clusters and avoiding the corresponding ED computation. However,

a major problem of this approach is that it cannot guarantee high pruning (hence, high efficiency), as it depends on the features of the objects in the specific dataset. Another pruning method to reduce the number of EDs calculation is described in [KLC+08]. Such a method exploits Voronoi diagrams and has been recognized as more effective than the basic bounding-box-based techniques. In [LKC07], the *CK-means* algorithm is proposed as a variant of UK-means that exploits the moment of inertia of rigid bodies in order to reduce the execution time needed for computing EDs. Unfortunately, the soundness of the CK-means criterion for the ED computation is guaranteed only if the mean squared error for the definition of the EDs is used and the distance function is based on the Euclidean norm.

The methods proposed in [NKC+06, LKC07, KLC+08] attempt to reduce the execution time required for computing the EDs. The Approximation by Single Gaussian (ASG) method [XH07] belongs to a more general approach, which aims to approximate the distance between an uncertain object and another uncertain object. ASG has been proved to achieve accuracy close to the exact distance calculation while improving the efficiency. Another approach to the distance computation consists in defining a univariate pdf, or *fuzzy distance function*, for each pair of uncertain objects; this univariate pdf stores a probability for each distance value for two given objects. The final distance between the objects is computed by deriving an aggregated, representative value (e.g., expected value) from the pdf of those objects. This method has been presented originally in [KP05b] and proved to be more effective than the standard Euclidean distance applied to vectors of deterministic values.

6.1.2 Density-based Methods

Devising a fuzzy distance function is a key aspect in density-based approaches to clustering uncertain objects [KP05a, KP05b]. In [KP05a], the $\mathcal{F}DBSCAN$ is proposed as a fuzzy version of the popular DBSCAN [EKSX96], which uses fuzzy distance functions to compute the core object and reachability probabilities. A similar approach is presented in $\mathcal{F}OPTICS$ [KP05b]. Like the well-known density-based clustering algorithm OPTICS [ABKS99], $\mathcal{F}$OPTICS produces an augmented ordering of the objects based on the notion of fuzzy object reachability-distance; this ordering can eventually be used to derive a cluster hierarchy.

While the majority of algorithms proposed for clustering uncertain objects are based on either partitional or density-based schemes, it should be noted that there is poor research on (agglomerative) hierarchical clustering of uncertain data. $\mathcal{F}$OPTICS is close to a hierarchical scheme, although is significantly different in constructing the cluster hierarchy. Indeed, $\mathcal{F}$OPTICS outputs a reachability plot, whereas the result of our algorithm is a dendrogram. While reachability plots may be easier to read than dendrograms for very large datasets, a dendrogram gives a clearer view of the cluster membership of individual objects than a reachability plot for a wide set of real cases [SQL+03];

also, when a reachability plot is produced, building the corresponding cluster hierarchy needs for a further step, which is typically performed by employing either visualization or automatic techniques [ABKS99]. Another important remark is that our U-AHC does not require any input parameter; by contrast, $\mathcal{F}$OPTICS requires in input a threshold for the distance of the neighbors and the minimum number of points in the neighborhood of every object.

7

Clustering of Uncertain Objects via *K*-medoids

Summary. In this chapter, we address the problem of clustering uncertain data by proposing a K-medoids-based algorithm, called *UK-medoids*. This algorithm is designed to overcome the two main problems related to the centroid-based approach to clustering uncertain objects used in the UK-Means, that are (i) an accuracy issue, since cluster centroids are computed as deterministic objects using the expected values of the pdfs of the clustered objects, and (ii) an efficiency issue, since the expected distance between uncertain objects and cluster centroids is computationally expensive. UK-medoids employs distance functions properly defined for uncertain objects, and exploits a K-medoids scheme. Experiments have shown that UK-medoids outperforms existing algorithms from an accuracy viewpoint while achieving reasonably good efficiency.

7.1 Introduction

As mentioned in Chapter 6, the K-means algorithm has been adapted to the uncertain data domain [CCKN06]. However, the resulting algorithm, named UK-means, has two major weak points. First, cluster centroids are defined as deterministic objects and computed as the mean of the expected values over the pdfs of the uncertain objects in the cluster; defining centroids in this way may result in loss of accuracy, since only the expected values of the pdfs of the uncertain objects are taken into account. Second, the computation of the Expected Distance (ED) between cluster centroids and uncertain objects is computationally expensive, as it requires non-trivial numerical integral estimations; this represents an efficiency bottleneck at each iteration of the algorithm.

In this chapter, *UK-medoids* will be presented, which is an algorithm for clustering uncertain objects based on the K-medoids clustering scheme. It exploits a distance function for uncertain objects, which is not limited to consider only scalar values derived from the pdfs associated to the objects (e.g., pdf expected values). This allows for better estimating the real distance between two uncertain objects, leading to significant improvement of the clustering

quality. Also, our algorithm does not require any expensive operation to be repeated at each iteration; indeed, the computation of the distances between uncertain objects in the dataset is performed only once, thus guaranteeing a significant improvement of the efficiency with respect to UK-means.

7.2 Uncertain Distance

To measure the distance between uncertain objects, we need to devise a suitable notion of *uncertain distance*, which is involved in the proposed clustering algorithm. Uncertain distance is defined in terms of an *uncertain distance function*. In order to make the uncertain distance independent from the chosen uncertainty model, we provide definitions of uncertain distance function for both multivariate and univariate uncertainty models.

Definition 7.1 (uncertain distance function). *Given a set of uncertain objects $\mathcal{D} = \{o_1, \ldots, o_N\}$, the* uncertain distance function *defined over $\mathcal{D}$ is a function $\Delta : \mathcal{D} \times \mathcal{D} \times \Re \rightarrow \Re_0^+$, for which the following conditions hold:*

$$\int_{x \in \Re} \Delta(o_u, o_v, x)\ \mathrm{d}x = 1, \quad \forall o_u, o_v \in \mathcal{D},$$

$$\Delta(o_u, o_v, x) = \begin{cases} 1, & \textit{if } \ u = v, x = 0 \\ 0, & \textit{if } \ u = v, x \neq 0 \end{cases}$$

For any pair of uncertain objects $o_u, o_v, u \neq v$, Δ can be derived from the pdfs associated to the uncertain objects. The definition of Δ depends on the uncertainty model used for representing o_u and o_v (cf. Section 2.2.2).

Uncertain distance function for multivariate objects

If $o_u = (R_u, p_u)$, $o_v = (R_v, p_v)$ are M-dimensional multivariate uncertain objects, Δ is defined as:

$$\Delta(o_u, o_v, x) = \int_{\mathbf{x} \in R_u} \int_{\mathbf{y} \in R_v} I[f(\mathbf{x}, \mathbf{y}) = x]\ p_u(\mathbf{x})\ p_v(\mathbf{y})\ \mathrm{d}\mathbf{x}\ \mathrm{d}\mathbf{y} \tag{7.1}$$

where $f(\mathbf{x}, \mathbf{y})$ is a distance measure between any pair $\mathbf{x}, \mathbf{y} \in \Re^M$ (e.g., Euclidean norm), and $I[A]$ is the *indicator function*, which is equal to 1 when the event A occurs, 0 otherwise.

Uncertain distance function for univariate objects

If $o_u = ((I_u^{(1)}, p_u^{(1)}), \ldots, (I_u^{(M)}, p_u^{(M)}))$, $o_v = ((I_v^{(1)}, p_v^{(1)}), \ldots, (I_v^{(M)}, p_v^{(M)}))$ are M-dimensional univariate uncertain objects, Δ is defined as:

$$\Delta(o_u, o_v, x) = \int_{x_1 \in \Re} \cdots \int_{x_M \in \Re} I[f\prime(x_1, \ldots, x_M) = x] \prod_{h=1}^{M} \Psi^{(h)}(o_i, o_j, x_h) \, \mathrm{d}x_1 \cdots \mathrm{d}x_M \tag{7.2}$$

where

- $\Psi^{(h)} : \mathcal{D} \times \mathcal{D} \times \Re \to \Re$,
- $\Psi^{(h)}(o_u, o_v, x_h) = \int_{r \in I_u^{(h)}} \int_{s \in I_v^{(h)}} I[|r - s| = x_h] \; p_u^{(h)}(r) \; p_v^{(h)}(s) \; \mathrm{d}r \; \mathrm{d}s, \quad h \in [1..M]$,
- $f\prime : \Re^m \to \Re$ is a function that computes a scalar value from the components of a vector $(x_1, \ldots, x_M)$. In the following, we assume that $f\prime = (1/M)\sqrt{\sum_{h=1}^{M} {x_h}^2}$.

It can be proved that the condition $\int_{x \in \Re} \Delta(o_u, o_v, x) \, \mathrm{d}x = 1$ holds for both the definitions of Δ, for all o_u, o_v in the dataset.

Given an uncertain distance function Δ, we now provide a definition of uncertain distance by extracting a single, well-representative numerical value from Δ.

Definition 7.2 (uncertain distance). *Given a set of uncertain objects $\mathcal{D} = \{o_1, \ldots, o_N\}$, let Δ be the uncertain distance function defined over $\mathcal{D}$. The* uncertain distance *is a function $\delta : \mathcal{D} \times \mathcal{D} \to \Re_0^+$, which is defined as:*

$$\delta(o_u, o_v) = \int_{x \in \Re} x \Delta(o_u, o_v, x) \, \mathrm{d}x \tag{7.3}$$

According to (7.3), $\delta(o_u, o_v)$ is the expected value of the uncertain distance function Δ between o_u and o_v. Note that, if o_u, o_v are multivariate uncertain objects, $\delta(o_u, o_v)$ can be directly computed as:

$$\delta(o_u, o_v) = \int_{\mathbf{x} \in R_u} \int_{\mathbf{y} \in R_v} f(\mathbf{x}, \mathbf{y}) \; p_u(\mathbf{x}) \; p_v(\mathbf{y}) \; \mathrm{d}\mathbf{x} \; \mathrm{d}\mathbf{y} \tag{7.4}$$

whereas, if o_u, o_v are univariate uncertain objects, $\delta(o_u, o_v)$ can be calculated as:

$$\delta(o_u, o_v) = f\prime(\psi^{(1)}(o_u, o_v), \ldots, \psi^{(M)}(o_u, o_v)) \tag{7.5}$$

where

$$\psi^{(h)}(o_u, o_v) = \int_{x \in I_u^{(h)}} \int_{y \in I_v^{(h)}} |x - y| \; p_u^{(h)}(x) \; p_v^{(h)}(y) \; \mathrm{d}x \; \mathrm{d}y, \quad h \in [1..M].$$

Algorithm 6 UK-medoids

Input: a set of uncertain objects $\mathcal{D} = \{o_1, \ldots, o_N\}$;
the number of output clusters K
Output: a set of clusters $\mathcal{C}$

1: compute distances $\delta(o_u, o_v), \forall o_u, o_v \in \mathcal{D}$
2: compute the set $\mathcal{M} = \{m_1, \ldots, m_K\}$ of initial medoids
3: **repeat**
4: $\mathcal{M}' \leftarrow \mathcal{M}$
5: $\mathcal{M} \leftarrow \emptyset$
6: $\mathcal{C} = \{C_1, \ldots, C_K\} \leftarrow \{\emptyset, \ldots, \emptyset\}$
7: **for all** $o \in \mathcal{D}$ **do**
8: {*assign each object to the closest cluster, based on its uncertain distance to cluster medoids*}
9: $m_j \leftarrow \arg\min_{o' \in \mathcal{M}'} \delta(o, o')$
10: $C_j \leftarrow C_j \cup \{o\}$
11: **end for**
12: **for all** $C \in \mathcal{C}$ **do**
13: {*recompute the medoid of each cluster*}
14: $m \leftarrow \arg\min_{o \in C} \sum_{o' \in C} \delta(o, o')$
15: $\mathcal{M} \leftarrow \mathcal{M} \cup \{m\}$
16: **end for**
17: **until** $\mathcal{M} \neq \mathcal{M}'$

7.3 UK-medoids Algorithm

In this section we present our K-medoids-based algorithm for clustering uncertain objects, named *UK-medoids*. The outline of UK-medoids is given in Algorithm 6.

The input for the UK-medoids algorithm is a dataset $\mathcal{D}$ of N uncertain objects and the number K of clusters to be discovered, and the output is a set $\mathcal{C}$ of K clusters. Initially, all the uncertain distances between any pair of objects $o_u, o_v \in \mathcal{D}$ are computed (Line 1). The distances are calculated only once and are used at each iteration of the algorithm. Then, the set of K initial medoids is computed (Line 2). The initial medoids can be selected by means of either random chance or a suitable procedure aimed to choose well-separated medoids (e.g., that proposed for the *Partitioning Around Medoids* (PAM) algorithm [KR87]).

After the initialization steps, the algorithm performs the main loop (starting from Line 3) which is comprised of two phases. In the first phase (Lines $7-11$), each object o in $\mathcal{D}$ is assigned to the cluster represented by the medoid m closest to o. In the second phase, the medoids in the set $\mathcal{M}$ are recomputed according to the objects assigned to each cluster (Lines $12-16$). Such phases are iteratively repeated until a local optimum has not been reached, i.e., there has been some change in the current $\mathcal{M}$ with respect to the previous iteration (Line 17).

Proposition 7.3. *Given a dataset $\mathcal{D}$ of N uncertain objects, Algorithm 6 works in $\mathcal{O}(N^2 \times I)$, where I is the maximum number of iterations.*

7.4 Experimental Evaluation

We devised an experimental evaluation aimed to assess the ability of our algorithm in clustering uncertain objects, both in terms of accuracy and efficiency. We also compared our UK-medoids to K-means based uncertain data clustering algorithms, i.e., UK-means and its variant CK-means.

7.4.1 Settings

Datasets

Experimental analysis was performed on benchmark datasets from the UCI Machine Learning Repository.[1] We chose four datasets with numerical real-value attributes, namely Iris, Wine, Glass, and Ecoli.

Table 7.1. Datasets used in the experiments

dataset	*objects*	*attributes*	*classes*
Iris	150	4	3
Wine	178	13	3
Glass	214	10	6
Ecoli	327	7	5

Table 8.1 shows the main characteristics of the datasets. Iris contains measurements on different iris plants. Wine reports results of a chemical analysis of Italian wines derived from three different cultivars. In Glass, each glass instance is described by the values of its chemical components. Ecoli contains data on the Escherichia Coli bacterium, which are identified with values coming from different analysis techniques.

All the selected datasets originally contain deterministic values, hence the uncertainty was synthetically generated for each object of any dataset. Let us now describe how univariate and multivariate uncertain objects were synthetically generated.

- Generation of univariate uncertainty — For each univariate object o, we generated the uncertain interval $I^{(h)}$ and the pdf $p^{(h)}$ defined over $I^{(h)}$, for each attribute $A^{(h)}$, $h \in [1..M]$. The interval $I^{(h)}$ was randomly chosen as a subinterval within $[min_{o_h}, max_{o_h}]$, where min_{o_h} (resp. max_{o_h}) is the

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

minimum (resp. maximum) deterministic value of the h-th attribute, over all the objects belonging to the same ideal class of o.
As concerns $p^{(h)}$, we considered Uniform, Normal and Gamma pdfs. We set the parameters of Normal and Gamma pdfs in such a way that their mode corresponded to the deterministic value of the h-th attribute of the object o.

- Generation of multivariate uncertainty — Similarly to the univariate case, for each multivariate object o we generated the uncertainty region R as the product of the intervals randomly generated for each attribute of o. The distributions involved were (multivariate) Uniform and Normal pdfs. The strategy for setting the parameters of the pdfs was the same as for the univariate case.

We performed experiments for multivariate uncertain objects as well. In this case, we generated uncertainty starting from the univariate model, assuming statistical independence for the pdfs of the attributes of any object. Since univariate and multivariate models gave similar results, here we report only results on the univariate models for the sake of brevity.

Clustering validity criteria

To assess the quality of clustering solutions we exploited the availability of reference classifications for the datasets. The objective was to evaluate how well a clustering fits a predefined scheme of known classes (natural clusters). To this purpose, we resorted to the F_1-measure external validity criteria (cf. Section 2.1.6).

Setup of the clustering methods

In K-means based approaches, the set of initial centroids is randomly selected. Therefore, to avoid that clustering results were biased by random chance, we averaged accuracy and efficiency measurements over 100 different runs. We made a similar choice also for UK-medoids, since we noted that the use of a refined strategy for selecting initial medoids (e.g., the procedure proposed in [KR87]) gave no significant improvement with respect to random selection.

We computed the integrals involved into the distances calculation by taking into account lists of samples derived from the pdfs. To accomplish this, we employed the classic *Monte Carlo* sampling method.[2] We also performed a preliminary tuning phase to properly set the number of samples S; in particular, for each method and dataset, we chose S in such a way that there was no significant improvement in accuracy for any $S' > S$. In general, the optimal S depended on the width of the uncertainty interval/region; however, according to our experiments, 50 and $400 \div 500$ samples represented a reasonably good choice, for univariate and multivariate uncertainty model, respectively.

[2] We used the SSJ library, available at http://www.iro.umontreal.ca/∼simardr/ssj/

7.4.2 Results

Accuracy

Table 7.2 summarizes the F_1-measure results obtained by UK-medoids and the other methods.

Table 7.2. Clustering quality results (F_1-measure)

dataset	*pdf*	UK-means	CK-means	**UK-medoids**
Iris	Uniform	0.45	*0.50*	**0.84**
	Normal	0.84	*0.85*	**0.88**
	Binomial	*0.62*	0.58	**0.87**
Wine	Uniform	0.46	*0.50*	**0.80**
	Normal	0.69	*0.70*	**0.70**
	Binomial	*0.63*	0.58	**0.73**
Glass	Uniform	0.26	*0.29*	**0.71**
	Normal	*0.63*	0.59	**0.68**
	Binomial	0.27	*0.29*	**0.67**
Ecoli	Uniform	0.30	*0.33*	**0.73**
	Normal	0.73	*0.74*	**0.77**
	Binomial	*0.50*	0.44	**0.72**

We can observe that UK-medoids drastically outperformed UK-means and CK-means on all the datasets, with Uniform and Binomial pdfs. In particular, compared to best competing method, the accuracy improvement obtained by our UK-medoids was from 34% to 42% with Uniform pdfs and from 10% to 38% with Binomial pdfs.

In case of Normal pdfs, UK-medoids performed $3 \div 5\%$ better than the other methods on three datasets, whereas all the methods behaved similarly in Wine. The reduction of gap between UK-medoids and K-means based approaches on Normal pdfs can be explained in that, according to our uncertainty generation scheme, the expected value of a Normal pdf associated to any attribute of each uncertain object was set equal to the deterministic value of the attribute for that object. This allowed the centroid generation strategy of UK-means and CK-means to perform well in that case.

It should be also noted that UK-means and CK-means performed similarly for all the pdfs and datasets, as expected, since they employ a similar clustering scheme; the only differences between the two methods are due to random choices, such as selection of initial centroids and pdf sampling for the computation of the integrals.

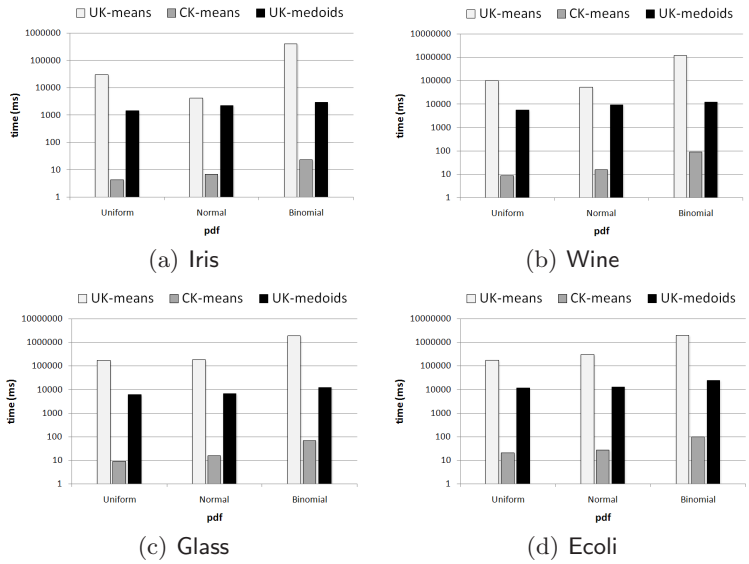


Fig. 7.1. Clustering time performances

Efficiency

To evaluate the efficiency of UK-medoids and the competing methods, we measured their time performances in clustering uncertain objects.[3] Figure 7.1 shows the total execution times (in milliseconds) obtained by the methods on the various datasets. For UK-medoids and CK-means, we calculated the sum of the times obtained for the pre-computing phase (i.e., uncertain distances computation for UK-medoids and cluster centroids computation for CK-means), together with the algorithm runtimes.

In the figure, it can be noted that our UK-medoids was $1 \div 2$ orders of magnitude faster than UK-means, which was the slowest method on all datasets. The slowness of UK-means is mainly due to the EDs computation needed for each object in the dataset, at each iteration of the algorithm.

As expected, CK-means outperformed UK-medoids on all datasets, which is explained by a difference between the computational complexities of the two algorithms. Indeed, both the phases of pre-computing and algorithm execution are quadratic (resp. linear) with the number of objects in the dataset for UK-medoids (resp. CK-means). However, it should be emphasized that the CK-means algorithm is less general than the other methods, as it works only if the mean squared error for the definition of the EDs is used and the distance function is based on the Euclidean norm.

[3] Experiments were conducted on a platform Intel Pentium IV 3GHz with 2GB memory and running Microsoft WinXP Pro

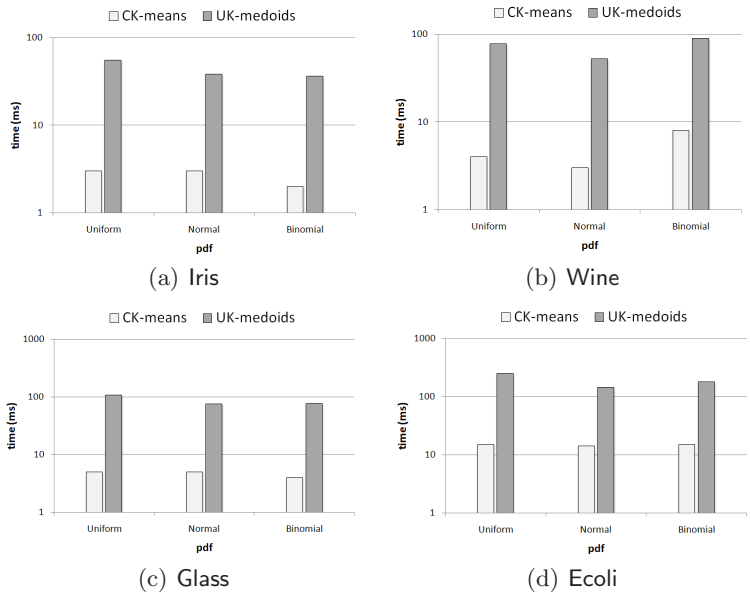


Fig. 7.2. Performance of the algorithm runtimes (pre-computing phases are ignored)

We also measured separately the times of the pre-computing phases, which involve the calculation of uncertain distances (in UK-medoids) and cluster centroids (in CK-means). Figure 7.2 shows that the gap between UK-medoids and CK-means was reduced with respect to that measured by including the total runtimes (Figure 7.1). This result confirms that the major difference between UK-medoids and CK-means is given by the pre-computing phase. Thus, in case of multiple runs of the two algorithms, we can state that the performance of UK-medoids and CK-means are comparable, since the pre-computing phase has to be performed once.

8

Information-Theoretic Hierarchical Clustering of Uncertain Objects

Summary. Recently, there has been a growing interest in clustering uncertain data, with a special emphasis on partitional and density-based approaches, whereas hierarchical clustering schemes have drawn less attention. In this chapter, we propose a centroid-linkage-based agglomerative hierarchical method for clustering uncertain objects, named *U-AHC*. A major novelty of this research lies in a well-founded information-theoretic approach to the computation of distance between uncertain objects, which is employed in the proposed hierarchical clustering framework. Theoretical analysis have been conducted to prove the validity of the intuitions regarding the distance function between uncertain objects and the definition of cluster prototypes as uncertain objects. Experiments conducted on various datasets have shown that our method outperforms state-of-the-art methods for clustering uncertain data from an accuracy viewpoint while achieving efficiency comparable to density-based clustering methods.
The chapter concludes by showing the way in which our uncertainty modeling strategy and our U-AHC scheme can be involved in a real case application, which consists in analyzing microarray data with probe-level uncertainty. We have conducted experiments on four large microarray datasets, in order to assess effectiveness of the proposed clustering method. Experimental results have shown high quality results in terms of compactness of the clustering solutions.

8.1 Introduction

In recent years, clustering of uncertain objects has been deeply investigated from the point of view of both partitional algorithms exploiting a relocation scheme (i.e., approaches K-Means- or K-Medoids-based) and density-based algorithms. In this view, we decide to investigate other possibilities and other approaches to perform clustering of uncertain objects in an highly-effective way. Since there are no proposals for hierarchical clustering schemes in this area, we propose an agglomerative hierarchical clustering framework that is able to handle uncertain objects. Within this view, *U-AHC* has been developed, which is a *centroid-linkage-based agglomerative hierarchical* algorithm

for clustering uncertain objects. To the best of our knowledge, this research brings for the first time a hierarchical approach to group uncertain data.

The proposed *U-AHC* algorithm has a number of features. First, the prototype (centroid) of any given cluster is computed as a mixture model that summarizes the probability distributions of all the objects within that cluster. Therefore, a cluster prototype is an uncertain object itself. The idea of adopting a centroid-based linkage measure to define, at each step of the algorithm, the two closest clusters to be merged may become crucial in uncertainty similarity detection, especially from an efficiency viewpoint, since it suffices to calculate only the distance between uncertain cluster prototypes, instead on computing the distance for each pair of uncertain objects in the two clusters. In addition, it has been also proved that this definition of cluster prototype is well-suited to represent uncertain objects.

Second, we propose a different approach to the computation of distance between uncertain objects. The intuition behind this proposal is based on two different ways of comparing probability distributions that represent uncertain data objects, that are (i) measuring the distance by involving the whole probability distributions, and (ii) computing the difference between the expected values of the distributions. To achieve such issues, we define a way to compare information-theoretic based distance measure and aggregate information coming from expected value, obtaining a new distance measure for uncertain objects. Since we defined cluster prototypes as uncertain objects, the hierarchical clustering scheme proposed equipped with this new distance measure is able to find, at each step of the computation, the two closest clusters to be merged.

The intuition behind this definition of distance lies in the fact that comparing two distributions by an information-theoretic distance is powerful but, in principle, not always applicable; on the contrary, expected value of distributions is always computable but represents a concise information which, in general, is not able to capture the real proximity (which also depends on the shapes) between probability distributions. A deep insight into the properties of the compound distance measure introduced has been provided, by demonstrating its effectiveness and soundness in comparing probability distributions of uncertain objects. Furthermore, a notion of *adequacy* of computing the distance between any two probability distributions (of uncertain objects) by means of a given information-theoretic distance has been introduced. Intuitively, this notion expresses to what degree an information-theoretic distance measure is worth comparing two uncertain objects by involving only their pdfs. Indeed, a high value of adequacy implies that the distance computation can exploit most of the information in the probability distributions of the objects, thus ensuring high accuracy in detecting dissimilarities. It has been also proved that the adequacy of comparing any two probability distributions provides an upper-bound for the computation of the information-theoretic measure adopted in our framework.

In the following, we present our definition of uncertain cluster prototype, the distance measure used to compare them, and then the clustering framework for uncertain objects. Finally, we also involve our U-AHC in a real case application, which consists in clustering microarray data with probe-level uncertainty. The proposed analysis methodology is shown in Section 8.6.

8.2 Uncertain Prototype

In this section, we introduce the notion of *uncertain prototype*, or simply *prototype*, as a new uncertain object computed to properly summarize the features of all the uncertain objects in a given set. Since uncertain objects are represented by probability distributions, it is reasonable to represent an uncertain prototype as a *finite mixture*, whose components are the pdfs associated to the objects within the set to be summarized.

Finite mixtures have long been used to model various phenomena in which more (independent) variables sum to a whole, being this characterized by the fraction of each variable (component) [MP00]. Applications have arisen in disjoint scientific disciplines and led to the development of several variants and extensions for special cases. As concerns data clustering, finite mixture models provide a foundation for probabilistic clustering (e.g., mixtures are often used to model the form of cluster members).

In the following, we provide the definitions of uncertain prototype for both multivariate and univariate uncertainty models.

Definition 8.1 (multivariate uncertain prototype). *Let* $C = \{o_1, \ldots, o_n\}$ *be a set of* n *multivariate uncertain objects, where* $o_u = (R_u, p_u)$, $R_u = [L_u^{(1)}, U_u^{(1)}] \times \ldots \times [L_u^{(M)}, U_u^{(M)}]$, *for each* $u \in [1..n]$. *The* multivariate uncertain prototype *of* C *is a pair* $P_C = (R_C, p_C)$, *where*

$$R_C = \left[\min_{u \in [1..n]} L_u^{(1)}, \max_{u \in [1..n]} U_u^{(1)} \right] \times \cdots \times \left[\min_{u \in [1..n]} L_u^{(M)}, \max_{u \in [1..n]} U_u^{(M)} \right] \quad (8.1)$$

$$p_C(\mathbf{x}) = \frac{1}{n} \sum_{u=1}^{n} p_u(\mathbf{x}) \quad (8.2)$$

Definition 8.2 (univariate uncertain prototype). *Let* $C = \{o_1, \ldots, o_n\}$ *be a set of* n *univariate uncertain objects, where* $o_u = ((I_u^{(1)}, p_u^{(1)}), \ldots, (I_u^{(M)}, p_u^{(M)}))$, $I_u^{(h)} = [L_u^{(h)}, U_u^{(h)}]$, *for each* $h \in [1..M]$, $u \in [1..n]$. *The* univariate uncertain prototype *of* C *is a tuple* $P_C = ((I_C^{(1)}, p_C^{(1)}), \ldots, (I_C^{(M)}, p_C^{(M)}))$ *such that, for each* $h \in [1..M]$*:*

$$I_C^{(h)} = \left[\min_{u \in [1..n]} L_u^{(h)}, \max_{u \in [1..n]} U_u^{(h)} \right] \quad (8.3)$$

$$p_C^{(h)}(x) = \frac{1}{n}\sum_{u=1}^{n} p_u^{(h)}(x) \tag{8.4}$$

Proposition 8.3. *Let $C = \{o_1, \ldots, o_n\}$ be a set of n uncertain objects. The uncertain prototype P_C is an uncertain object.*

Proof. Being C a set of multivariate uncertain objects, to prove that $P_C = (R_C, p_C)$ is a (multivariate) uncertain object, we need to demonstrate that:

1. p_C is a pdf,
2. Equation (2.1) of Definition 2.2 holds, and
3. Equation (2.2) of Definition 2.2 holds.

Condition (1) is true since p_C represents a mixture of pdfs of the form $\sum_{u=1}^{n} \alpha_u p_u(\mathbf{x})$, where $\alpha_u = 1/n$. As concerns condition (2), we have that:

$$\int_{\mathbf{x}\in\Re^M\setminus R_C} p_C(\mathbf{x})\mathrm{d}\mathbf{x} = \\ = \int_{\mathbf{x}\in\Re^M\setminus R_C} \frac{1}{n}\sum_{u=1}^{n} p_u(\mathbf{x})\mathrm{d}\mathbf{x} \;=\; \frac{1}{n}\sum_{u=1}^{n}\int_{\mathbf{x}\in\Re^M\setminus R_C} p_u(\mathbf{x})\mathrm{d}\mathbf{x}$$

Since $R_C \supseteq R_u$, $\forall u \in [1..n]$ (Equation (8.1)), we have:

$$\frac{1}{n}\sum_{u=1}^{n}\int_{\mathbf{x}\in\Re^M\setminus R_C} p_u(\mathbf{x})\mathrm{d}\mathbf{x} = \\ = \frac{1}{n}\sum_{u=1}^{n}\left(\int_{\mathbf{x}\in\Re^M\setminus R_u} p_u(\mathbf{x})\mathrm{d}\mathbf{x} \;-\; \int_{\mathbf{x}\in R_C\setminus R_u} p_u(\mathbf{x})\mathrm{d}\mathbf{x}\right) \\ = \frac{1}{n}\sum_{u=1}^{n}(0-0) = 0$$

which proves condition (2).

To prove condition (3), it can be straightforwardly noted that, due to the definition of prototype region (Equation (8.1)), $\forall \mathbf{x} \in R_C$, there must exist at least one p_u, $u \in [1..n]$, such that $p_u(\mathbf{x}) > 0$. Consequently, the following holds: $(1/n)\sum_{u=1}^{n} p_u(\mathbf{x}) > 0,\ \forall \mathbf{x} \in R_C$, i.e.,

$$p_C(\mathbf{x}) > 0,\ \forall \mathbf{x} \in R_C \tag{8.5}$$

which proves condition (3).

The following propositions are introduced to describe what is the form of a multivariate/univariate uncertain prototype for a new cluster resulting

from the union of any two given clusters. Note that the proof of each of these propositions is trivial since they can be derived straightforwardly from Definition 8.1 and Definition 8.2, respectively.

Proposition 8.4. *Given two disjoint sets C_i, C_j of M-dimensional multivariate uncertain objects and the corresponding prototypes $P_{C_i} = (R_{C_i}, p_{C_i})$ and $P_{C_j} = (R_{C_j}, p_{C_j})$, let $\tilde{C} = C_i \cup C_j$ be the set composed by the objects in C_i and C_j, and $C_{\tilde{C}} = (R_{\tilde{C}}, p_{\tilde{C}})$ the resulting prototype. It holds that:*

$$R_{\tilde{C}} = \Big[\min_{u\in\{i,j\}} L_u^{(1)}, \max_{u\in\{i,j\}} U_u^{(1)}\Big] \times \cdots \times \Big[\min_{u\in\{i,j\}} L_u^{(M)}, \max_{u\in\{i,j\}} U_u^{(M)}\Big] \tag{8.6}$$

$$p_{\tilde{C}} = \frac{|C_i|}{|\tilde{C}|} p_{C_i} + \frac{|C_j|}{|\tilde{C}|} p_{C_j} \tag{8.7}$$

Proposition 8.5. *Given two disjoint sets C_i, C_j of M-dimensional univariate uncertain objects and the corresponding prototypes $P_{C_i} = ((I_{C_i}^{(1)}, p_{C_i}^{(1)}), \ldots, (I_{C_i}^{(M)}, p_{C_i}^{(M)}))$ and $P_{C_j} = ((I_{C_j}^{(1)}, p_{C_j}^{(1)}), \ldots, (I_{C_j}^{(M)}, p_{C_j}^{(M)}))$, let $\tilde{C} = C_i \cup C_j$ be the set composed by the objects in C_i and C_j, and $P_{\tilde{C}} = ((I_{\tilde{C}}^{(1)}, p_{\tilde{C}}^{(1)}), \ldots, (I_{\tilde{C}}^{(M)}, p_{\tilde{C}}^{(M)}))$ the resulting prototype. For each $h \in [1..M]$, it holds that:*

$$I_{\tilde{C}}^{(h)} = \Big[\min_{u\in\{i,j\}} L_u^{(h)}, \max_{u\in\{i,j\}} U_u^{(h)}\Big] \tag{8.8}$$

$$p_{\tilde{C}}^{(h)} = \frac{|C_i|}{|\tilde{C}|} p_{C_i}^{(h)} + \frac{|C_j|}{|\tilde{C}|} p_{C_j}^{(h)} \tag{8.9}$$

8.3 Computing Distance between Uncertain Objects

8.3.1 Distance Measure for pdfs

Probability density functions are usually compared by using *information-theoretic* (IT) measures, such as those falling into the Ali-Silvey class of distance functions [AS66]. These functions have been widely used in several application contexts, such as signal processing, pattern recognition, and speech recognition [AJ56, Bas89].

Two of the most frequently used distance measures for probability densities are the *Kullback-Leibler* (KL) divergence [KL51, Kul59] and the *Chernoff* distance [Che52]. From a similarity viewpoint, a very useful notion is represented by the *Bhattacharyya coefficient* [Bha43, Kai67]. Given any two continuous pdfs p and $p\prime$, the Bhattacharyya coefficient (ρ) is defined as:

$$\rho(p, p\prime) = \int_{\mathbf{x}\in\Re^M} \sqrt{p(\mathbf{x})\; p\prime(\mathbf{x})}\; \mathrm{d}\mathbf{x} \tag{8.10}$$

The original "discrete" definition of ρ in [Bha43] considers p and $p\prime$ as multinomial populations, each one consisting of k classes with associated probabilities. In any case, ρ has a relevant geometric interpretation: it can be seen as the cosine between the two vectors for p and $p\prime$, whose components are the square root of the probabilities of the k classes that compose p and $p\prime$. This interpretation also holds in the extended definition reported in Equation (8.10), which defines Bhattacharyya coefficient for continuous pdfs.

Based on the Bhattacharyya coefficient, various distance functions can in principle be defined [Kai67]. In particular, the following measure

$$\mathrm{B}(p, p\prime) = \sqrt{1 - \rho(p, p\prime)} \tag{8.11}$$

has a number of advantages with respect to other Bhattacharyya distances, such as the commonly used definition of the form $-\log \rho$, and other information-theoretic measures, including as Kullback-Leibler or Chernoff. Unlike all the other mentioned measures, B ranges within the interval [0,1], which makes it particularly suitable to be combined with measures that capture other aspects when comparing two pdfs. Also, unlike the Chernoff distance (which is a more general case), B is easier and less expensive to compute and satisfies the additive property for probability distributions even though the random variables are not identically distributed. Such a property states that the distance between two joint distributions of statistically independent random variables equals the sum of the marginal distances. Finally, B is symmetric (unlike Kullback-Leibler) and satisfies the triangle inequality (unlike $-\log \rho$, Kullback-Leibler, and Chernoff).

Combining Information-Theoretic Notions and Expected Value on pdfs

Using an IT proximity measure represents a natural solution for devising a notion of distance between uncertain objects; in particular, this choice is essential to establish a function that is able to compare two pdfs by exploiting the whole information stored in the pdfs. However, this holds provided that the comparison makes sense: indeed, it should be taken into account that IT measures work out for pdfs that share a common event space (domain region). By contrast, if the two pdfs do not have any intersection in their event spaces (i.e., there is no common region in which both pdfs are greater than zero), the distance according to any IT measure is always equal to the maximum value possible (e.g., one for B, ∞ for KL).

We introduce a notion, called *IT-adequacy*, which quantifies to what degree an information-theoretic distance measure is worth comparing two uncertain objects by involving only their pdfs.

Definition 8.6 (IT-adequacy). *Let p and $p\prime$ be two M-dimensional pdfs ($M \geq 1$), and $R_p \subseteq \Re^M$, $R_{p\prime} \subseteq \Re^M$ be two M-dimensional regions such that:*

$$\int_{\mathbf{x}\in\Re^M\setminus R_p} p(\mathbf{x})\mathrm{d}\mathbf{x} = 0 \quad \textit{and} \quad p(\mathbf{x}) > 0 \ \ , \ \forall \mathbf{x} \in R_p,$$

and

$$\int_{\mathbf{x}\in\Re^M\setminus R_{p\prime}} p\prime(\mathbf{x})\mathrm{d}\mathbf{x} = 0 \quad \textit{and} \quad p\prime(\mathbf{x}) > 0 \ \ , \ \forall \mathbf{x} \in R_{p\prime}.$$

The IT-adequacy *between p and $p\prime$ with respect to R_p and $R_{p\prime}$ is defined as:*

$$\Upsilon_{R_p,R_{p\prime}}(p,p\prime) = \frac{1}{2}\left(\int_{\mathbf{x}\in R_p\cap R_{p\prime}} p(\mathbf{x})\ \mathrm{d}\mathbf{x} \ + \int_{\mathbf{x}\in R_p\cap R_{p\prime}} p\prime(\mathbf{x})\ \mathrm{d}\mathbf{x}\right) \tag{8.12}$$

$\Upsilon_{R_p,R_{p\prime}}(p,p\prime)$, which ranges within $[0,1]$, expresses the adequacy of computing the distance between p and $p\prime$ by using a certain IT measure. In particular, the higher $\Upsilon_{R_p,R_{p\prime}}(p,p\prime)$, the more the information coming from p and $p\prime$ that is exploited in the comparison.

For the sake of simplicity of notation, we will use the symbols Υ (resp. $\Upsilon^{(h)}$) to denote the IT-adequacy relative to the comparison of any two multivariate (resp. univariate) uncertain objects.

Formally, for any two multivariate uncertain objects $o_u = (R_u, p_u)$ and $o_v = (R_v, p_v)$:

$$\Upsilon(o_u, o_v) = \Upsilon_{R_u,R_v}(p_u, p_v) \tag{8.13}$$

Analogously, if o_u and o_v are any two univariate uncertain objects, $o_u = ((I_u^{(1)}, p_u^{(1)}), \ldots, (I_u^{(M)}, p_u^{(M)}))$ and $o_v = ((I_v^{(1)}, p_v^{(1)}), \ldots, (I_v^{(M)}, p_v^{(M)}))$:

$$\Upsilon^{(h)}(o_u, o_v) = \Upsilon_{I_u^{(h)},I_v^{(h)}}(p_u^{(h)}, p_v^{(h)}) \tag{8.14}$$

It should be emphasized that a poor IT-adequacy may be computed when the pdfs of uncertain objects being compared have small (or empty) overlapping areas. To address such cases, it may be advisable to express the proximity between pdfs by resorting to the difference of their expected values. Within this view, the main intuition underlying our notion of distance measure between uncertain objects is to suitably combine an IT measure (which in principle is not always applicable) with a concise (but always available) information given by the expected values. Formally, we propose a distance measure Δ for uncertain objects o_u and o_v which is expressed as a function of two different terms:

$$\Delta(o_u, o_v) = \mathrm{f}(\Delta_{IT}(o_u, o_v), \Delta_{ED}(o_u, o_v)) \tag{8.15}$$

where Δ_{IT} involves a comparison by means of a certain IT measure, and Δ_{ED} measures the distance proportionally to the difference of the expected values.

8.3.2 Distance Measure for Uncertain Objects

We now provide the definitions of our distance measure for uncertain objects in both the multivariate and univariate case. More precisely, according to Equation (8.15), we specify the choices for (i) the IT measure used for computing Δ_{IT}, and (ii) the way of combining Δ_{IT} and Δ_{ED}.

As regards the first point, we chose the Bhattacharyya distance (B), as defined in Equation (8.11). In Section 8.3.1, we have already given motivations for which this measure has been preferred to other IT measures (such as, e.g., $-\log\rho$, Kullback-Leibler or Chernoff). We point out that B ranges within $[0, 1]$, which makes this measure easily comparable and combinable with other distance functions, which represents a major focus on this research. A further notable remark is that B can be proved to be strictly related to Υ (Definition 8.6). Indeed, B is based on ρ (the Bhattacharyya coefficient), for which the following nice property holds.

Proposition 8.7. *Let p and $p\prime$ be two M-dimensional pdfs ($M \geq 1$), and $R_p \subseteq \Re^M$, $R_{p\prime} \subseteq \Re^M$ be two M-dimensional regions such that:*

$$\int_{\mathbf{x}\in\Re^M\setminus R_p} p(\mathbf{x})\mathrm{d}\mathbf{x} = 0 \quad \textit{and} \quad p(\mathbf{x}) > 0 \ \ , \ \forall \mathbf{x} \in R_p,$$

and

$$\int_{\mathbf{x}\in\Re^M\setminus R_{p\prime}} p\prime(\mathbf{x})\mathrm{d}\mathbf{x} = 0 \quad \textit{and} \quad p\prime(\mathbf{x}) > 0 \ \ , \ \forall \mathbf{x} \in R_{p\prime}.$$

It holds that:

$$\rho(p, p\prime) \leq \Upsilon_{R_p, R_{p\prime}}(p, p\prime)$$

Proof. Since p and $p\prime$ have a limited region of definition (i.e., R_p and $R_{p\prime}$, respectively), the following statement holds:

$$\rho(p, p\prime) = \int_{\mathbf{x}\in\Re^M} \sqrt{p(\mathbf{x})\ p\prime(\mathbf{x})}\ \mathrm{d}\mathbf{x} \ = \int_{\mathbf{x}\in R_p\cap R_{p\prime}} \sqrt{p\prime(\mathbf{x})\ p\prime(\mathbf{x})}\ \mathrm{d}\mathbf{x}$$

Hence, to prove that $\rho(p, p\prime) \leq \Upsilon_{R_p, R_{p\prime}}(p, p\prime)$, it is sufficient to demonstrate that:

$$\int_{\mathbf{x}\in R_p\cap R_{p\prime}} \sqrt{p(\mathbf{x})\ p\prime(\mathbf{x})}\ \mathrm{d}\mathbf{x} \leq \int_{\mathbf{x}\in R_p\cap R_{p\prime}} \frac{1}{2}\Big(p(\mathbf{x}) + p\prime(\mathbf{x})\Big)\ \mathrm{d}\mathbf{x}$$

, which it is obviously to demonstrate since it derives from the arithmetic mean-geometric mean (AM-GM) inequality.

Proposition 8.7 shows that the upper bound of the computation of ρ for any two given pdfs is equal to the IT-adequacy between those pdfs. This represents a key aspect in our framework since it supports the theoretical validity for combining Δ_{IT} and Δ_{ED}.

Once established the IT distance measure to be used in Δ_{IT}, we have to define a way to properly combine Δ_{IT} and Δ_{ED}. A first way is to introduce a weighting factor ranging within $[0, 1]$ to control the contribution due to B with respect to the difference between the expected values, and define Δ as a linear combination of Δ_{IT} and Δ_{ED}. This weighting factor can be defined proportionally to the width of the domain region shared between the pdfs of the uncertain objects being compared.[1]

The way of combining Δ_{IT} and Δ_{ED} described above may represent a reasonable choice to a certain extent, since the larger is the portion of the pdfs involved into the Bhattacharyya distance computation, the smaller is the need for comparing the pdfs by also considering the corresponding expected values. However, it may happen the case in which two pdfs share a large domain region but have most of their values (main portions of their areas) over different intervals.

Within this view, a more robust way to weight the contribution due to B might be based on the degree of overlap of the pdf areas. As previously stated, this property is at the basis of the notion of IT-adequacy, which also represents an upper bound for ρ. For this purpose, we control the overall (linear) combination of Δ_{IT} and Δ_{ED} by a factor that is proportional to the IT-adequacy of the pdfs of the objects to be compared.

Definition 8.8 (multivariate uncertain distance). *The* multivariate uncertain distance *between two multivariate uncertain objects* $o_u = (R_u, p_u)$ *and* $o_v = (R_v, p_v)$ *is defined as*

$$\Delta(o_u, o_v) = \mathrm{B}(p_u, p_v) - \sqrt{1 - \Upsilon(o_u, o_v)}\; e^{-dist(E[p_u], E[p_v])} \tag{8.16}$$

In Equation (8.16), $dist : \Re^M \rightarrow \Re_0^+$ is a function that measures the distance between M-dimensional points (e.g., Euclidean norm), and $E[p_\cdot]$ denotes the expected value of the pdf $p_\cdot$. Note that the exponential function is used to make the distance between expected values ranging within $[0, 1]$.

Definition 8.9 (univariate uncertain distance). *The* univariate uncertain distance *between two univariate uncertain objects* $o_u = ((I_u^{(1)}, p_u^{(1)}), \ldots, (I_u^{(M)}, p_u^{(M)}))$ *and* $o_v = ((I_v^{(1)}, p_v^{(1)}), \ldots, (I_v^{(M)}, p_v^{(M)}))$ *is defined as*

$$\Delta(o_u, o_v) = f'(\delta^{(1)}, \ldots, \delta^{(M)}) \tag{8.17}$$

[1] In our earlier work [GPTG08], we used the parameter γ, which was defined in terms of the domain regions of the uncertain objects to be compared. Specifically, γ was defined as equal to the ratio of the width of the shared region to the width of the union of the two regions.

where

$$\delta^{(h)} = \mathrm{B}(p_u^{(h)}, p_v^{(h)}) - \sqrt{1 - \Upsilon^{(h)}(o_u, o_v)}\; e^{-dist(E[p_u^{(h)}] - E[p_v^{(h)}])}$$

for each $h \in [1..M]$.

In Equation (8.17), $f' : \Re^M \rightarrow \Re_0^+$ is a function that computes a scalar value from the components of an M-dimensional vector. In the following, we assume $f'(\delta^{(1)}, \ldots, \delta^{(M)}) = (1/M)\sqrt{\sum_{h=1}^{M} (\delta^{(h)})^2}$.

Proposition 8.10. *Given any two uncertain objects* o_u *and* o_v, $\Delta(o_u, o_v)$ *assumes values within* $[0, 1]$.

Proof.

$$\begin{aligned}
\Delta(o_u, o_v) &= \mathrm{B}(p_u, p_v) - \sqrt{1 - \Upsilon(o_u, o_v)}\; e^{-dist(E[p_u], E[p_v])} \\
&= \sqrt{1 - \rho(p_u, p_v)} - \sqrt{1 - \Upsilon(o_u, o_v)}\; e^{-dist(E[p_u], E[p_v])} \\
&= 1 - [\Delta'_{uv} + \Delta''_{uv}]
\end{aligned}$$

where

$$\Delta'_{uv} = 1 - \sqrt{1 - \rho(p_u, p_v)} \quad \text{and}$$

$$\Delta''_{uv} = \left(1 - \left(1 - \sqrt{1 - \Upsilon(o_u, o_v)}\right)\right) e^{-dist(E[p_u], E[p_v])}$$

Since $\rho(p_u, p_v) \leq \Upsilon(o_u, o_v)$ (Proposition 8.7), we have that $\Delta'_{uv} \leq 1 - \sqrt{1 - \Upsilon(o_u, o_v)}$, consequently:

$$\begin{aligned}
\min \Delta(o_u, o_v) &= 1 - [\max \Delta'_{uv} + \max \Delta''_{uv}] \\
&= 1 - \left[1 - \sqrt{1 - \Upsilon(o_u, o_v)} + \left(1 - \left(1 - \sqrt{1 - \Upsilon(o_u, o_v)}\right)\right)\right] \\
&= 0
\end{aligned}$$

Moreover, since Δ'_{uv} and Δ''_{uv} are minimum if and only if $\rho(f_u, f_v)$ and $e^{-dist(E[p_u], E[p_v])}$ are individually minimum (i.e., $\rho(p_u, p_v) = 0$ and $dist(E[p_u], E[p_v]) \rightarrow +\infty$), then:

$$\begin{aligned}
\max \Delta(o_u, o_v) &= 1 - [\min \Delta'_{uv} + \min \Delta''_{uv}] \\
&= 1 - \left[0 + \sqrt{1 - \Upsilon(o_u, o_v)} e^{-\infty}\right] \\
&= 1
\end{aligned}$$

We now give an insight into the behavior of the proposed distance function Δ, when the two terms Δ_{IT} and Δ_{ED} are close to their extreme values, i.e., they are close to either zero or one. For the sake of brevity, we show a number of properties only for the multivariate definition of Δ, since it is easy to prove that they also hold for each $\delta^{(h)}$ term ($h \in [1..M]$) of the univariate version of Δ.

1. Since $\mathrm{B}(p_u, p_v) = 0$ if and only if $o_u = o_v$, and $\mathrm{B}(p_u, p_v) = 0$ implies that $\Upsilon(o_u, o_v) = 1$, we have:
$$\lim_{\mathrm{B}(p_u,p_v)\to 0} \Delta(o_u, o_v) = 0 \tag{8.18}$$
Indeed, if $o_u = o_v$ then $\Delta(o_u, o_v)$ is equal to zero.
2. Let us denote $\Delta^{(2)} = 1 - \sqrt{1 - \Upsilon(o_u, o_v)} \times e^{-dist(E[p_u],E[p_v])}$. It is easy to see that:
$$\lim_{\mathrm{B}(p_u,p_v)\to 1} \Delta(o_u, o_v) = \Delta^{(2)} \tag{8.19}$$
To prove the soundness of this property, it should be noted that:
$$\lim_{\Upsilon(o_u,o_v)\to 0} \Delta^{(2)} = 1 - e^{-dist(E[p_u],E[p_v])}$$
i.e., if the IT-adequacy between o_u and o_v is low, the only information exploited for computing Δ is the distance between $E[p_u]$ and $E[p_v]$, which is clearly reasonable. Moreover:
$$\lim_{\Upsilon(o_u,o_v)\to 1} \Delta^{(2)} = 1$$
which is intuitively correct since, if $\Upsilon(o_u, o_v)$ is high, Δ would tend to the distance value computed by B, which tends to one in this case.
3. Let $\Delta^{(3)} = \mathrm{B}(p_u, p_v) - \sqrt{1 - \Upsilon(o_u, o_v)}$. It holds that:
$$\lim_{dist(E[p_u],E[p_v])\to 0} \Delta(o_u, o_v) = \Delta^{(3)} \tag{8.20}$$
This property holds due to the following additional considerations.
$$\lim_{\Upsilon(o_u,o_v)\to 0} \Delta^{(3)} = 0$$
Indeed, if o_u and o_v have low IT-adequacy, Δ would take into account only $dist(E[p_u], E[p_v])$, which tends to 0 in this case. Also:
$$\lim_{\Upsilon(o_u,o_v)\to 1} \Delta^{(3)} = \mathrm{B}(p_u, p_v)$$
In this case, since the IT-adequacy is high, it is correct to focus on the shapes of the pdfs, i.e., to compute Δ by only considering the term $\mathrm{B}(p_u, p_v)$;

Algorithm 7 U-AHC

Input: a set of uncertain objects $\mathcal{D} = \{o_1, \ldots, o_N\}$;
the number S of samples used for integral estimations
Output: a set of partitions $\mathbf{C}$
$\{initialization\}$
$\mathcal{C} \leftarrow \emptyset$, $\mathbf{Q} \leftarrow \emptyset$
for all $o \in \mathcal{C}$ **do**
  object_sampling(o, S)
  $C \leftarrow \{o\}$, $P_C \leftarrow o$
  $\mathcal{C} \leftarrow \mathcal{C} \cup \{C\}$
end for
$\mathbf{C} \leftarrow \{\mathcal{C}\}$
for all $C_i, C_j \in \mathcal{C}, C_i \neq C_j$ **do**
  $d \leftarrow$ **prototype_distance**$(C_i, C_j, P_{C_i}, P_{C_j})$
  $\mathbf{Q}.insert(C_i, C_j, d)$
end for
$\{main\ loop\}$
repeat
  $(C_i, C_j) \leftarrow \mathbf{Q}.removeMin()$
  $\tilde{C} \leftarrow C_i \cup C_j$
  $P_{\tilde{C}} \leftarrow$ compute_prototype$(C_i, C_j, P_{C_i}, P_{C_j})$ $\{Eqs.\ 8.6\text{–}8.9\}$
  for all $C \in \mathcal{C}, C \neq C_i, C \neq C_j$ **do**
    $\mathbf{Q}.remove(C, C_i)$
    $\mathbf{Q}.remove(C, C_j)$
    $d \leftarrow$ **prototype_distance**$(C, \tilde{C}, P_C, P_{\tilde{C}})$
    $\mathbf{Q}.insert(C, \tilde{C}, d)$
  end for
  $\mathcal{C} \leftarrow \mathcal{C} \setminus \{C_i, C_j\} \cup \{\tilde{C}\}$
  $\mathbf{C} \leftarrow \mathbf{C} \cup \{\mathcal{C}\}$
until $|\mathcal{C}| = 1$

4. Finally:

$$\lim_{dist(E[p_u], E[p_v]) \to +\infty} \Delta(o_u, o_v) = \mathrm{B}(f_u, f_v) \tag{8.21}$$

which holds since o_u and o_v may be similar to a certain degree (in a way inversely proportional to $\mathrm{B}(p_u, p_v)$) even if the expected values of the corresponding pdfs p_u and p_v are dissimilar. This highlights as Δ allows for overcoming limitations of the distance between expected values and further supports for the robustness of Δ.

8.4 U-AHC Algorithm

Here we present our AHC-based algorithm for clustering uncertain objects, named *U-AHC*. The outline of U-AHC is given in Algorithm 7.

The input for U-AHC algorithm is a dataset $\mathcal{D}$ of N uncertain objects and a positive integer S, which denotes the number of pdf samples used for integral estimations; the output is a hierarchy of clusters $\mathbf{C}$. The algorithm follows the classic AHC scheme. The proposed implementation makes use of a priority queue ($\mathbf{Q}$) to store the inter-cluster distances in order to improve the computational efficiency—the lower the distance between a pair of clusters, the higher the corresponding priority in $\mathbf{Q}$.

The initialization steps (Lines 1-11) are in charge of computing the list of samples for each object (procedure *object_sampling*) and the initial set of clusters composed by N singletons along with the corresponding prototypes (Lines 2-6). The initial pair-wise distances are computed by the *prototype_distance* procedure (Function 8) and inserted into the priority queue (Lines 8-11). Note that, in order to estimate the integrals numerically, each uncertain object (including the uncertain prototypes) is associated with its corresponding list of samples during the whole execution of the algorithm (procedure *get_samples*).

The main loop of the algorithm (Lines 12-24) is repeated until the whole hierarchy has been built, i.e., the number of clusters in the current clustering $\mathcal{C}$ is equal to one. At each iteration, the pair of the two clusters having the minimum distance is extracted from the priority queue (Line 13) and the two clusters are merged to form a new cluster $\tilde{C}$ (Line 14). Next, the procedure *compute_prototype* (Line 15) updates the prototype of $\tilde{C}$ by computing the new region/intervals of definition (Equations (8.6)-(8.8)), pdfs (Equations (8.7)-(8.9)) and samples. More precisely, computing the new list of samples is accomplished by involving the *Markov Chain Monte Carlo* (MCMC) sampling method [Has70] which is suitable for mixture densities. Once the merging step has been completed, the priority queue is updated (Lines 16-21). For each cluster C in the current clustering, the distances between C and the earlier clusters C_i, C_j are removed from $\mathbf{Q}$ (Lines 17-18) and replaced with the distance from the new cluster $\tilde{C}$ (Lines 19-20). We assume that any removal operation $\mathbf{Q}.remove(C_i, C_j)$ may implicitly perform also $\mathbf{Q}.remove(C_j, C_i)$.

The *prototype_distance* function (Function 8) computes a distance value d between the prototypes of any two given clusters C_i and C_j according to either Definition 8.8 or Definition 8.9. This function consists of three main phases. First, a set U_B is computed to contain the upper-bound values of the Bhattacharyya coefficient (i.e., the IT-adequacy values) between the pdfs of the C_i's prototype and the C_j's prototype (Line 1); these values are calculated according to Equation (8.12), and involve the cumulative distribution functions (cdfs) of the objects within the clusters C_i and C_j. Next, the *compute_B_values* procedure computes the set D_{IT} of Bhattacharyya distance values according to Equation (8.11) (Line 14); this procedure estimates the integrals needed for deriving the Bhattacharyya coefficient by taking into account the set of samples $\mathcal{S}$ and the set $\mathcal{P}$ of "new" probabilities (Lines 3-13). The sets $\mathcal{P}$ and $\mathcal{S}$ are created as follows: the set $\mathcal{S}$ is built up with the samples of the C_i's prototype (resp. C_j's prototype) if $|C_i|$ is smaller than (resp. greater than or equal to) $|C_j|$, and $\mathcal{P}$ contains the probabilities of each sample in $\mathcal{S}$

Function 8 prototype_distance

Parameters: a pair of clusters C_i, C_j and their respective prototypes P_{C_i}, P_{C_j}
Returned data: a real value d within $[0, 1]$

1: $U_B \leftarrow$ compute_ub_values$(C_i, C_j, P_{C_i}, P_{C_j})$ {*Eq. 8.12* }
2: $\mathcal{P} \leftarrow \emptyset, \mathcal{S} \leftarrow \emptyset$
3: **if** $|C_i| < |C_j|$ **then**
4: $\quad \mathcal{S} \leftarrow$ get_samples(P_{C_i})
5: $\quad$ **for all** $s \in \mathcal{S}$ **do**
6: $\quad\quad \mathcal{P} \leftarrow \mathcal{P} \cup$ compute_probability(C_j, s)
7: $\quad$ **end for**
8: **else**
9: $\quad \mathcal{S} \leftarrow$ get_samples(P_{C_j})
10: $\quad$ **for all** $s \in \mathcal{S}$ **do**
11: $\quad\quad \mathcal{P} \leftarrow \mathcal{P} \cup$ compute_probability(C_i, s)
12: $\quad$ **end for**
13: **end if**
14: $D_{IT} \leftarrow$ compute_B_values$(P_{C_i}, P_{C_j}, \mathcal{S}, \mathcal{P})$ {*Eq. 8.11*}
15: $D_{ED} \leftarrow$ compute_ED_values$(P_{C_i}, \mathcal{P}_{C_j})$ {*Eqs. 8.16–8.17*}
16: $d \leftarrow$ uncertain_distance(D_{IT}, D_{ED}, U_B) {*Eqs. 8.16–8.17*}
17: **return** d

computed according to the pdfs of the objects in C_j (resp. C_i). In the third phase, the *compute_ED_values* procedure computes the set D_{ED} of distances based on the expected values; in particular, D_{ED} is populated by computing the exponential of the negative of the distances between the expected values of the prototypes (Line 15). Finally, the *uncertain_distance* procedure computes the final distance value d according to either Equation (8.16) (multivariate case) or Equation (8.17) (univariate case), by considering the values in the sets D_{IT}, D_{ED} and U_B (Line 16). Note that $|U_B| = |D_{IT}| = |D_{ED}| = 1$, in the multivariate case, whereas $|U_B| = |D_{IT}| = |D_{ED}| = M$, in the univariate case.

Computational aspects

Let us now discuss the computational complexity of Algorithm 7, given a dataset $\mathcal{D}$ of N uncertain objects and a number S of samples. As previously discussed, Algorithm 7 uses the *prototype_distance* function (Function 8), whose complexity can be trivially proved to be $\mathcal{O}(S \times \min\{|C_i|, |C_j|\})$. Furthermore, we assume that (i) the operations of insertion/deletion/extraction of any object into/from the priority queue $\mathbf{Q}$ of size Z are performed in $\mathcal{O}(\log Z)$, and (ii) the number M of dimensions of the uncertain objects is a constant. The costs of the various steps of Algorithm 7 are summarized as follows:

- the initialization steps, i.e., computing the lists of samples for each uncertain object and the set of initial clusters (Lines 2-6), and initializing the

priority queue (Lines 8-11), are executed in $\mathcal{O}(S \times N)$ and $\mathcal{O}(N^2 \times \log N)$, respectively;

- the main loop (Lines 12-24) is repeated $N-1$ times; therefore, each step of this loop has the following total cost:
  - extracting from $\mathbf{Q}$ the pairs of clusters having the minimum distance (Line 13) is $\mathcal{O}(N \times \log N)$;
  - merging the clusters C_i and C_j (Line 14) is $\mathcal{O}(\sum_{k=1}^{N-1} \max_{C \in \mathcal{C}^{(k)}} |\mathcal{C}|) = \mathcal{O}(\sum_{k=1}^{N-1} k) = \mathcal{O}(N^2)$, where $\mathbf{C}^{(k)}$ is the set of clusters computed at the k-th iteration;
  - in the *compute_prototype* function, the most expensive operation is resampling from the new mixture density (i.e., the new prototype), which is $\mathcal{O}(S \times (|C_i| + |C_j|))$. Therefore, computing a new prototype for all the iterations of the algorithm (Line 15) is $\mathcal{O}(\sum_{k=1}^{N-1} S \times \max_{C \in \mathcal{C}^{(k)}} |C|) = \mathcal{O}(S \sum_{k=1}^{N-1} k) = \mathcal{O}(S \times N^2)$;
  - in the internal loop (Lines 16-21), the operations of insertion/deletion into/from the priority queue (Lines 17, 18, and 20) are performed in $\mathcal{O}(N^2 \times \log N)$. The prototype distance (Line 19) is computed by taking into account the list of samples of one of the two clusters, which has a cost $\mathcal{O}(\sum_{k=1}^{N-1} \sum_{C \in \mathcal{C}^{(k)}} S \times |C|) = \mathcal{O}(S \times \sum_{k=1}^{N-1} \sum_{C \in \mathcal{C}^{(k)}} |C|) = \mathcal{O}(S \times N^2)$;
- updating $\mathcal{C}$ (Line 22) and $\mathbf{C}$ (Line 23) is obviously $\mathcal{O}(N)$.

In conclusion, we can state that Algorithm 7 works in $\mathcal{O}(N^2(S + \log N))$. For many real cases, the condition that S is $\Omega(\log N)$ typically holds; under this assumption, the computational complexity of Algorithm 7 can be rewritten as $\mathcal{O}(S \times N^2)$.

Impact of IT-adequacy on the behavior of U-AHC

As discussed in Section 8.3.2, a crucial point of the proposed compound distance measure Δ is that, for any two given uncertain objects o_i and o_j having low IT-adequacy, $\Delta(o_u, o_v)$ tends to be much more weighted by the distance between the expected values. Indeed, the high accuracy in comparing pdfs usually assured by IT measures cannot be guaranteed when the corresponding IT-adequacy is low.

In this section, we give a theoretical justification of the impact of IT-adequacy on the distance computation for cluster prototypes defined as mixture densities in a centroid-based AHC algorithm, like U-AHC. We show the results only in the multivariate case, since an analogous reasoning also holds in the univariate case.

Let us first provide the following background notations—to avoid overloading the notation, we shall assume that clustering and related items refer to the k-th iteration of the U-AHC algorithm, with $k \in [1..N-1]$.

Lemma 8.11. *Let $o = (R, p)$ be a multivariate uncertain object. Given any $R' \subset \Re^M$, it holds that:*

$$\int_{R \cap R'} p(\mathbf{x}) \mathrm{d}\mathbf{x} = \int_{R'} p(\mathbf{x}) \mathrm{d}\mathbf{x}$$

Proof. To prove the lemma, it is easy to see that:

$$\int_{R \cap R'} p(\mathbf{x}) \mathrm{d}\mathbf{x} = \int_{R'} p(\mathbf{x}) \mathrm{d}\mathbf{x} - \int_{R' \setminus R} p(\mathbf{x}) \mathrm{d}\mathbf{x} = \int_{R'} p(\mathbf{x}) \mathrm{d}\mathbf{x}$$

If $R' \subset R$, then the above equality holds trivially, since $R \cap R' = R'$ and p is also defined over R'. Otherwise, if $R \subset R \cup R'$, it holds that

$$\int_{R' \setminus R} p(\mathbf{x}) \mathrm{d}\mathbf{x} = 0$$

Finally, as a special case, if $R' \cap R = \emptyset$ then the integral of p over the empty set is equal to zero as well as the integral of p over any external region R'.

- $\mathbf{P} = \{P_1, \ldots, P_{N-k+1}\}$ is a set of prototypes of the form $P_q = (R_q, p_q)$, for $q \in [1..N-k+1]$, which correspond to the clustering solution $\mathcal{C} = \{C_1, \ldots, C_{N-k+1}\}$ obtained by U-AHC (at the k-th iteration);
- $i, j \in [1..N-k+1]$ are two indices such that $C_i, C_j \in \mathcal{C}$ are the pair of clusters to be merged, and
- $\tilde{C} = C_i \cup C_j$ is the new cluster formed;
- $\tilde{P} = (\tilde{R}, \tilde{p})$ is the prototype of $\tilde{P}$.

Theorem 8.12. *Let $\alpha \in [0, 1]$ be a constant, $\Psi_q(\mathcal{C}, \alpha) = \Upsilon(P_q, \tilde{P}) - (\alpha \Upsilon(P_q, P_i) + (1-\alpha)\Upsilon(P_q, P_j))$. Given $\hat{\alpha} = |C_i|/(|C_i| + |C_j|)$, for each $q \in [1..N-k+1]$, $q \neq i$, $q \neq j$, it holds that:*

$$\Psi_q(\mathcal{C}, \hat{\alpha}) = \frac{1}{2}\left((1-\hat{\alpha}) \int_{R_i} p_u(\mathbf{x}) \mathrm{d}\mathbf{x} + \hat{\alpha} \int_{R_j} p_u(\mathbf{x}) \mathrm{d}\mathbf{x}\right)$$

Proof. Let us denote with $\mathcal{I}_\phi(\Phi)$ an integral of the form $\int_\Phi p_\phi(\mathbf{x}) \mathrm{d}\mathbf{x}$. By setting α equal to $\hat{\alpha}$, we have that:

$$\begin{aligned}
\Psi_q(\mathcal{C}, \hat{\alpha}) = \frac{1}{2}\left(\mathcal{I}_q(R_q \cap \tilde{R}) + \tilde{\mathcal{I}}(R_q \cap \tilde{R})\right) + \\
- \left[\frac{\hat{\alpha}}{2}\left(\mathcal{I}_q(R_q \cap R_i) + \mathcal{I}_i(R_q \cap R_i)\right) + \right. \\
\left. + \frac{1-\hat{\alpha}}{2}\left(\mathcal{I}_q(R_q \cap R_j) + \mathcal{I}_j(R_q \cap R_j)\right)\right]
\end{aligned}$$

Since for any multivariate uncertain object $o = (R, f)$ and any $R' \subset \Re^M$ it holds that (Lemma 8.11):

$$\int_{R \cap R'} p(\mathbf{x}) \mathrm{d}\mathbf{x} = \int_{R'} p(\mathbf{x}) \mathrm{d}\mathbf{x}$$

and $\tilde{p} = \hat{\alpha} p_i + (1 - \hat{\alpha}) p_j$ (Equation 8.7), then:

$$\begin{aligned}
\Psi_q(\mathcal{C}, \hat{\alpha}) &= \frac{1}{2}\Bigg(\mathcal{I}_q(R_i) + \mathcal{I}_q(R_j) + \hat{\alpha}\mathcal{I}_i(R_q) + \\
&\quad + (1 - \hat{\alpha})\mathcal{I}_j(R_q) \Bigg) - \Bigg[\frac{\hat{\alpha}}{2}\Big(\mathcal{I}_q(R_i) + \mathcal{I}_i(R_q) \Big) + \\
&\quad + \frac{1 - \hat{\alpha}}{2}\Big(\mathcal{I}_q(R_j) + \mathcal{I}_j(R_q) \Big) \Bigg] \\
&= \frac{1}{2}\Big((1 - \hat{\alpha})\mathcal{I}_q(R_i) + \hat{\alpha}\mathcal{I}_q(R_j) \Big) \\
&= \frac{1}{2}\Bigg((1 - \hat{\alpha}) \int_{R_i} p_q(\mathbf{x}) \mathrm{d}\mathbf{x} + \hat{\alpha} \int_{R_j} p_q(\mathbf{x}) \mathrm{d}\mathbf{x} \Bigg)
\end{aligned}$$

Corollary 8.13. *For each $q \in [1..n - k + 1]$, $q \neq i$, $q \neq j$, if $\Upsilon(P_q, P_i) \neq 0$ or $\Upsilon(P_q, P_j) \neq 0$, then $\Psi(\mathcal{C}, \hat{\alpha}) > 0$; otherwise $\Psi(\mathcal{C}, \hat{\alpha}) = 0$.*

The above theorem and corollary state that the notion of IT-adequacy plays an important role when comparing the (prototype of) clusters in every iteration of our hierarchical algorithm. In particular, given any two clusters C_i, C_j being merged and any other cluster C_q, there is a strong relation between the individual IT-adequacy of P_i and P_j with respect to P_q, and the IT-adequacy of the prototype of the (new) merged cluster with respect to P_q. More precisely, the IT-adequacy of the prototype of the (new) merged cluster to P_q is not lower than (the combination of) the individual IT-adequacy of P_i and P_j with respect to P_q. As a special case, if there is an overlap between the region of P_i (or P_j) and P_q, the IT-adequacy resulting from the merging step will increase.

8.5 Experimental Evaluation

Our U-AHC algorithm was evaluated in performing effective clustering of uncertain data. The experimental evaluation was also conducted to give a comparison of U-AHC with existing partitional algorithms (i.e., UK-means, CK-means, and UK-medoids) and density-based algorithms (i.e., $\mathcal{F}$DBSCAN and $\mathcal{F}$OPTICS).

In the following, we discuss the evaluation methodology, which includes a description of the datasets, the measures to assess the quality of the clustering

solutions, and the setups of the various clustering methods. In Section 8.5.2, we present the main experimental results from both accuracy and efficiency viewpoints.

8.5.1 Settings

Datasets

Experiments were performed on benchmark datasets. These datasets were originally established as collections of data with deterministic values.[2] We synthetically generated uncertainty in these collections according to the strategy adopted in Section 7.4.1.

We selected eight datasets with numerical real-valued attributes, namely Iris, Wine, Glass, Ecoli, Yeast, ImageSegmentation, Abalone, and LetterRecognition. Table 8.1 summarizes the main characteristics of these datasets.

Table 8.1. Datasets used in the experiments

dataset	*# of objects*	*# of attributes*	*# of classes*
Iris	150	4	3
Wine	178	13	3
Glass	214	10	6
Ecoli	327	7	5
Yeast	1,484	8	10
ImageSegmentation	2,310	19	7
Abalone	4,124	7	17
LetterRecognition	7,648	16	10

Iris, Wine, Glass, and Ecoli have been described while testing our UK-medoids. For dataset description see Section 7.4.1. Yeast objects describe the main features and the localization of various proteins. ImageSegmentation contains objects that were randomly drawn from a database of seven outdoor images; the images (3x3 regions) were hand-segmented to create a classification for each pixel. Abalone is about different types of abalone shells. LetterRecognition contains character images corresponding to the capital letters in the English alphabet; we selected the instances of each letter from A to J.

Clustering validity criteria

To assess the quality of clustering solutions for the benchmark datasets, we exploited the availability of reference classifications. The objective was to evaluate how well a clustering fits a predefined scheme of known classes (natural clusters). For this purpose, we resorted to the well-known F_1-measure (cf. Section 2.1.6).

[2] UCI Machine Learning Repository — http://archive.ics.uci.edu/ml/

Setup of the clustering methods

We computed the integrals involved into the distance calculation by taking into account lists of samples derived from the pdfs. For this purpose, we employed the classic *Monte Carlo* and *Markov Chain Monte Carlo* sampling methods.[3]

Computing the set of initial centroids (in K-means based methods) and computing the set of samples (for all methods) correspond to non-deterministic operations; therefore, to avoid that clustering results were biased by random chance, the accuracy and efficiency measurements for all methods were averaged over 100 different runs.

We performed a tuning phase for the parameters ϵ (i.e., the threshold for the distance of the neighbors of an object) and μ (i.e., the minimum number of points within the neighborhood of an object) required by the density-based approaches $\mathcal{F}$DBSCAN and $\mathcal{F}$OPTICS. We set these parameters to the values that allowed each method to achieve the best accuracy results.

For our U-AHC and $\mathcal{F}$OPTICS, we also needed for selecting a partition of proper size from the clustering solution obtained by each of these methods. In particular, we considered the partition obtained by cutting the dendrogram at the level corresponding to the desired number of output clusters (e.g., equal to the desired number of ideal classes for each benchmark dataset). For $\mathcal{F}$OPTICS, we initially derived a cluster hierarchy by converting the reachability plot into a dendrogram according to the automatic procedure described in [SQL$^+$03].

8.5.2 Results

Accuracy

Table 8.2 and Table 8.3 summarize the F_1-measure results obtained by U-AHC and the other methods on the various datasets for the univariate and multivariate models, respectively.

In either table, the last two rows contain the average F_1-measure score obtained by each method and the average percentage of quality gain obtained by U-AHC against each other method.

A first important remark is that U-AHC achieved the highest quality on all the datasets. On average, U-AHC outperformed the other methods, with average gains that ranged from 8.2% (vs. UK-medoids) to about 18.4% (vs. $\mathcal{F}$DBSCAN), in the univariate case, and from 10% (vs. UK-medoids) to 18% (vs. $\mathcal{F}$OPTICS), in the multivariate case.

Overall, we observe that all the clustering methods obtained their respective best performances according to the uniform distribution in nearly all datasets. Moreover, in general, the algorithm performances improved in the multivariate case.

[3] We used the SSJ library, available at http://www.iro.umontreal.ca/∼simardr/ssj/

Table 8.2. Accuracy results (F_1-measure) for univariate models

dataset	*pdf*	UK-means	CK-means	UK-medoids	$\mathcal{F}$DBSCAN	$\mathcal{F}$OPTICS	**U-AHC**
Iris	Uniform	0.841	*0.963*	0.886	0.919	0.886	**0.993**
	Normal	0.849	0.849	0.855	0.871	*0.907*	0.905
	Gamma	0.622	0.501	0.848	0.893	*0.905*	0.628
Wine	Uniform	0.500	0.724	*0.810*	0.664	0.695	**0.984**
	Normal	0.500	0.704	0.578	0.653	*0.713*	0.954
	Gamma	0.500	0.581	0.581	0.692	*0.713*	0.595
Glass	Uniform	0.639	0.670	0.697	*0.768*	0.718	**0.828**
	Normal	*0.577*	0.552	0.513	0.514	0.438	0.822
	Gamma	0.379	0.314	*0.644*	0.468	0.438	0.550
Ecoli	Uniform	0.653	*0.795*	0.696	0.436	0.477	**0.915**
	Normal	0.609	*0.741*	0.528	0.544	0.477	0.726
	Gamma	0.533	0.412	*0.693*	0.401	0.477	0.450
Yeast	Uniform	0.497	0.562	*0.618*	0.515	0.543	**0.719**
	Normal	*0.471*	0.458	0.288	0.291	0.316	0.577
	Gamma	0.403	0.306	*0.469*	0.331	0.316	0.406
Image Segm.	Uniform	*0.810*	0.798	0.769	0.426	0.419	0.552
	Normal	0.623	*0.655*	0.451	0.416	0.419	**0.836**
	Gamma	0.545	0.353	*0.656*	0.339	0.419	0.503
Abalone	Uniform	0.331	0.294	*0.590*	0.447	0.439	**0.719**
	Normal	*0.288*	0.217	0.265	0.136	0.209	0.577
	Gamma	0.360	0.200	0.313	0.565	*0.607*	0.406
Letter Recogn.	Uniform	0.529	0.629	*0.776*	0.344	0.318	**0.792**
	Normal	0.449	0.451	*0.490*	0.247	0.318	0.531
	Gamma	0.432	0.215	*0.584*	0.265	0.318	0.603
	avg. score	0.539	0.539	0.608	0.506	0.521	0.690
	avg. gain	15.1%	15.1%	8.2%	18.4%	16.9%	—

Among the competing methods, UK-medoids behaved better than the other ones—10 out of 24 times (in the univariate case) and 6 out of 16 times (in the multivariate case)—obtaining an average quality gain up to 10% and 8% in the univariate and multivariate cases, respectively. The partitional algorithms performed slightly better than the density-based algorithms in the univariate case, whereas this gap tended to increase in the multivariate case. Moreover, ignoring the results that refer to uniform pdfs, the density-based algorithms (especially $\mathcal{F}$OPTICS) performed as good as or better than the partitional ones. It should also be noted that $\mathcal{F}$OPTICS and $\mathcal{F}$DBSCAN behaved closely, as well as UK-means and CK-means. This was not surprising since the two couples of algorithms employ similar clustering schemes.

Efficiency

As regards as the efficiency evaluation, we were actually interesting in measuring the runtime performance of our U-AHC in relation to the other methods. For the sake of brevity of presentation, here we present results concerning the univariate case on the four largest benchmark datasets, namely Yeast, ImageSegmentation, Abalone, and LetterRecognition; however, the performance

Table 8.3. Accuracy results (F_1-measure) for multivariate models

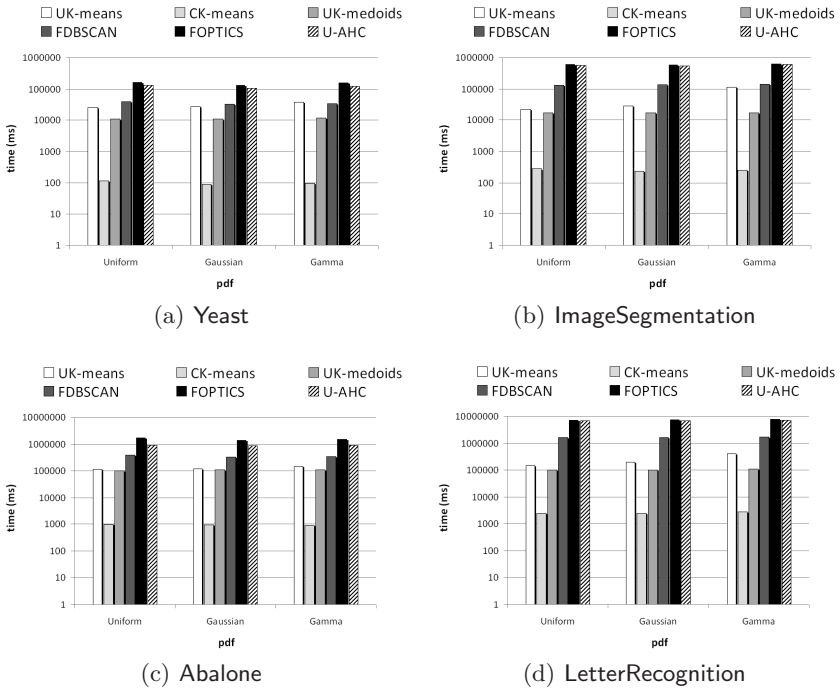
dataset	*pdf*	UK-means	CK-means	UK-medoids	$\mathcal{F}$DBSCAN	$\mathcal{F}$OPTICS	**U-AHC**
Iris	Uniform	0.948	*0.962*	0.907	0.929	0.907	**1**
	Normal	0.859	0.897	0.888	*0.929*	0.907	0.962
Wine	Uniform	0.735	0.747	0.761	*0.767*	0.713	**0.826**
	Normal	0.707	0.705	*0.749*	0.691	0.713	0.795
Glass	Uniform	0.677	*0.703*	0.653	0.575	0.636	0.779
	Normal	0.540	0.551	0.579	*0.868*	0.828	**0.891**
Ecoli	Uniform	0.787	*0.790*	0.728	0.443	0.477	0.743
	Normal	*0.745*	0.740	0.560	0.416	0.477	**0.795**
Yeast	Uniform	0.533	0.538	*0.622*	0.599	0.528	**0.684**
	Normal	0.455	*0.457*	0.318	0.374	0.420	0.486
Image	Uniform	0.780	*0.801*	0.765	0.482	0.419	**0.837**
Segm.	Normal	0.628	0.637	*0.649*	0.415	0.419	0.684
Abalone	Uniform	0.288	0.290	*0.531*	0.499	0.439	0.492
	Normal	0.215	0.217	0.288	0.497	*0.558*	**0.572**
Letter	Uniform	0.637	0.636	*0.763*	0.320	0.318	**0.798**
Recogn.	Normal	0.442	0.435	*0.595*	0.353	0.318	0.613
	avg. score	0.624	0.632	0.647	0.571	0.567	0.747
	avg. gain	12.3%	11.5%	10.0%	17.6%	18.0%	—

trends observed on both the rest of benchmark datasets and microarray datasets (including the multivariate case) were roughly similar to those presented in the following.[4] Note also that we left out of consideration times for operations which are usually carried out in an off-line mode, i.e., computing the list of samples for every uncertain object (required by all algorithms), the matrix of pair-wise distances (UK-medoids and U-AHC), the gravity centers (expected values), and the distances between each uncertain object and its gravity center (CK-means).

Figure 8.1 shows the clustering time performances (in milliseconds) of the various methods on the selected datasets. We can see that, as we expected, CK-means outperformed all the other methods, including UK-means; clearly, in this case, the significant gain observed is mainly due to the optimization employed by CK-means for the EDs calculation. On the other hand, it should be recalled that the CK-means algorithm works only if the mean squared error for the definition of the EDs is used and the distance function is based on the Euclidean norm.

UK-medoids was the second fastest method, even better than UK-means; we indeed observed that UK-medoids in general required less iterations for the convergence than UK-means. Our U-AHC compared closely to the density-based algorithms. More precisely, $\mathcal{F}$DBSCAN performed better than U-AHC, although in times of the same order of magnitude, and U-AHC as close as or slightly better than $\mathcal{F}$OPTICS in turn.

[4] Experiments were conducted on a platform Intel Pentium IV 3GHz with 2GB memory and running Microsoft WinXP Pro.



(a) Yeast (b) ImageSegmentation

(c) Abalone (d) LetterRecognition

Fig. 8.1. Clustering time performances in the univariate model

It is also easy to see how the plots in Figure 8.1 are very similar over the various datasets. In general, it should be noted that the clustering performances followed the (on-line) computational complexities of the corresponding algorithm, namely $\mathcal{O}(t \times S \times N)$ for UK-means, $\mathcal{O}(t \times N)$ for CK-means, $\mathcal{O}(t \times N^2)$ for UK-medoids — t is the number of iterations required for the algorithm convergence — $\mathcal{O}(N^2)$ for $\mathcal{F}$DBSCAN, and $\mathcal{O}(S \times N^2)$ for $\mathcal{F}$OPTICS and U-AHC.

8.6 U-AHC for Clustering Microarray Data

In the next sections, we define a way to apply our U-AHC clustering scheme in a real-case application, which consists in analyzing microarray data with probe-level uncertainty.

8.6.1 Microarray Data and Probe-level Uncertainty

A *DNA Microarray* is a technology based on multiplexer assay, which is able to produce multiple measurements simultaneously in a single experiment. It is composed by thousands of spots organized in arrays. Spots contain DNA oligonucleotides (i.e., features), each of which has picomoles of a specific DNA sequence, called *probes*. DNA microarray is widely used genomics fields, such as molecular biology, since it is able to produce lots of data in a single experiment [Dră03, BC09].

Handling microarray data is particularly challenging mainly due to the high dimensionality of such data, which demands for computer-aided methods, and to the intrinsic difficulty of devising notions of proximity between spots of array traps.

Many approaches to microarray data analysis have been proposed by the research community [Dră03]. Most of them are essentially based on data mining techniques, in particular *clustering* methods [GMW07, JD88]. Clustering allows for understanding the huge mass of data in microarrays by grouping them in homogeneous subsets (clusters). In this way, cluster analysis aims to discover natural structures within the data and to help the analyst in identifying common structures and patterns in microarrays; for instance, finding similar expression patterns (i.e., co-expressed genes) which are related to cellular functions.

Microarray clustering approaches can be divided into three main categories [JTZ04]: (i) *gene-based clustering*, which treats genes as objects and samples as clustering features; (ii) *sample-based clustering*, where samples are the objects to be clustered and genes are the features; (iii) *co-clustering* approaches, where genes and samples are treated symmetrically (samples and genes can be both objects and features).

After several years of quantitative measurements of microarray probe-level data, new models have been proposed in order to manage the uncertainty of gene expression levels both in a single chip and across multiple chips. A novel probabilistic modeling approach is presented in [LMLR05a], where the binding affinity of probe-pairs across multiple chips is modeled through a probabilistic model using Gamma distributions. In [LLAR07], a gene expression clustering algorithm has been proposed, which exploits the probabilistic modeling described above in order to improve performances with respect to classic techniques.

In this section, we propose a new approach to modeling probe-level uncertainty in microarray data. Uncertainty is modeled according to the strategies

proposed in Section 2.3.1, while the analysis approach follows the intuitions presented in Section 8.1—8.4. In particular, we use the clustering scheme proposed in Section 8.4 equipped with an information-theoretic-based distance measure (cf. Sections 8.3) between uncertain cluster prototypes. The adoption of cluster prototypes as mixture densities enables a notion of information-theoretic distance measure that exploits an advantageous characteristic of the cluster prototypes: the overlaps between the cluster prototypes' domain regions are generally larger than the overlaps between the individual objects' regions.

We have tested the U-AHC algorithm on four large microarray datasets, and evaluated performance in achieving effective clustering. Experimental results have shown that U-AHC achieves high results in terms of compactness of the clustering solution.

8.6.2 Experimental Evaluation

The U-AHC algorithm was evaluated in performing effective clustering of microarray data with a probe-level uncertainty. In this section, we first discuss the evaluation methodology, which includes a description of the datasets, the uncertainty modeling, and the measures to assess the quality of the clustering solutions. Then, we present preliminary experimental results.

Settings

Datasets

Experiments were performed on four large microarray datasets, each of which describes the expressions of thousands of genes in biological tissues, as shown in Table 8.4.

Table 8.4. Microarray datasets used in the experiments

dataset	*# of genes*	*# of attributes*
Leukaemia	22,690	21
Neuroblastoma	22,282	14
Myelodysplastic	22,277	25
Mouse	45,101	10

Three datasets, namely Leukaemia, Neuroblastoma and Myelodysplastic are cancer tissue data of humans,[5] while Mouse is about mouse tissues.[6] Leukaemia describes the transformation process of leukaemia stem cells initiated by MLL-AF9 fusion gene. Neuroblastoma contains expression-based screening results for neuroblastoma differentiation. In Myelodysplastic, somatic chromosomal deletions in cancer are measured by means of an RNA-mediated interference (RNAi)-based approach to discovery of the $5q^-$ disease gene, which is a subtype of the myelodysplastic syndrome characterized by a defect in erythroid differentiation. Mouse contains a transcription profiling of mouse cochlea Reissner's membrane (RM). This is grown as explants and treated with dexamethasone and then subject to RNA extraction to investigate gene expressions.

Uncertainty models

The probe-level uncertainty for microarray datasets was extracted by exploiting the multi-mgMOS method [LMLR05a].[7] For each dimension, the multi-mgMOS method yields a set of information that includes mean, standard deviation and principal percentiles (i.e., 5%, 25%, 50%, 75%, 95%).

The information outputted by multi-mgMOS was exploited to model uncertainty according to the univariate model (Section 2.3.1). In particular, the univariate pdfs of each uncertain object (i.e., each row in the microarray matrix) was built by employing two different methods:

- *Normal* method, where Normal pdfs were easily derived from a combination of mean values with standard deviations;
- *Percentiles-based* method, where suitable statistical models were involved to fit pdfs to percentiles [Sil86].

Clustering validity criteria

We performed a *gene-based clustering* in such a way that each group describes a particular macroscopic phenotype, such as cancer expressions or biological states [JTZ04]. Since there is no available reference classification for such data, we resorted to internal validity criteria based on the *cophenetic correlation coefficient* (cf. Section 2.1.6).

[5] Cancer Program dataset page of the Broad Institute of MIT and Harvard, available at http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi

[6] Microarray Data resource page of the European Bioinformatics Institute (EMBL-EBI), available at http://www.ebi.ac.uk/microarray-as/ae/browse.html

[7] We used the Bioconductor package PUMA (*Propagating Uncertainty in Microarray Analysis*), available at http://www.bioinf.manchester.ac.uk/resources/puma/

Results

Table 8.5 summarizes the quality results in terms of cophenetic correlation for each dataset and for each pdf.

Table 8.5. Accuracy results for univariate models

dataset	*pdf form*	*cophenetic value*
Leukaemia	Normal	0.76
	Percentiles-based	0.82
Neuroblastoma	Normal	0.67
	Percentiles-based	0.75
Myelodysplastic	Normal	0.80
	Percentiles-based	0.89
Mouse	Normal	0.84
	Percentiles-based	0.92

It can be noted that the U-AHC algorithm obtained good accuracy results on all the datasets, from 67% to 84% with Normal pdfs. Also, the uncertainty generation based on percentiles generally led to higher quality results than the previous case (about 8% on average); this improvement can be easily explained by the fact that percentiles provide a more refined representation of the uncertainty than the summarized information of mean value and standard deviation used for Normal pdf modeling.

Part IV

Document Clustering

9

Generative Models for Document Representation

Summary. In this chapter, we describe a framework for clustering documents according to their mixtures of topics. The proposed framework combines the expressiveness of generative models for document representation with a properly chosen information-theoretic distance measure to group documents by means of an agglomerative hierarchical clustering scheme. The clustering solution reflects an organization of the documents according to their set of topics, without exploiting any soft/fuzzy clustering method. Experimental results obtained on large, real-world collections of documents evidence the effectiveness of our approach in detecting non-overlapping clusters that contain documents sharing similar mixtures of topics.

9.1 Introduction

Document clustering is a major topic of research in data management and retrieval due to an ever increasing availability of textual sources in several application domains (e.g., news articles, scientific papers, legal documents, web pages). Document clustering methods traditionally fall into the category of *discriminative* (distance-based) approaches, as they require the computation of distance/similarity between documents to group them into clusters. Moreover, such methods usually exploit classic retrieval models which represent the document contents in a term (word) feature space, such as the popular "bag-of-words" model.

Most of the existing applications of document clustering have traditionally assumed to assign each document to a unique cluster. In this way, any cluster refers to a group of documents that discuss the same main topic (i.e., a concept which is somehow described by the most representative terms of the documents within the cluster). However, this assumption is not effective for a broad variety of text data, such as interdisciplinary documents. To address this issue, there have been various proposals which assign documents to more than one cluster, usually performing a soft/fuzzy scheme [ZK04, KDK03].

In recent years, a number of approaches to model document contents have been developed, and they are based on the idea that any document can be

represented as a *mixture* of probability distributions over its terms, and each component of the mixture refers to a *topic* [Hof01, BNJ03, ZG03, ZG05]. The document representation is hence obtained by a *generative process*, which expresses the document features as mixture models. In this way, generative model-based methods are able to provide a finer-grained modeling of documents, which is particularly useful to fit the case of interdisciplinary or multi-topic documents.

For document clustering purposes, there can be found various ways of inducing an organization of the documents according to the topic-space based representation offered by generative models. For instance, each document can be assigned to the most probable topic, or to a set of topics according to a given probability threshold. Both these strategies however incur evident drawbacks, as the first one is not able to support multi-topic assignment, whereas the second one may lead to overlay overlapping clusters.

In this chapter, we propose a framework for hard clustering of documents according to their mixtures of topics [PT09]. Documents are assumed to be modeled by a generative process, which provides a mixture of probability mass functions (pmfs) to model the topics that are discussed within any specific document. The framework combines the expressiveness of generative models for document representation with a properly chosen information-theoretic distance measure, which is defined to compute the distance between any two probability distributions over a topic feature space. The proposed clustering method refers to a centroid-linkage-based agglomerative hierarchical scheme. Moreover, it is "hard" in the sense that it produces a clustering solution for a given collection of documents in such a way that each document is assigned to a unique cluster and each cluster contains documents that share the same topic assignment.

The effectiveness of the proposed approach has been experimentally evaluated over large collections of documents. A major goal of this evaluation was to assess the advantages of combining the expressiveness of state-of-the-art generative models for document representation with information-theoretic distance; our strategy revealed to be a compelling solution for topic-based hard clustering of documents.

9.2 State of the Art

Identifying a topic feature space in a given document collection is traditionally accomplished by mapping the term-document representation to a lower-dimensional latent "semantic" space [DDF$^+$90]. Following this line, one of the earliest proposals for topic-based document representation via a generative model is *Probabilistic Latent Semantic Analysis* (PLSA) [Hof01]. As a probabilistic version of LSA [DDF$^+$90], originally introduced to better handling problems of term polysemy in document retrieval applications (e.g., [Bel98, LD97, WSR$^+$98]), PLSA defines a statistical latent topic model

in which the conditional probability between documents and terms is modeled as a latent variable. In this way, it is possible to assign an unobserved class variable to each observation (i.e., the occurrence of a word in a given document), since each document is composed by a mixture of distributions. Each term may belong to one or more classes and a document may discuss more than one topic.

PLSA assumes a bag-of-words document representation, in which the word order is disrupted. The so-called assumption of *exchangeability* for the words within a document can also be extended to the documents in a collection. The *Latent Dirichlet Allocation* (LDA) [BNJ03] is able to consider mixture models that express the exchangeability of both words and documents. In LDA, the generative process of a document collection consists of a three-level scheme that involves the whole corpus, the documents, and the words in each document. More precisely, for each document, a distribution over topics is sampled from a Dirichlet distribution; for each word in a document, a single topic is selected according to this distribution, and each word is sampled from a multinomial distribution over words specific to the sampled topic. In this way, LDA defines a more sophisticated generative model for a document collection, whereas PLSA generates a model for each document separately from the other ones in the collection.

Generative models for document representation like PLSA and LDA put the basis for a relatively recent corpus of study on model-based document classification and clustering. In the context of supervised classification, the *Parametric Mixture Model* (PMM) [US02] assumes that a multi-labeled text contains words that support for a mixture of categories. A basis vector is used for each document to express which category the document belongs to, and the model parameters of each single topic are mixed with an equal ratio. Recently, *Parametric Dirichlet Mixture Model* (PDMM) [SN07] has been proposed as an improved PMM which aims to address the problem of considering each topic with equal mixture ratio by using a Dirichlet distribution.

Ext-PLSA [KPAG08] has very recently been proposed as an extension of PLSA for document clustering tasks. The main idea is to overcome some limits deriving from a straightforward use of PLSA to derive clustering solutions, such as the one-to-one correspondence between clusters and topics. To meet the requirement that the number of desired clusters does not necessarily match the number of topics, Ext-PLSA introduces a new latent variable that allows words and documents to be clustered simultaneously.

9.3 A Framework for Topic-based Hard Clustering of Documents

In the following, we propose a framework to cluster multi-topic documents. The clustering scheme performs "hard" assignments, in such a way that each cluster is composed by documents that share similar multi-topic structure. In

order to achieve this objective, documents are modeled via a generative process, which maps documents form term feature space to topic feature space. The output of the generative process is a vector of probabilities underlying a mixture model, in which each mixture component reflects a particular topic. We resort to a proper distance function that is able to compare mixture models, and then perform clustering task to identify groups of documents with similar mixtures (i.e., similar multi-topic assignments).

Let $\mathcal{D} = \{d_1, \ldots, d_N\}$ be a collection of documents and $\mathcal{W} = \{w_1, \ldots, w_M\}$ be a set of terms occurring in the documents in $\mathcal{D}$, which corresponds to the vocabulary of $\mathcal{D}$. Moreover, let $\mathcal{T} = \{\tau_1, \ldots, \tau_T\}$ denote the set of topics underlying the documents in $\mathcal{D}$. Using a generative model for document representation, $\mathcal{T}$ is typically associated to a latent variable, since $\mathcal{T}$ is unknown. The topic distribution of any given document is represented as a probability mass function (pmf), since the variable associated to the topic distribution $\mathcal{T}$ can be seen as a discrete random variable, such that $\sum_{t=1}^{T} \Pr(\tau_t|d_i) = 1, \forall i \in [1..N]$. We denote with $\hat{d}_i$ the document d_i modeled as a probability distribution according to a given generative process and with $\hat{\mathcal{D}}$ the probabilistic representation of the document collection.

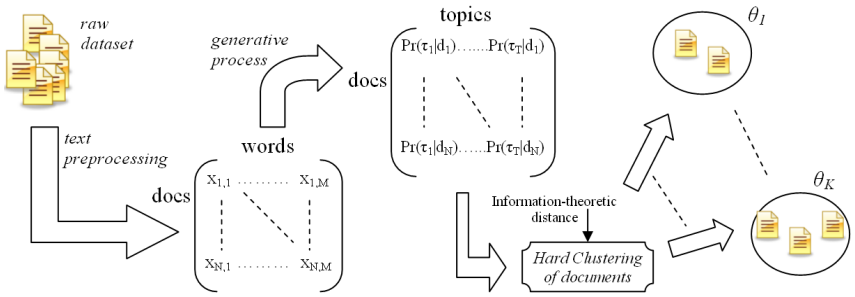


Fig. 9.1. Conceptual architecture of topic-based hard clustering of documents

Figure 9.1 depicts the conceptual structure of the framework, which consists of three main steps. First, the input corpus $\mathcal{D}$ of raw texts is structured by using standard preprocessing techniques for text data (e.g., lexical analysis, removal of stopwords, stemming); this step yields a word-occurrence matrix, where each element x_{ij} represents the number of occurrences of the word w_j in the document d_i.

The second step consists in applying a generative model to represent the documents into a feature space over the topics in $\mathcal{T}$. This step produces a probability matrix which expresses the mixture of topics underlying the input documents; precisely, the value $\Pr(\tau_t|d_i)$ is computed to measure the probability that the topic τ_t is associated to the document d_i, $\forall i \in [1..N]$ and $\forall t \in [1..T]$.

The final step is to cluster the documents such that each of the resulting clusters is characterized by a mixture of topics underlying the documents within that cluster. Given a clustering solution $\mathcal{C} = \{C_1, \ldots, C_K\}$ for $\hat{\mathcal{D}}$, there exists a set of topic-sets $\Theta = \{\theta_1, \ldots, \theta_K\}$, where $\theta_k \subset \mathcal{T}, \forall k \in [1..K]$ and $\bigcup_{k=1}^{K} \theta_k = \mathcal{T}$, such that each $C_k \in \mathcal{C}$ is described by one topic-set $\theta_k \in \Theta$.

It should be noted that while the assignment of documents to clusters is performed in a disjoint way, clusters correspond to possibly overlapping topic-sets. In a sense, this strategy introduce softness in the clustering solution (to a topic-level), although it is conceptually simpler than, e.g., fuzzy clustering because it does not allow overlaps of cluster in terms of their members (documents).

9.4 Distance Measure for Model-based Documents

The clustering algorithm in our framework is equipped with a distance measure that is able to compute the proximity between documents according to their mixtures of topics. In this respect, *information theory* represents a fruitful area of research. Two of the most frequently used measures for probability distributions are the Kullback-Leibler divergence [KL51, Kul59] and the Chernoff distance [Che52], which fall into the Ali-Silvey class of information-theoretic distance measures [AS66]. However, the use of such distances for document retrieval might not be a good idea due to a number of drawbacks. For instance, Kullback-Leibler divergence is not symmetric, while the Chernoff distance has a high computational complexity; in addition, both these functions do not satisfy the triangle inequality.

A well-suited distance measure exploits instead the notion of *Bhattacharyya coefficient* [Bha43, Kai67]. Given a discrete random variable X over a set of values $\mathcal{S} = \{s_1, \ldots, s_S\}$, $s_l \in \Re, \forall l \in [1..S]$, and any two probability distributions p and $p\prime$ for X, the Bhattacharyya coefficient is defined as:

$$\rho(p, p\prime) = \sum_{x \in S} \sqrt{p(x)\ p\prime(x)} \tag{9.1}$$

The Bhattacharyya coefficient has an important geometric interpretation. In fact, it represents the cosine between the two vectors for p and $p\prime$, which are composed by the square root of the probabilities of the mixtures that shape p and $p\prime$ [Bha43]. This property is important in our setting, since each model-based document is expressed as a pmf which is a mixture of topics.

Among the various distance measures which can be defined based on the Bhattacharyya coefficient (e.g., [Kai67]), we resort to the following definition:

$$\mathrm{B}(p, p\prime) = \sqrt{1 - \rho(p, p\prime)} \tag{9.2}$$

Equation 9.2 has many advantages with respect to other Bhattacharyya distances; for instance, it satisfies the triangle inequality and is easier to compute in practice than its more general case (i.e., the Chernoff distance).

Algorithm 7 MuToT-AHC

Input: a corpus of model-based text data $\hat{\mathcal{D}} = \{\hat{d}_1, \ldots, \hat{d}_N\}$;
(optionally) a desired number $K = |\Theta|$ of topic-sets
Output: a set of partitions $\mathbf{C}$
1: $\mathcal{C} \leftarrow \{C_1, \ldots, C_N\}$ such that $C_i = \{\hat{d}_i\}, \forall i \in [1..N]$
2: $P_{C_i} \leftarrow \hat{d}_i, \forall i \in [1..N]$, as initial cluster prototypes
3: $\mathbf{C} \leftarrow \{\mathcal{C}\}$
4: **repeat**
5: let C_i, C_j be the pair of clusters in $\mathcal{C}$ such that $\frac{1}{2}(dist(P_{C_i \cup C_j}, P_{C_i}) + dist(P_{C_i \cup C_j}, P_{C_j}))$ is minimum
6: $\mathcal{C} \leftarrow \{C \mid C \in \mathcal{C}, C \neq C_i, C \neq C_j\} \cup \{C_i \cup C_j\}$
7: $\mathbf{C} \leftarrow \mathbf{C} \cup \{\mathcal{C}\}$
8: update prototypes P_C, $C \in \mathcal{C}$
9: **until** $|\mathcal{C}| = 1$ or $|\mathcal{C}| = K$

It should be noted that we resorted to Bhattacharyya distance also to deal with uncertain objects (cf. Section 8.3.1). However, in this case we use an information-theory-based distance measure not to compare uncertainty regions, but to analyze proximity between probability distributions related to mixture models. Therefore, since each pmf is related to a mixture model defined over the same random variable (i.e., the one which models topics in the document collection), Bhattacharyya distance measures how two mixture models are different each other.

9.5 A Centroid-based-linkage Agglomerative Hierarchical Algorithm for Multi-Topic Text data (MuToT-AHC)

The proposed clustering method is a centroid-based-linkage agglomerative hierarchical algorithm (AHC), called *Mu*lti-*To*pic *T*ext data (MuToT-AHC). The main features of MuToT-AHC, which is shown in Algorithm 7, concern (i) the computation of cluster prototypes (i.e., cluster centroids), and (ii) the cluster merging criterion. A cluster centroid is represented as a mixture that summarizes the pmfs of the documents within that cluster. The cluster merging criterion, which decides the pair of clusters to be merged at each step, uses a proper information-theoretic distance to compare the cluster centroids—as we previously discussed, in this chapter we employ the Bhattacharyya distance (Equation 9.2) as default.

Given a corpus $\hat{\mathcal{D}}$ of text data modeled via a generative algorithm, MuToT-AHC follows the classic AHC scheme to yield a hierarchy $\mathbf{C}$ of clustering solutions. Optionally, the algorithm requires a number of clusters which corresponds to a given number of topic-sets ($K = |\Theta|$). At each iteration of the algorithm, the prototype of each cluster is represented as the mean of the pfms of the documents within that cluster. The merging score criterion

Table 9.1. Datasets used in the experiments

dataset	*size (# of docs)*	*# of words*	*# of topic labels*	*# of topic-sets* ($|\Theta|$)	*# of docs*
RCV1	6,588	37,688	23	49	5,896
PubMed	3,687	85,771	15	50	2,627
CaseLaw	2,550	50,567	20	1,697	2,550
20Newsgroups	2,544	22,386	25	23	2,234

(Line 5) applies to each pair of clusters C_i and C_j, and computes the average distance between the prototype of each of such clusters (P_{C_i} and P_{C_j}) and the prototype of the union cluster ($P_{C_i \cup C_j}$). The pair of clusters which minimizes such a distance computation is then chosen as the pair of clusters to be merged. Intuitively, this criterion aims to measure the lowest error merging as the one which is closest to both the original clusters. The algorithm stops when the number of clusters is equal to one, or the desired number of clusters is reached.

9.5.1 Experimental Evaluation

A major goal of our experimental evaluation is to test the advantages of the model-based representation combined with the information-theoretic based distance. In particular, we exploited our algorithm on LDA, PLSA and Ext-PLSA. These methods require only the setting of the latent variable used in the generative process; since this variable is related to the topics, it is reasonable to set the number of its possible values to the number of topics for each dataset. Ext-PLSA has a further latent variable related to the size of the desired clustering.

Settings

Datasets

We used four multi-topic datasets, called RCV1, PubMed, CaseLaw, and 20Newsgroups, whose main characteristics are shown in the first three columns of Table 9.1.

RCV1 is a subset of the Reuters Corpus Volume 1 [LYR+04], which contains news headlines discussing topics related to, e.g., markets, politics, wars, crimes. PubMed is a collection of full free texts of biomedical articles available from the PubMed website.[1] Fifteen topics were selected from the Medline's Medical Subject Headings (MeSH) taxonomy ensuring that no ancestor-descendant relationship holds for any pair of the selected topics (e.g., viruses, medical informatics, mass spectrometry, genetics, etc.). CaseLaw is a corpus of tagged case law documents,[2] which is comprised of very long texts dis-

[1] http://www.ncbi.nlm.nih.gov/sites/entrez/
[2] http://caselaw.lawlink.nsw.gov.au/

cussing topics such as, e.g., sex, leases and rents, immigration, employment, divorces, etc. Finally, 20Newsgroups is a subset of an original collection of approximately 20,000 newsgroup documents, partitioned over 20 different newsgroups, including politics, religion, culture, and sciences.[3]

Generating reference partitions

To assess the proposed hard clustering framework, we chose to compare the obtained clustering solutions with a reference partition of the documents for each dataset used in the evaluation. Since topic-labels are available for any specific dataset (cf. Table 9.1), a reference partition was built up by (i) identifying a number of topic-sets, each of which covers a significant portion of the document collection, and finally (ii) assigning each document to one of the classes (topic-sets) identified.

Figure 9.2 shows an example in which five topics are distributed over six documents, where any cross denotes the assignment of a document to a topic-label. Three topic-sets can be identified in this example, which correspond to a partitioning of the document collection in three classes. Note that each topic can be included in more classes.

$$
\begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{array}
\begin{array}{c}
\begin{array}{ccccc} \tau_1 & \tau_2 & \tau_3 & \tau_4 & \tau_5 \end{array} \\
\left(\begin{array}{ccccc}
 & & \times & & \times \\ \hline
\times & & & \times & \\ \hline
\times & \times & & & \times \\ \hline
\times & & & \times & \\ \hline
 & & \times & & \times \\ \hline
\times & & & \times &
\end{array}\right)
\end{array}
\Rightarrow
\begin{array}{lcl}
\theta_1 = \{\tau_3, \tau_5\} & \rightarrow & \{d_1, d_5\} \\
\theta_2 = \{\tau_1, \tau_4\} & \rightarrow & \{d_2, d_4, d_6\} \\
\theta_3 = \{\tau_1, \tau_2, \tau_5\} & \rightarrow & \{d_3\}
\end{array}
$$

Fig. 9.2. Example of identification of topic-sets

The last two columns of Table 9.1 show the number of topic-sets identified ($|\Theta|$) and the corresponding number of documents involved for each dataset. We experimentally determined the minimum number of documents to belong to any given topic-set for each dataset, by tuning this parameter and performing different runs of the clustering algorithm. In nearly all cases this minimum number of documents was selected as equal to 20, which revealed to be a reasonable choice to obtain a good trade-off between accuracy results and cluster size. An exception is represented by CaseLaw, which has a high number of topic-sets; as a consequence, there are few documents in this dataset that share the same multi-topic labeling, i.e., many topic-sets individually include less than 20 documents.

[3] http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html

Table 9.2. Summary of accuracy results: (a) without term selection, (b) filtering out terms with DF<3%, and (c) with DF<5%

(a)

dataset	*model*	F_B	F_{KL}
RCV1	LDA	**.66**	.58
	PLSA	.57	.49
	Ext-PLSA	.60	.50
PubMed	LDA	**.54**	.42
	PLSA	.49	.36
	Ext-PLSA	.50	.38
CaseLaw	LDA	**.79**	.62
	PLSA	.77	.60
	Ext-PLSA	.77	.61
20Newsgroups	LDA	**.61**	.48
	PLSA	.52	.42
	Ext-PLSA	.54	.44

(b)

dataset	*model*	F_B	F_{KL}
RCV1	LDA	**.71**	.61
	PLSA	.62	.51
	Ext-PLSA	.64	.53
PubMed	LDA	**.59**	.45
	PLSA	.51	.40
	Ext-PLSA	.52	.42
CaseLaw	LDA	**.81**	.65
	PLSA	.78	.63
	Ext-PLSA	.79	.63
20Newsgroups	LDA	**.68**	.52
	PLSA	.59	.45
	Ext-PLSA	.61	.47

(c)

dataset	*model*	F_B	F_{KL}
RCV1	LDA	**.69**	.60
	PLSA	.60	.50
	Ext-PLSA	.63	.51
PubMed	LDA	**.58**	.43
	PLSA	.49	.38
	Ext-PLSA	.51	.39
CaseLaw	LDA	**.79**	.64
	PLSA	.77	.62
	Ext-PLSA	.78	.62
20Newsgroups	LDA	**.65**	.51
	PLSA	.52	.43
	Ext-PLSA	.53	.44

Assessing clustering solutions

Since we have a reference labeling for each document of the datasets we used, we resort to a well-known external validity criterion called F_1-measure (cf. Section 2.1.6) to assess the validity of the proposed clustering scheme.

Results

In this section, we present the main results obtained by our clustering algorithm on the various datasets. We also discuss how the accuracy trends vary as the clustering size decreases.

As far as the setup of the framework, we varied the generative model used for document representation, involving LDA, PLSA, and Ext-PLSA. Each of such models requires the setting of the latent variable used in the generative process; since this variable is related to the document topics, it is reasonable to set the number of its possible values to the number of topics for each dataset. Ext-PLSA has a further latent variable related to the size of the desired clustering. Moreover, MuToT-AHC was equipped with either symmetrized Kullback-Leibler (KL) or Bhattacharyya (B) distance.

Accuracy

We tested MuToT-AHC on the evaluation datasets which were preprocessed according to three different settings, each having a different impact on the vocabulary of the collection: the first setting corresponds to using the entire vocabulary, whereas the second and third settings correspond to filtering out terms with document frequency (DF) lower than 3% and 5%, respectively.

A number of observations can be made analyzing Table 9.2, which summarizes the F_1-measure scores obtained by our algorithm by varying the distance measure (i.e., F_B and F_{KL}) and the document representation model. First, our

clustering algorithm obtains significant improvements when equipped with the Bhattacharyya distance rather than with the Kullback-Leibler distance, regardless of the generative model and term selection used. This result supports what we mentioned in Section 9.2, that is, Bhattacharyya is more effective than Kullback-Leibler in estimating the best pair of clusters to be merged at each step; indeed, the computation of the symmetrized Kullback-Leibler leads to infinity when a topic is associated to a zero probability for only one of the two (document) pmfs [CT06].

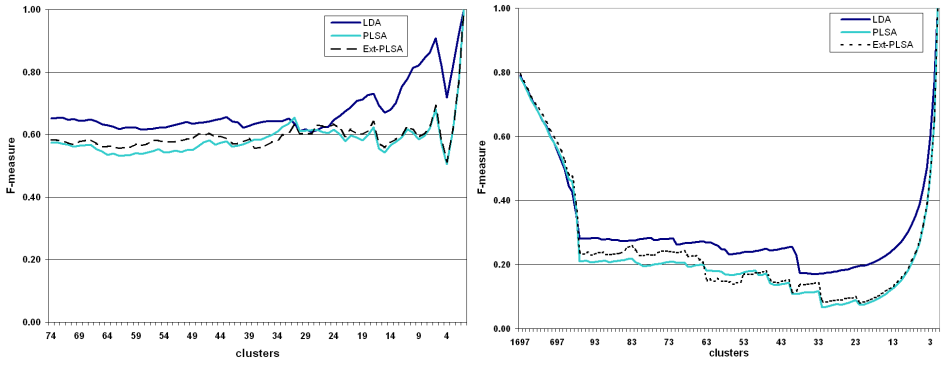
Second, LDA always performs better than the other generative models, and the improvement is significant on most datasets — for instance, in relation to the setting DF<3%, the improvement is up to 9% on RCV1 and 20Newsgroups, 8% on PubMed. An exception is represented by CaseLaw, for which the accuracy gap between the generative models is less evident. This mainly depends on the fact that few documents share the same topic-set in CaseLaw, since there are many distinct topic-sets in that dataset. As a consequence, the benefit deriving from considering the whole document collection in the generative process by LDA is lower than other cases. Another remark concerns Ext-PLSA, which behaves as good as or slightly better than PLSA.

A final remark can be made on the impact of term selection on the clustering performance. Filtering out terms that are "rare" to a certain degree may lead to higher clustering quality. In particular, the best results are obtained by setting DF<3%, whereas the setting DF<5% produce results that are quite close to those obtained when no term selection is carried out.

Scalability

We investigated the impact of different clustering sizes on the the clustering performance. For this purpose, we varied the size of the clustering solutions for each dataset from the original number of topic-sets (i.e., $|\Theta|$) to a minimum of 2 clusters; this corresponds to evaluating each level of the dendrogram produced by the MuToT-AHC algorithm according to a specific reference partition which was derived from the original set of topic-sets. Such reference partitions were built by merging, at each iteration of the algorithm, the two topic-sets sharing the maximal number of topics and having similar size. We accomplished this by computing the Jaccard distance on each pair of topic-sets θ_i and θ_j, which is formally expressed as $J(\theta_i, \theta_j) = 1 - |\theta_i \cap \theta_j|/|\theta_i \cup \theta_j|$. In this way, the pair θ_i and θ_j that maximizes the distance $J(\theta_i, \theta_j)$ is selected for merging.

Figure 9.3 shows F_1-measure trends of MuToT-AHC equipped with Bhattacharyya, as the clustering size decreases. For the sake of brevity of presentation, we report the graphs for two datasets only, namely 20Newsgroups and CaseLaw. As we can see in the figure, the relative difference of performance between the three generative models by varying the clustering size confirms the observations previously discussed. This also holds for the remaining datasets. However, CaseLaw represents a distinguished case, in which the reduction of



(a) 20Newsgroups (b) CaseLaw

Fig. 9.3. Clustering performance by varying the clustering size

clustering size impacts on the quality significantly. In particular, the clustering quality drastically decreases during the earlier iterations of the algorithm, due to a very high number of topic-sets for this dataset (cf. Table 9.1).

10

Collaborative Clustering of XML Documents

Summary. This chapter presents a distributed collaborative approach to XML document clustering. XML documents are mapped to a transactional domain, based on a data representation model which exploits the notion of XML tree tuple. This XML transactional model is well-suited to the identification of semantically cohesive substructures from XML documents, according to structure as well as content information. The proposed clustering framework employs a centroid-based partitional clustering paradigm in a distributed environment. Each peer in the network is allowed to compute a local clustering solution over its own data, then exchanges cluster centroids with other peers. The exchanged centroids correspond to recommendations offered by a peer to peers allowed to compute global representatives. Exploiting these recommendations, each peer becomes responsible for computing a global set of centroids for a given set of clusters. The overall clustering solution is hence computed in a collaborative way according to data from all the peers. Our approach has been evaluated on real XML document collections varying the number of peers. Results have shown that collaborative clustering leads to accurate overall clustering solutions, with a relatively low network overloading.

10.1 Introduction

The increasing availability of heterogeneous XML informative sources has raised a number of issues concerning how to represent and manage semistructured data. Indeed, XML allows the definition of semantic markup, that is, customized tags describing the data enclosed by them. As a consequence, the variety of application scenarios within XML is used makes XML information sources exhibit not only different structures and contents but also different ways to semantically annotate the data. However, most of the available information from such XML sources is of syntactic kind, consequently a basic assumption is that if certain syntactic properties are satisfied then semantic relationships in XML data can be discovered. Moreover, most XML documents available are often schema-less, that is no information about the document type definition is provided, making inferring semantics even more difficult.

In this context, a challenge is inferring semantics from XML documents according to the available syntactic information, namely *structure* and *content* features. This has several interesting application domains, such as integration of data sources and query processing, document retrieval, and Web intelligence, which can be seamlessly generalized to any kind of semistructured data.

The clustering problem finds in text databases a fruitful research area. Since today semistructured text data has become more prevalent on the Web, and XML is the de-facto standard for such data, *clustering XML documents* has increasingly attracted great attention. Any application domain that needs an organization of complex document structures (e.g., hierarchical structures with unbounded nesting, object-oriented hierarchies) as well as data containing a few structured fields together with some largely unstructured text components can be profitably assisted by an XML document clustering task. Clearly, clustering XML documents by facing both structure and content information turns out to be more difficult than the structure-only case. Indeed, mining XML content inherits some problems faced in traditional knowledge discovery in text (e.g., semantic ambiguity), while new ones arise when content is, as usual, contextually dependent on the logical XML structure.

Another problem with managing collections of XML documents is that often the size of such data is huge and inherently distributed, therefore classical centralized approaches may be not efficient. On the other hand, transferring all data to a central clustering service is prohibitive in large-scale systems.

Unfortunately, existing methods and systems for clustering XML data are designed to work only on a centralized environment. This partly depends on an inherent difficulty in devising suitable representation models for taking into account both structure and content information in such data. Moreover, most clustering strategies cannot easily be distributed, since there is an additional level of complexity due to the design and implementation of scalable and effective protocols for communication which allow nodes to minimize exchanged data.

The proposal is focused on the development of a distributed framework for efficiently clustering XML documents [GGPT09]. The distributed environment consists of a peer-to-peer network where each node in the network has access to a portion of the whole document collection and communicates with all the other nodes to perform a clustering task in a collaborative fashion.

The proposed framework is based on modeling and clustering XML documents taking into account both structure and content [TG06, TG09]. Indeed, XML documents are transformed into *transactional data* based on the notion of *tree tuple*. XML tree tuples enable a flat, relational-like XML representation that is well-suited to meet the requirements for clustering XML documents according to structure and content information.

The framework is based on the well-known paradigm of *centroid-based partitional clustering* to conceive the distributed, transactional clustering algorithm. It should be emphasized that such a clustering paradigm is particularly

appealing to a distributed environment. Indeed, the availability of a summarizing description of the clustered data provided by the cluster centroids is highly desirable especially when the input data is spread across different peers. Cluster centroids are hence used to describe portions of the document collection and can conveniently be exchanged with other nodes on the network.

The key idea underlying the collaborative clustering approach proposed in this framework can be intuitively described according to these two main features:

- Each node yields a local clustering solution (i.e., a partition of its own set of XML data);
- For each local cluster, the corresponding (local) centroid is computed and sent to nodes which are in charge of computing the "global" centroids. More precisely, every node computes a subset of the K global centroids; the i-th node computing the global centroid for a set of clusters, receives from each other node the centroid for the corresponding local cluster;
- Once computed, the global centroids are finally sent back to all the nodes to update their local clusters.

Experiments have been conducted on two large, real-world collections of XML documents, which are particularly suitable to assess the ability of the proposed framework in performing collaborative clustering of XML documents by structure and content. The number of nodes collaborating in the distributed environment for the computation of clusters was varied and the communication among nodes was kept minimized. Results have shown that, although the final clustering accuracy is typically reduced with respect to the centralized case, the parallelism due to a relatively small number of collaborating nodes in the network leads to a drastic reduction of the overall runtime needed for the clustering task.

10.2 State of the Art

10.2.1 XML Document Representation

A major issue in XML document clustering is the definition of a representation model which is well-suited to handle both structure and content information in XML data. Representing semistructured data has been traditionally addressed by labeled rooted trees. Consequently, handling such data has leveraged results from research on tree matching, including a number of algorithms for computing tree edit distances (e.g., [NJ02]). To address this issue, the level similarity measure is introduced in [NX06] to compute the structural match between elements according to the level information of each object (i.e., a tree, or a set of trees). Since the complexity issues relating to edit distances, summarization models have been also proposed to concisely represent XML

data while preserving some structural relationships between XML elements (e.g., [LCMY04, CMOT04, PG02]).

Recently attention has been drawn toward using simple Vector-space models to represent XML data, which substantially differ in the definition of feature space (e.g., [DL06, VFGL05, CTT05]). In [VFGL05], an XML document is projected into a vectorial space whose features take into account the frequency of structure within the documents. Each feature is characterized by a number of properties relating to subpaths, such as the path length, the root node label, and the number of path nodes. In [CTT05], XML documents are transformed into sets of attribute-values by focusing on different tree relationships among the nodes within the document trees (e.g., parent-child and next-sibling relations, set of distinct paths). In [DL06], the K-means algorithm is used and two feature sets are generated from the XML element texts and labels, respectively.

In [TG06, TG09], it has been originally introduced an XML representation model that allows for mapping XML document trees into *transactional data*. In a generic application domain, a transaction data set is a multi-set of variable-length sequences of objects with categorical attributes; in the XML domain, we devise a transaction as a set of items, each of which embeds a distinct combination of structure and content features from the original XML data. Within this view, XML documents are not directly transformed to transactional data, rather they are initially decomposed on the basis of the notion of *tree tuple*. Intuitively, given any XML document, a tree tuple is a tree representation of a complete set of distinct concepts that are correlated according to the structure semantics of the original document tree. Tree tuples extracted from the same tree maintain similar or identical structure while reflect different ways of associating content with structure as they can be naturally inferred from the original tree.

10.2.2 Transactional Clustering

Traditional clustering techniques assume data is memory-resident. However, this assumption does not hold in many large scale systems. In this respect, the development of clustering methods in parallel and distributed environment is becoming important. Indeed, distributed computing plays a key role since clustering and, in general, data mining tasks often require huge amounts of resources in storage space and computation time. Moreover, data is often inherently distributed into several databases, making a centralized analysis of such data inefficient and prone to security risks.

One of the earliest studies on distributed data mining is proposed in [KHS97], where an agent-based architecture is defined in such a way that each agent has a local model of the world and agents cooperate to improve solutions. The problem of document clustering in a distributed peer-to-peer network has been addressed recently. For instance, in [EMH03], the significance of centroid-based partitional clustering like K-means is leveraged as an efficient

approach to distributed clustering of documents. In [HK06], a collaborative approach to distributed clustering of document clustering is presented. In this collaborative approach, the individual local clustering solutions are improved exploiting the distributed environment on the basis of recommendations exchanged by the various peers. Since the data domain is textual, the document cluster summaries are modeled in form of keyphrases.

The collaborative approach to distributed clustering proposed in [HK06] is very close to ours. However, to the best of our knowledge, the method presented in this chapter addresses for the first time the problem of clustering XML documents by structure and content in a distributed network.

10.3 XML Transactional Representation

Preliminaries on XML trees and paths

A *tree* T is a tuple $T = \langle r_T, N_T, E_T, \lambda_T \rangle$, where $N_T \subseteq \mathbb{N}$ denotes the set of nodes, $r_T \in N_T$ is the distinguished root of T, $E_T \subseteq N_T \times N_T$ denotes the (acyclic) set of edges, and $\lambda_T : N_T \mapsto \Sigma$ is a function associating a node with a label in the alphabet Σ. Let Tag, Att, and Str be alphabets of tag names, attribute names, and strings respectively. An *XML tree* XT is a pair $XT = \langle T, \delta \rangle$, such that:

- T is a tree defined on the alphabet $\Sigma = Tag \cup Att \cup \{\texttt{S}\}$, where symbol $\texttt{S} \notin Tag \cup Att$ is used to denote the `#PCDATA` content model;
- given $n \in N_T$, $\lambda_T(n) \in Att \cup \{\texttt{S}\} \Leftrightarrow n \in Leaves(T)$;
- $\delta : Leaves(T) \mapsto Str$ is a function associating a string to a leaf node of T.

An *XML path* p is a sequence $p = s_1.s_2. \ldots .s_m$ of symbols in $Tag \cup Att \cup \{\texttt{S}\}$. Symbol s_1 denotes the tag name of the document root element. An XML path can be categorized into two types: *tag path*, if $s_m \in Tag$, or *complete path*, if $s_m \in Att \cup \{\texttt{S}\}$. We denote as $\mathcal{P}_{XT}$ the set of complete paths in XT. The length of the longest path in $\mathcal{P}_{XT}$ determines the depth of XT, denoted as $depth(XT)$.

Let $XT = \langle T, \delta \rangle$ be an XML tree, and $p = s_1.s_2. \ldots .s_m$ be an XML path. The application of p to XT identifies a set of nodes $p(XT) = \{n_1, \ldots, n_h\}$ such that, for each $i \in [1..h]$, there exists a sequence of nodes, or *node path*, $np_i^p = [n_{i_1}, \ldots, n_{i_m}]$ with the following properties:

- $n_{i_1} = r_T$ and $n_{i_m} = n_i$;
- $n_{i_{j+1}}$ is a child of n_{i_j}, for each $j \in [1..m\text{-}1]$;
- $\lambda(n_{i_j}) = s_j$, for each $j \in [1..m]$.

Moreover, we say that the application of a path to an XML tree yields an *answer*. Given an XML tree XT and a path p, the answer of p on XT is defined as either $\mathcal{A}_{XT}(p) \equiv p(XT)$ (i.e., the set of node identifiers $p(XT)$) if p is a tag path, or $\mathcal{A}_{XT}(p) = \{\delta_T(n) \mid n \in p(XT)\}$ (i.e. the set of string values associated to the leaf nodes identified by p) if p is a complete path.

XML tree tuples

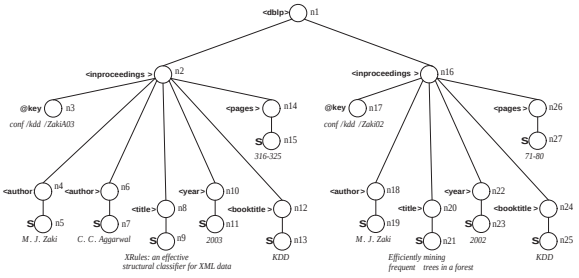
Tree tuple resembles the notion of tuple in relational databases and has been proposed to extend functional dependencies to the XML setting [AL04, FFGZ03]. In a relational database, a tuple is a function assigning each attribute with a value from the corresponding domain.

Definition 10.1 ([TG06, TG09]). Given an XML tree XT, an *XML tree tuple* τ derived from XT is a maximal subtree of XT such that, for each (tag or complete) path p in XT, the size of the answer of p on τ is not greater than 1, i.e., $|\mathcal{A}_\tau(p)| \leq 1$.

We denote with $\mathcal{T}_{XT}$ and $\mathcal{T}$ the set of tree tuples that can be derived from any given tree XT and from the collection $\mathcal{XT}$, respectively. Also, we use $\mathcal{P}_\tau$ to denote the set of complete paths in a tree tuple τ.

```
<dblp>
  <inproceedings key="conf/kdd/ZakiA03">
    <author>M. J. Zaki</author>
    <author>C. C. Aggarwal</author>
    <title>XRules: an effective structural
      classifier for XML data
    </title>
    <pages>316-325</pages>
    <year>2003</year>
    <booktitle>KDD</booktitle>
  </inproceedings>
  <inproceedings key="conf/kdd/Zaki02">
    <author>M. J. Zaki</author>
    <title>Efficiently mining
      frequent trees in a forest
    </title>
    <pages>71-80</pages>
    <year>2002</year>
    <booktitle>KDD</booktitle>
  </inproceedings>
</dblp>
```

(a)



(b)

Fig. 10.1. Example DBLP XML document (a) and its tree (b)

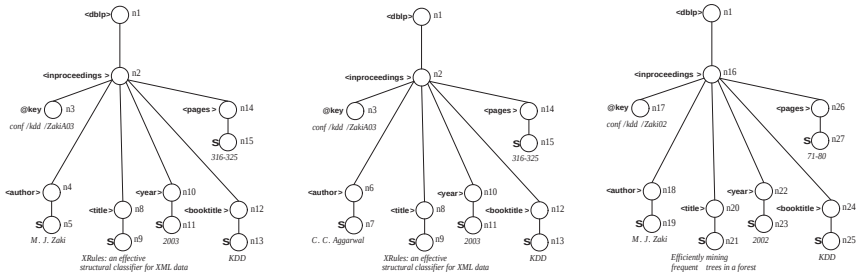


Fig. 10.2. The tree tuples extracted from the XML tree of Figure 10.1(b)

Example 10.2. Figure 10.1(a) shows a simplified XML document (from the DBLP archive) representing two conference papers, which is graphically represented by the XML tree in Figure 10.1(b). In the tree, any internal node has a unique label denoting a tag name. Each leaf node corresponds to either an attribute or `#PCDATA` content, and is labeled with either name and value of the attribute, or symbol `S` and the string corresponding to `#PCDATA`. As examples of path answers, path dblp.inproceedings.title yields the set of node identifiers $\{n_8, n_{20}\}$, whereas path dblp.inproceedings.author.S yields the set of strings {'M. J. Zaki', 'C. C. Aggarwal'}.

From the tree of Figure 10.1(b), we can extract three tree tuples, as shown in Figure 10.2. One tree tuple is extracted from the right subtree rooted in the dblp element (Figure 10.2(c)). Two distinct tree tuples are extracted from the left subtree rooted in dblp, as in this subtree there are two paths dblp.inproceedings.author, each of which yields a distinct path answer corresponding to one author of a paper. Suppose now that node n_3 is pruned from the subtree of Figure 10.2(a): in this case, the resulting tree is no more a tree tuple as it is not a maximal subtree.

A transactional model for XML tree tuples

Given a set $\mathcal{I} = \{e_1, \ldots, e_m\}$ of distinct categorical values, or *items*, a transactional database is a multi-set of transactions $tr \subseteq \mathcal{I}$. In our setting, the item domain is built over all the leaf elements in a given collection of XML tree tuples, that is the set of distinct answers of complete paths applied to the tree tuples. A transaction is then modeled with the set of items associated to the leaf elements of a specific tree tuple. The intuition behind such a model lies mainly on the definition of XML tree tuple itself: each path applied to a tree tuple yields a unique answer, thus each item in a transaction indicates information on a concept that is distinct from that of other items in the same transaction.

Definition 10.3 ([TG06, TG09]). Given an XML tree tuple τ and a path $p \in \mathcal{P}_\tau$, an *XML tree tuple item* in τ is a pair $\langle p, \mathcal{A}_\tau(p)\rangle$. The *XML trans-*

action corresponding to τ is the set of XML tree tuple items of τ, which is $\mathcal{I}_\tau = \{\langle p, \mathcal{A}_\tau(p)\rangle \mid p \in \mathcal{P}_\tau\}$. Given a collection $\mathcal{XT}$ of XML trees, the *XML transaction set* $\mathcal{S}$ for $\mathcal{XT}$ is defined as $\mathcal{S} = \bigcup_{XT \in \mathcal{XT}} \mathcal{S}_{XT}$, where $\mathcal{S}_{XT} = \{\mathcal{I}_\tau \mid \tau \in \mathcal{T}_{XT}\}$.

Example 10.4. In order to model XML tree tuples as transactions, we can decompose each tree tuple into its distinct paths and respective answers, as shown in Figure 10.3(a). For example, in tree tuple τ_1, the application of path dblp.inproceedings. @key yields the attribute value 'conf/kdd/ZakiA03' corresponding to node n_3. Then, item e_1 is associated to the above pair path-answer. Yet, the answer of path dblp. inproceedings.booktitle.S is the string 'KDD' corresponding to two nodes, n_{13} of tree tuples τ_1 and τ_2, and n_{25} of tree tuple τ_3.

Once the item domain has been completely defined, a transaction is assigned with each tree tuple by mapping its pairs path-answer into the corresponding items. A transactional representation of the tree tuples of Figure 10.2 is shown in Figure 10.3(c). Notice that, in the example, all the transactions contain the same number of tree tuple items, as their corresponding tree tuples have the same number of leaf nodes. Clearly, transactions might be differently sized, depending on the specific structure of the associated tree tuples.

10.4 Distributed XML Transactional Clustering

In this section, we describe how XML tree tuples modeled as transactions can be compared each other and clustered by applying a centroid-based partitional algorithm suitably designed for a collaborative environment.

10.4.1 XML Tree Tuple Item Similarity

As discussed in the previous section, XML features are represented by tree tuple items. To compare XML data in our transactional domain, we define a measure of similarity between tree tuple items according to their structure and content features.

Definition 10.5 ([TG06, TG09]). *Let e_i and e_j be two tree tuple items. The* tree tuple item similarity *function is defined as*

$$sim(e_i, e_j) = f \times sim_S(e_i, e_j) + (1 - f) \times sim_C(e_i, e_j), \tag{10.1}$$

where sim_S (resp. sim_C) denotes the structural (resp. content) similarity between the items, and $f \in [0..1]$ is a factor that tunes the influence of the structural part to the overall similarity.

path (p)	$\mathcal{A}_{\tau_1}(p)$	*node ID*
dblp.inproceedings.@key	'conf/kdd/ZakiA03'	n_3
dblp.inproceedings.author.S	'M. J. Zaki'	n_5
dblp.inproceedings.title.S	'XRules: an effective ...'	n_9
dblp.inproceedings.year.S	'2003'	n_{11}
dblp.inproceedings.booktitle.S	'KDD'	n_{13}
dblp.inproceedings.pages.S	'316-325'	n_{15}

path (p)	$\mathcal{A}_{\tau_2}(p)$	*node ID*
dblp.inproceedings.@key	'conf/kdd/ZakiA03'	n_3
dblp.inproceedings.author.S	'C. C. Aggarwal'	n_7
dblp.inproceedings.title.S	'XRules: an effective ...'	n_9
dblp.inproceedings.year.S	'2003'	n_{11}
dblp.inproceedings.booktitle.S	'KDD'	n_{13}
dblp.inproceedings.pages.S	'316-325'	n_{15}

path (p)	$\mathcal{A}_{\tau_3}(p)$	*node ID*
dblp.inproceedings.@key	'conf/kdd/Zaki02'	n_{17}
dblp.inproceedings.author.S	'M. J. Zaki'	n_{19}
dblp.inproceedings.title.S	'Efficiently mining ...'	n_{21}
dblp.inproceedings.year.S	'2002'	n_{23}
dblp.inproceedings.booktitle.S	'KDD'	n_{25}
dblp.inproceedings.pages.S	'71-80'	n_{27}

(a)

item ID	*associated node IDs*
e_1	n_3
e_2	$n_5,\ n_{19}$
e_3	n_9
e_4	n_{11}
e_5	$n_{13},\ n_{25}$
e_6	n_{15}
e_7	n_7
e_8	n_{17}
e_9	n_{21}
e_{10}	n_{23}
e_{11}	n_{27}

(b)

tr_1	$e_1\ e_2\ e_3\ e_4\ e_5\ e_6$
tr_2	$e_1\ e_7\ e_3\ e_4\ e_5\ e_6$
tr_3	$e_8\ e_2\ e_9\ e_{10}\ e_5\ e_{11}$

(c)

Fig. 10.3. Transactional representation of the tree tuples of Figure 10.2: (a) paths and answers, (b) item domain, and (c) transaction set

Since the combination of structure and content information characterizes an XML tree tuple item, it is advisable to take tolerance on computing similarity between XML tree tuple items. For this purpose, we introduce a similarity threshold that represents the minimum similarity value for considering two XML tree tuple items as similar.

Definition 10.6 ([TG06, TG09]). Given a real value $\gamma \in [0..1]$, two XML tree tuple items e_i and e_j are said to be γ-*matched* if $sim(e_i, e_j) \geq \gamma$.

Similarity by Structure

Structural similarity between two tree tuple items e_i and e_j is evaluated by comparing their respective tag paths.

Computing the similarity between any two paths is essentially accomplished by referring to it as a simple case of string matching of their respective element names, and finally averaging the (weighted) matchings. Given any two tags t and t', the Dirichlet function (δ) is applied in such a way that $\delta(t, t')$ is equal to one if the tags match, otherwise $\delta(t, t')$ is equal to zero.

Definition 10.7. Let e_i and e_j be XML tree tuple items, $p_i = t_{i_1}.t_{i_2}.\ \ldots\ .t_{i_n}$ and $p_j = t_{j_1}.t_{j_2}.\ \ldots\ .t_{j_m}$ be their respective tag paths. The *structural similarity* between e_i and e_j is defined as

$$sim_S(e_i, e_j) = \frac{1}{n+m}\left(\sum_{t \in p_i} sim(t, p_j) + \sum_{t \in p_j} sim(t, p_i)\right)$$

such that, for each $t_{i_h} \in p_i$

$$sim(t_{i_h}, p_j) = \text{avg}_{t_{j_k} \in p_j}\left\{\frac{1}{1+|h-k|} \times \delta(t_{i_h}, t_{j_k})\right\}$$

It should be noted that the tag matchings are corrected by a factor which is inversely proportional to the absolute difference of location of the tags in their respective paths: this avoids that two paths having the same but differently located tags are identified as highly similar to each other.

Similarity by Content

Content features are generated from the texts associated to XML tree tuple items. We refer to a *textual content unit* (for short, TCU) as the preprocessed text of a tree tuple item, i.e., a `#PCDATA` element content or an attribute value. Text preprocessing is accomplished by means of language-specific operations such as lexical analysis, removal of stopwords and word stemming [BYRN99].

Two statistical criteria are typically considered for measuring syntactic relevance of terms, namely term density in a given text and term rarity in the text collection. The popular *tf.idf* (term frequency - inverse document frequency) weighting function [BYRN99] takes both criteria into account. However, our XML transactional domain requires a more refined and structured modeling of term relevance, which is able to consider the term occurrences with respect to a context that includes TCUs, tree tuples and original document trees suitably.

Definition 10.8 ([TG06, TG09]). Given a collection of XML trees $\mathcal{XT}$, let w_j be an index term in a TCU u_i, which belongs to a tree tuple $\tau \in \mathcal{T}$ extracted from a tree $XT \in \mathcal{XT}$. The *ttf.itf* (*Tree tuple Term Frequency - Inverse Tree tuple Frequency*) weight of w_j in u_i with respect to τ is defined as

$$ttf.itf(w_j, u_i|_\tau) = tf(w_j, u_i) \times \exp\left(\frac{n_{j,\tau}}{N_\tau}\right) \times \frac{n_{j,XT}}{N_{XT}} \times \ln\left(\frac{N_\mathcal{T}}{n_{j,\mathcal{T}}}\right)$$

where:

- $tf(w_j, u_i)$ is the number of occurrences of w_j in u_i,
- $n_{j,\tau}$ is the number of TCUs in τ that contain w_j,
- N_τ is the number of TCUs in τ,
- $n_{j,XT}$ is the number of TCUs in XT that contain w_j,

- N_{XT} is the number of TCUs in XT,
- $n_{j,\mathcal{T}}$ is the number of TCUs in $\mathcal{T}$ that contain w_j,
- $N_{\mathcal{T}}$ is the number of TCUs in $\mathcal{T}$. □

Using the *ttf.itf* weighting function, the relevance of a term increases with the term frequency within the local TCU, with the term popularity across the TCUs of the local tree tuple (transaction) and the TCUs of the local document tree, and with the term rarity across the whole collection of TCUs.

Content similarity between any two tree tuple items is measured by comparing their respective TCUs. Given a collection of XML tree tuples $\mathcal{T}$, any TCU u_i is modeled with a vector $\mathbf{u}_i$ whose j-th component corresponds to an index term w_j and contains the *ttf.itf* relevance weight. The size of each TCU vector is equal to the size of the collection vocabulary, i.e., the set of index terms extracted from all TCUs in $\mathcal{T}$. The well-known *cosine similarity* [DM01] is used to measure the similarity between TCU vectors [SGM00].

Definition 10.9. Let e_i and e_j be tree tuple items, and $\mathbf{u}_i$ and $\mathbf{u}_j$ their respective TCU vectors. The *content similarity* between e_i and e_j is defined as

$$sim_C(e_i, e_j) = \frac{\mathbf{u}_i \cdot \mathbf{u}_j}{\|\mathbf{u}_i\| \times \|\mathbf{u}_j\|}$$

10.4.2 XML Transactional Clustering Algorithm for Collaborative Distributed Environment (CXK-means)

XML tree tuples modeled as transactions can be efficiently clustered by applying a partitional algorithm devised for the XML transactional domain.

Generally, given a set of data objects and a positive number K, a partitional clustering algorithm identifies K non-empty, disjoint groups each containing a homogeneous subset of objects. An important class of partitional approaches is based on the notion of *centroid*, or *representative*, of cluster: each object is assigned to a cluster C according to its distance from a data point c, which is the centroid of C.

In [TG06, TG09], a centroid-based partitional clustering algorithm has been developed, which is essentially a variant of the K-means algorithm for the XML transactional domain. From clustering strategy viewpoint, this algorithm works as a traditional centroid-based method to compute $K+1$ clusters: starts choosing K objects as the initial cluster centroids, then iteratively reassigns each remaining object to the closest cluster until all cluster centroids do not change. The $(K+1)$-th cluster, called *trash cluster*, is created to contain unclustered objects, i.e. objects having an empty intersection with each cluster centroid and so are not assigned to any of the first K clusters.

Two major aspects in the XML transactional clustering algorithm are (i) the notion of proximity used to compare XML transactions and (ii) the notion of cluster centroid.

In generic transactional domains, a widely used proximity measure is the Jaccard coefficient, which determines the degree of matching between any two transactions as directly proportional to their intersection (i.e., number of common items) and inversely proportional to their union. However, computing exact intersection between XML transactions is not effective, since XML tree tuple items may share structural or content information to a certain degree even though they are not identical. For this purpose, the notion of standard intersection between sets of items is enhanced with one able to capture even minimal similarities from content and structure features of XML elements.

Definition 10.10 ([TG06, TG09]). *Let tr_1 and tr_2 be two transactions, and $\gamma \in [0..1]$ be a similarity threshold. The set of* γ-shared items *between tr_1 and tr_2 is defined as*

$$match^{\gamma}(tr_1, tr_2) = match^{\gamma}(tr_1 \rightarrow tr_2) \ \cup \ match^{\gamma}(tr_2 \rightarrow tr_1), \tag{10.2}$$

where

$$match^{\gamma}(tr_i \rightarrow tr_j) = \{e \in tr_i \mid \exists e_h \in tr_j, \ sim(e, e_h) \geq \gamma, \\ \nexists e' \in tr_i, sim(e', e_h) > sim(e, e_h)\}.$$

The set of γ-shared items resembles the intersection between transactions at a degree greater than or equal to a similarity threshold γ. This notion of (enhanced) intersection is also at the basis of the following similarity function.

Definition 10.11 ([TG06, TG09]). *Let tr_1 and tr_2 be two transactions, and $\gamma \in [0..1]$ be a similarity threshold. The* XML transaction similarity *function between tr_1 and tr_2 is defined as*

$$sim_J^{\gamma}(tr_1, tr_2) = \frac{|match^{\gamma}(tr_1, tr_2)|}{|tr_1 \cup tr_2|}. \tag{10.3}$$

We adapted the XML transactional clustering algorithm to a collaborative distributed environment. Figure 10.4 sketches the main phases of the *CXK-means* algorithm, which has the following characteristics.

- Data are distributed over m nodes and each node communicates with all the other ones sending "local" representatives and receiving "global" representatives. An initial process corresponding to a node N_0 defines a partition of the k clusters into m subsets $Z_j, j \in [1..m]$. Each partition Z_j contains the identifiers of the clusters for which the node N_j has the responsibility of computing the global representatives.
- Each node N_i is in charge of computing local clusters $C_1^i, \ldots, C_k^i$ and local representatives $c_1^i, \ldots, c_k^i$, but also a subset of the global representatives $c_{i_1}, \ldots, c_{i_{q_i}}$ (using the local representatives computed by all nodes).
- The local representative of a cluster C is computed by starting from the set of γ-shared items among all the transactions within C. More precisely, for

Global Input:
A set $\mathcal{S}$ of XML transactions distributed over m nodes;
The desired number K of clusters;
A similarity threshold γ.
Global Output:
A partition C of $\mathcal{S}$ in K clusters distributed over m nodes;

Process N_0
Method:
define a partition of $\{1..K\}$ into m subsets $Z_1, \ldots, Z_m$;
for $i = 1$ **to** m **do**
send $(\{Z_1, \ldots, Z_m\}, K, \gamma)$ to N_i;

Process N_i
Input:
A set $\mathcal{S}^i = \{tr_1^i, \ldots, tr_{n_i}^i\}$ of XML transactions;
Output:
A partition $C^i = \{C_1^i, \ldots, C_k^i\}$ of $\mathcal{S}^i$ into K clusters.
Method:
receive $(\{Z_1, \ldots, Z_m\}, K, \gamma)$ from N_0;
let $Z_i = \{i_1, \ldots, i_{q_i}\}$, with $0 \leq q_i \leq k$;
/ selects q_i initial global clusters */*
select $\{c_{i_1}, \ldots, c_{i_{q_i}}\}$ transactions coming from distinct original trees;
$C_j^i = \{\}$, $\forall j \in [1..k]$;
repeat
send (broadcast) $\{c_{i_1}, \ldots, c_{i_{q_i}}\}$ to $N_1, ..., N_m$;
receive c_j from N_h with $h \in [1..m]$ and $j \in Z_h$;
repeat */* computes local clusters */*
$C_j^i := \{tr \mid tr \in S^i \wedge sim_J^\gamma(tr, c_j^i) > sim_J^\gamma(tr, c_l^i), l \in [1..K]\}$, $\forall j \in [1..K]$;
$C_{K+1}^i := \{tr \mid sim_J^\gamma(tr, c_j^i) = 0\}$, $\forall j \in [1..K]$;
$c_j^i :=$ computeLocalRepresentative(C_j^i), $\forall j \in [1..K]$;
until $\mathcal{Q}(C^i)$ is maximized;
if c_j^i does not change, $\forall j \in [1..K]$ **then**
send (broadcast) $([], done)$;
else
send $(\{\langle c_j^i, |C_j^i|\rangle | j \in Z_h\}, continue)$ to N_h, $\forall h \in [1..m]$;
receive $(\{c_j^h | j \in Z_h\}, V_h)$ from N_h, $\forall h \in [1..m]$
if $(\exists h \in [1..m]$ s.t. $V_h = continue)$ **then**
for $j \in Z_i$ **do**
$c_j =$ computeGlobalRepresentative$(\{c_j^1, \ldots, c_j^m\})$;
until $V_1 = \cdots = V_m = done$;

Function computeLocalRepresentative$(C) : rep$;
$I_C = \{e \mid e \in tr \wedge tr \in C\}$;
let $P_C = \{p/h \mid \exists h$ items $(p, u) \in I_C\}$;
let $\boldsymbol{\mu_C} = \text{avg}_{e_i \in I_C}\{\mathbf{u}_i\}$;
for each $e \in I_C$ **do**
let $rank_S(e) = sum\{h \mid \exists e' = (p', u') \in I_C \wedge p'/h \in P_C \wedge sim_S(e, e') \geq \gamma\}/|P_C|$;
let $rank_C(e) = (\mathbf{u} \cdot \boldsymbol{\mu_C})/(\|\mathbf{u}\| \times \|\boldsymbol{\mu_C}\|)$, where $\mathbf{u}$ is the e's TCU vector;
$rank(e) = f \cdot rank_S(e) + (1 - f) \cdot rank_C(e)$;
let R_C be the list containing the element in I_C ordered by $rank$ values;
return GenerateTreeTuple(R_C);

Function computeGlobalRepresentative$(C) : rep$;
$I_C = \{e \mid e \in tr \wedge tr \in C\}$;
let $P_C = \{p/h \mid \exists h$ items $(p, u) \in I_C\}$;
let $\boldsymbol{\mu_C} = \text{avg}_{e_i \in I_C}\{\mathbf{u}_i\}$;
for each $e \in I_C$ **do**
let $g\_rank_S(e) = sum\{h | \exists e' = (p', u') \in I_C \wedge p'/h \in P_C \wedge sim_S(e, e') \geq \gamma\}/|P_C|$;
let $g\_rank_C(e) = (\mathbf{u} \cdot \boldsymbol{\mu_C})/(\|\mathbf{u}\| \times \|\boldsymbol{\mu_C}\|)$, where $\mathbf{u}$ is the e's TCU vector;
$g\_rank(e) = f \cdot g\_rank_S(e) + +(1-f) \cdot g\_rank_C(e)$;
let $\overline{I}_C$ be the list containing the element in I_C ordered by g_rank values;
return GenerateTreeTuple$(\overline{I}_C)$;

Function GenerateTreeTuple$(I_C) : rep$;
let $I_C^* \subseteq I_C$ be the set of items in I_C with the highest rank;
$rep := conflateItems(I_C^*)$;
$s_0 := \sum_{tr \in C} sim^\gamma(tr, rep)$; */*refines representative*/*
$I_C := I_C - I_C^*$;
let $|tr_{max}|$ be the max length of transaction within C;
while $(I_C \neq \emptyset \ \wedge \ |rep| \leq |tr_{max}|)$ **do**
let $I_C^* \subseteq I_C$ be the set of items in I_C with the highest rank;
$rep' := conflateItems(rep \cup I_C^*)$;
$s' := \sum_{tr \in C} sim^\gamma(tr, rep')$;
if $(s' \geq s_0)$ **then**
$I_C := I_C - I_C^*$; $s_0 := s'$; $rep := rep'$;
else return rep;
return rep;

Fig. 10.4. The *CXK-means* algorithm

each transaction in C, the union of the γ-shared item sets with respect to all the other transactions in C is obtained; this guarantees no dependence of the order of examination of the transactions. Then, a raw representative

is computed by selecting the items from these union sets with the highest frequency: the raw representative, however, may not have the form of a tree tuple, as some items therein may refer to the same path but with different answers. Function $conflateItems$ is applied to a set of items and, for each subset $I = \{e_{i1}, \ldots, e_{ik}\}$ of items sharing the same path p, yields one item that has p as path and the concatenation of the contents of items in I as its content. Finally, a greedy heuristic refines the current representative by iteratively adding the remaining most frequent items until the sum of pair-wise similarities between transactions and representative cannot be further maximized. Again, any refinement must guarantee that the resulting representative satisfies Def. 10.1.

- The global representative of a cluster C is computed by considering the m local representatives $c^1, \ldots, c^m$. The only difference with respect to the computation of the local cluster is that in the computation of the structural rank (here called g_rank) associated with an item e we consider the rank associated with each item (instead of the number of items) having a γ-matching.

Complexity

The complexity of the *CXK-means* algorithm depends on the cost of a single iteration. The relocation phase (i.e., assignment of transactions to the nearest clusters) is $\mathcal{O}(|\mathcal{S}^i| \times K \times \mathrm{c}_{tr})$, where K is the desired number of clusters, and c_{tr} is the cost of computing the similarity between two transactions. This is bounded by $\mathcal{O}(|tr_{max}|^2 \times \mathrm{c}_e)$, where $|tr_{max}|$ is the maximum length of a transaction and c_e is the cost of computing the similarity between two items. Since $|tr_{max}|$ can be assumed to be constant, as it depends on the schema and c_e depends on the size of the vocabulary $\mathcal{W}$, the cost is bounded by $\mathcal{O}(|\mathcal{S}^i| \times K \times |\mathcal{W}|)$.

In the computeLocalRepresentative procedure, two phases can be distinguished: (a) computing item rankings, and (b) generation of the tree tuple representatives.

Given any cluster C_j^i (j-th cluster in the i-th node), phase (a) requires the computation of structural and content rankings. For structural ranking, the complexity is bounded by $\mathcal{O}(|I_{C_j^i}|)$ since the cost of computing the structural similarity between two paths is constant. For content ranking, the complexity is bounded by $\mathcal{O}(|I_{C_j^i}| \times \mathrm{c}_e) = \mathcal{O}(|S^i| \times |tr_{max}|^2 \times |\mathcal{W}|) = \mathcal{O}(|S^i| \times |\mathcal{W}|)$.

Phase (b) is performed by the function GenerateTreeTuple and consists of the two subphases: (b.1) item conflating, and (b.2) refinement. Selecting the items with the highest rank is bounded by $\mathcal{O}(|I_C|)$, whereas concatenating the TCUs of items having the same path (function $conflateItems$) has a cost bounded by c_e. Therefore, phase (b.1) has a global cost $\mathcal{O}(|I_C| \times |\mathcal{W}|)$. Phase (b.2) iteratively refines the current representative in at most $|I_C|$ steps. This refinement requires to compute the sum of similarities between the current representative and all the transactions in the cluster, with cost $\mathcal{O}(|C_j^i| \times \mathrm{c}_{tr})$, at

most $|tr_{max}|$ times. Therefore, the cost of phase (b.2) is bounded by $\mathcal{O}(|C_j^i| \times |\mathcal{W}|) = \mathcal{O}(|S^i| \times |\mathcal{W}|)$, which is also the upper bound for the global cost of function computeLocalRepresentative. Since this function is called K times, we have a cost $\mathcal{O}(K \times |S^i| \times |\mathcal{W}|)$.

Analogously, the cost of computing global representatives by a node N_i is bounded by $\mathcal{O}(m \times |\mathcal{W}|)$, since m is the size of the input dataset. Moreover each node computes $m_i - m_{i-1} + 1$ of global representatives at a cost $\mathcal{O}((m_i - m_{i-1} + 1) \times m \times |\mathcal{W}|) = \mathcal{O}(\lceil K/m \rceil) = \mathcal{O}(K \times |\mathcal{W}|)$. Therefore, the complexity of the main memory operations is $\mathcal{O}(|S^i| \times K \times |\mathcal{W}|)$.

Considering the cost of communications, we have that each node sends at each step:

- $(m_i - m_{i-1} + 1)$ tree tuples (global representatives) to all nodes with cost $\mathcal{O}(\lceil K/m \rceil \times m \times |tr_{max}|) = \mathcal{O}(K \times |\mathcal{W}|)$, and
- K tree tuples (local representatives) to a single node with cost $\mathcal{O}(K \times |\mathcal{W}|)$.

Moreover, each node also receives K tree tuples (global representatives) with cost $\mathcal{O}(K \times |\mathcal{W}|)$. Therefore, the global cost of communications is bounded by $\mathcal{O}(K \times |\mathcal{W}|)$.

Assuming that the number of iteration is q, the cost of the main memory operations and communications are bounded by, respectively, $\mathcal{O}(|S^i| \times K \times |\mathcal{W}| \times q)$ and $\mathcal{O}(K \times |\mathcal{W}| \times q)$. Considering the case where the set of tree tuples $\mathcal{S}$ is uniformly partitioned over the nodes ($|S^i| = |S|/m$), the complexity of the main memory operations performed at each step is bounded by $\mathcal{O}(|S|/m \times K \times |\mathcal{W}|)$. Let c_1 be the cost of the main memory operations and c_2 be the cost of transferring an element of the vocabulary (term). The global cost at each step is $\mathcal{O}((|S|/m \times c_1 + c_2) \times K \times |\mathcal{W}|)$, whereas for the centralized case (i.e., $m = 1$ and $c_2 = 0$) the cost is $\mathcal{O}(|S| \times K \times |\mathcal{W}|)$. Therefore, if $|\mathcal{S}| > c_2/c_1$, the distributed computation is less expensive than the centralized one and the improvement is equal to m. However, for small datasets, i.e., for datasets such that $|\mathcal{S}| < c_2/c_1$, the performance of the distributed computation decreases and the centralized computation overcomes the distributed one.

10.5 Experimental Evaluation

10.5.1 Settings

We assessed the proposed framework in performing clustering according to structure, content, or both information. We hereinafter refer to these kinds of solutions as *structure-driven*, *content-driven*, and *structure/content-driven* clustering, respectively. The first two types of clustering concern the detection of groups of XML data which are homogeneous by either structure or content. The third type (i.e., structure/content-driven clustering) includes a variety of scenarios, ranging from detecting common structures across different topics,

or conversely, to identifying classes of tree tuples that both cover common topics and belong to the same structural category.

The three types of clustering correspond to different settings of the parameters f and γ, which control the XML transaction similarity function. According to [TG06, TG09], we varied f within [0..1] with step 0.1, and γ within [0.5..1) with step 0.05—we chose $\gamma = 0.5$ as the maximum tolerance threshold in computing similarities. Also, since the setting of f depends on the clustering goal, we decided to partition the (discrete) interval [0..1] as follows: [0..0.3] for content-driven clustering, [0.4..0.6] for structure/content-driven clustering, and [0.7..1] for structure-driven clustering.

Network topology is characterized by the number of nodes. We performed experiments by varying this parameter from a minimum number of 1 up to 19 nodes, in order to assess the impact of the network size on the clustering task in terms of both effectiveness and efficiency. Clearly, a number of nodes equal to 1 refers to centralized clustering, which represents the baseline case. In the distributed case, data was equally partitioned over the nodes.

Datasets

We used two real word document collections for the evaluation which are particularly suited to be used in each of the three types of clustering.

The IEEE data set refers to the IEEE collection version 2.2, which has been used as a benchmark in the INEX document mining track 2008.[1] IEEE consists of 4,874 articles originally published in 23 different IEEE journals from 2002 to 2004. Such articles follow a complex schema which includes front matter, back matter, section headings, text formatting tags and mathematical formulas. We kept most of the logical structure elements and removed the stylistic markups, as shown in the DTD of Figure 10.5. In our XML transactional domain, the IEEE collection has 211,909 transactions and 135,869 items. Also, the number of leaf nodes is 228,869, the maximum fan out is 43, and the average depth is about 5.

In IEEE, the article journals determine the categories that were used to partition the collection, which strictly follow the original INEX categorization. Precisely, two structural categories correspond to "Transactions" and "non-Transactions" articles, respectively, whereas the classification by content organizes the articles by the following 8 topic-classes: "Computer", "Graphics", "Hardware", "Artificial Intelligence", "Internet", "Mobile", "Parallel", and "Security". Moreover, 14 hybrid classes are identified according to these structural and content classes.

The second evaluation data set is a subset of the DBLP archive,[2] a digital bibliography on computer science which contains citations on journal articles, conference papers, books, book chapters, and theses. DBLP is comprised of 3,000 documents which correspond to 5,884 transactions and 8,231 items.

[1] http://www.inex.otago.ac.nz/data/documentcollection.asp
[2] http://dblp.uni-trier.de/xml/

```
<!ELEMENT IEEE (article+)>
<!ELEMENT article (front_matter, body, back_matter?)>
<!ELEMENT front_matter (#PCDATA | abstract | author |
          bibliography | editor | editorial | figure |
          headers | keywords)*>
<!ELEMENT body (#PCDATA | ack | author |
          bibliography | figure | index | section |
          table | vita)*>
<!ELEMENT back_matter (#PCDATA | ack | author |
          bibliography | figure | footnote |
          section | table | vita)*>
<!ELEMENT section (#PCDATA | author | ack |
          bibliography | figure | index |
          sub_subsection | subsection | table |
          title | vita)*>
<!ELEMENT subsection (#PCDATA | author | ack |
          bibliography | figure | sub_subsection |
          table | title | vita)*>
<!ELEMENT sub_subsection (#PCDATA | bibliography |
          figure | table | title | vita)*>
<!ELEMENT ack (#PCDATA | figure | title)*>
<!ELEMENT author (#PCDATA | affiliation | email |
          first_name | surname)*>
<!ELEMENT editor (#PCDATA | affiliation | email |
          first_name | surname)*>
<!ELEMENT bibliography (#PCDATA | reference)*>
<!ELEMENT headers (header+)>
<!ELEMENT header (#PCDATA | title)*>
<!ELEMENT index (title, entries)>
<!ELEMENT figure (caption?)>
<!ELEMENT table (title, feet?)>
<!ELEMENT abstract (#PCDATA)>
<!ELEMENT affiliation (#PCDATA)>
 ...
<!ELEMENT vita (#PCDATA)>
```

Fig. 10.5. DTDs of the IEEE dataset

```
<!ELEMENT dblp (article | inproceedings | book |
          incollection)+>
<!ENTITY %field "author | editor | publisher |
         title | booktitle | journal | series">
<!ELEMENT article (%field;)*>
<!ELEMENT inproceedings (%field;)*>
<!ELEMENT proceedings (%field;)*>
<!ELEMENT book (%field;)*>
<!ELEMENT incollection (%field;)*>
<!ELEMENT author (#PCDATA)>
<!ELEMENT editor (#PCDATA)>
...
<!ELEMENT series (#PCDATA)>
```

Fig. 10.6. DTD of the DBLP dataset

DBLP is characterized by a small average depth (3), whereas the number of leaf nodes is 13,209 and the maximum fan out is 20. According to its element type definition (Figure 10.6), DBLP exhibits short text descriptions (e.g., author names, paper titles, conference names), and a moderate structural variety which corresponds to 4 main structural categories, namely 'journal articles' (article), 'conference papers' (inproceedings), 'books' (book), and 'book chapters' (incollection). Also, 6 topical classes are identified in DBLP, which are 'multimedia', 'logic programming', 'web and adaptive systems', 'knowledge based systems', 'software engineering', and 'formal languages'. If both content and structure information are taken into account, 16 classes are identified.

Note that the DTDs shown in Figure 10.5 and Figure 10.6 are given only for purposes of description of the data structures, whereas they were not used during the evaluation since our approach does not require the availability of XML schemas.

Cluster validity measures

To assess the quality of clustering solutions for the datasets, we exploited the availability of reference classifications for XML documents. The objective was to evaluate how well a clustering fits a predefined scheme of known classes (natural clusters). For this purpose, we resorted to the well-known F_1-measure (cf. Section 2.1.6).

10.5.2 Results

Tables 10.1–10.3 show the average clustering performance obtained on the various data sets by *CXK-means* varying the number of nodes and the type of clustering setting (i.e., structure-, content-, and structure/content-driven clustering). For each dataset and clustering setting, results refer to multiple (10) runs of the algorithm and were measured by averaging the F_1-measure scores over the range of f values specific of the clustering setting. As far as parameter γ, the best setting was found to be close to high values (typically above 0.85), for each dataset and type of clustering [TG09].

As it is reasonable to expect, the centralized case (i.e., one node) corresponds to an upper bound in terms of clustering quality for the collaborative distributed clustering. While our focus is not on the evaluation of the centralized case — the interested reader can find details in [TG09, TG06] — it can be noted how the clustering accuracy decreases as the number of nodes increases, regardless of the dataset and the type of clustering. This is explained since, the higher the number of nodes, the lower the distribution ratio of the transactions over the nodes; as a consequence, each node produces, at each step of the distributed algorithm, a local clustering solution over a too small portion of data, which cannot really represent the final overall solution. However, such a performance degradation remains relatively acceptable for a distributed environment, which is partly due to our model of cluster centroid in achieving good quality summaries for the clusters.

Figure 10.7 shows time performances on the two evaluation datasets by increasing the number of nodes and varying the dataset size. Both plots in the figure refer to structure/content-driven clustering experiments (i.e., $f \in [0.4..0.6]$). Here a noteworthy remark is that the performance of *CXK-means* takes major advantages with respect to a centralized setting in terms of runtime behavior. In fact, a higher number of nodes in the network leads to more parallelism, which results in a drastic reduction of the overall time needed for the clustering task, in spite of a relatively moderate decrease of the clustering quality.

dataset	*# of clusters*	*# of nodes*	F_1-*measure (avg)*
IEEE	8	1	0.593
		3	0.523
		5	0.485
		7	0.421
		9	0.376
DBLP	6	1	0.764
		3	0.702
		5	0.662
		7	0.612
		9	0.547

Table 10.1. Clustering results with $f \in [0..0.3]$ (content-driven similarity)

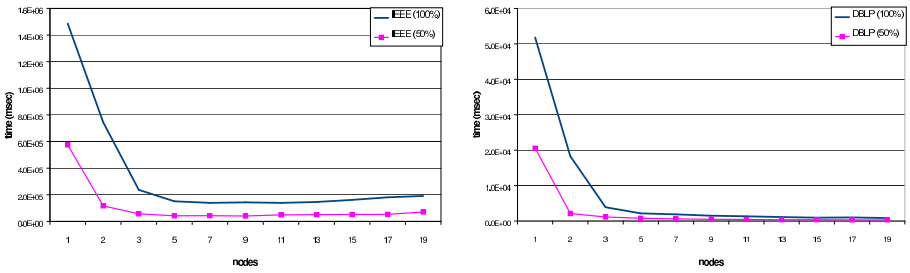
dataset	*# of clusters*	*# of nodes*	F_1-*measure (avg)*
IEEE	14	1	0.564
		3	0.497
		5	0.451
		7	0.404
		9	0.356
DBLP	16	1	0.772
		3	0.721
		5	0.676
		7	0.614
		9	0.558

Table 10.2. Clustering results with $f \in [0.4..0.6]$ (structure/content-driven similarity)

dataset	*# of clusters*	*# of nodes*	F_1-*measure (avg)*
IEEE	2	1	0.618
		3	0.542
		5	0.497
		7	0.433
		9	0.386
DBLP	4	1	0.988
		3	0.934
		5	0.882
		7	0.819
		9	0.716

Table 10.3. Clustering results with $f \in [0.7..1]$ (structure-driven similarity)

However, when the number of nodes grows up, the collaborative clustering algorithm also needs a higher number of iterations to converge. This fact affects negatively the network traffic (i.e., the centroid exchange) which might be not negligible anymore. Indeed, as we can see in Figure 10.7 for both datasets, after a drastic reduction of the runtime due to the use of just a few nodes, the runtime remains roughly constant for a certain range, then it starts to slightly increase when the number of nodes becomes significantly higher. In particular, time performances on IEEE tend to stabilize for six and



(a) (b)

Fig. 10.7. Clustering time performances varying the number of nodes and the dataset size: (a) IEEE, (b) DBLP

four nodes, respectively in the case of full and halved dataset; on DBLP, time performances tend to stabilize for a smaller number of nodes (four and two, respectively) which is due to a smaller size of DBLP with respect to IEEE, in terms of both transactions and vocabulary of terms.

Another important remark is that, as the dataset size is halved, the minimum number of nodes to bring down the clustering times tends to decrease. This suggests that the advantage of the collaborative distributed approach w.r.t. the centralized one tends to become less significant as the dataset size is reduced.

11

Conclusion

11.1 Thesis Review

In this thesis, we investigated the problem of complex data management. With the term "complex data" we referred to data that can not be represented in a fix structured scheme, or that can not be described by simple numerical values. The technological progress in many fields of business and commerce leads to the production of a large amount of complex data results of business activities. While in the past these data were collected by hand using papers and notes, nowadays it is no longer possible to use such storing methods because data are more complex and come from applications that have a high rate of data production. In addition, to define storing and modeling approaches allows these data to be analyzed by automatic or semi-automatic systems.

The research activities illustrated in this thesis focused on data mining tasks performed on complex data to highlight new relations and hidden information within them. There are many challenges in the field of managing complex data. However, in this thesis we faced two main issues, that are the way complex data can be modeled to capture main features within the data, and how defining data mining techniques to analyze complex data. Our researches in mining complex data produced interesting advances, especially in the fields of *time series data*, *uncertain data*, *text data*, and *biomedical data*.

In particular, as regards time series data management, *DSA* has been proposed, a representation model to support accurate and fast similarity detection in time series. DSA is able to transform a time series into a compact yet feature-rich sequence by combining the notions of derivative estimation, segmentation and segment modeling. DSA has been experimentally evaluated in clustering and classification frameworks, and compared to the state-of-the-art similarity measures and dimensionality reduction methods. Experiments conducted on various benchmark and real-world datasets have shown that performing dynamic time warping on DSA sequences leads to a good trade-off between effectiveness and efficiency in time series similarity detection.

In the field of uncertain data clustering, two algorithms have been proposed. The first one, i.e., *UK-medoids*, is a partitional clustering algorithm based on K-medoids clustering scheme. It exploits a new definition of cluster representative as an uncertain object, and uses a well-suited distance function for uncertain objects. Experiments have shown that UK-medoids outperforms other existing methods in terms of accuracy, regardless of the choice of uncertainty density function. Also, from an efficiency viewpoint, UK-medoids performs up to two orders of magnitude faster than the baseline method UK-means.

The second proposal for clustering uncertain data was *U-AHC*, a centroid-linkage-based agglomerative hierarchical algorithm. According to univariate and multivariate uncertainty models, a new notion of uncertain (cluster) prototype for uncertain objects has been defined, which is based on mixture densities from the pdfs associated to the objects belonging to the cluster. The cluster merging criterion in U-AHC exploits a new information-theoretic-based distance measure between uncertain prototypes. U-AHC has been experimentally shown to outperform major competing methods in terms of accuracy on all the datasets used in the evaluation. Also, from an efficiency viewpoint, U-AHC performs comparably to density-based clustering algorithms.

To manage and analyze text data, research activities produced two interesting results. The first one regarded the clustering of *multi topic documents*, as a framework for hard clustering of documents according to their mixtures of topics has been defined. Documents have been represented as probability distributions (mixtures of topics) and grouped together into disjoint clusters, each containing documents sharing the same topic-set. The framework combines the expressiveness of generative models for document representation with a properly chosen information-theoretic distance measure to group the documents. Experimental results have shown the effectiveness of the framework in clustering documents according to their mixtures of topics, and have highlighted the advantages offered by employing state-of-the-art document generative models and their combination with the Bhattacharyya distance.

The second proposal for text data management dealt with the definition of a collaborative distributed framework for clustering XML documents; to the best of our knowledge, this is the first collaborative approach to cluster XML documents by structure and content. XML documents are modeled in a transactional domain which is well-suited to extract XML structure as well as content features. The approach has been implemented in a distributed, centroid-based partitional clustering algorithm, where cluster centroids are used to describe portions of the document collection and can conveniently be exchanged with other nodes on the network. Each node yields a local clustering solution over its own set of XML data, and exchanges the cluster centroids with other nodes. These recommendations in form of centroids are used to compute global centroids, thus the overall clustering solution is computed in a collaborative way.

The last contributions regarded biomedical data. The focus here consisted in involving research results presented above to furnish analysis instruments that can help biomedical data analysts in better visualizing hidden relations within biomedical data. For these reasons, *MaSDA* has been defined, system for advanced data analysis and knowledge discovery from Mass Spectrometry data. MaSDA implements an approach to MS data representation that exploits a suitable model based on low-dimensional, dense time series (i.e., DSA). Using this MS data representation coupled with different choices of preprocessing settings, MS data can be effectively and efficiently managed and analyzed by employing data mining and knowledge discovery methods, including cluster analysis and classification, frequent pattern extraction, data summarization. The usefulness of MaSDA has been experimentally demonstrated in clinical applications, such as the unsupervised learning (clustering) of disease-related conditions for early detection of cancers.

As regards microarray data, U-AHC has been involved to perform gene-based clustering of microarray data with probe-level uncertainty. A strategy to model this uncertainty type in microarray data has been defined, in such a way that is possible to underline relevant data features. Experiments performed have shown the ability of our framework in discovering clustering solutions that are well-suited to capture the underlying gene-based patterns of microarray data.

11.2 Future Works

Though research results presented in this thesis represented important innovations and advances for complex data management, there is a continuous growth of new challenges in this field. Nevertheless, while facing issues related to complex data, it can be identified several aspects for further investigations in each proposal.

For time series data management, it can be found out that there are some particular application domains where it is crucial to distinguish time series according to their amplitude translation and scaling. DSA substantially works on a derivative version of a time series, and for this reason these two special requirements are not addressed. Moreover, since DSA revealed to achieve surprising results in 2-dimension, it would be interesting to extend it in multi-dimension domains, to perform similarity detection of trajectories, e.g., moving objects.

In the field of uncertain objects, there are two crucial aspects that need to be more investigated: the first one is related to the way uncertain objects are modeled, and the second one regards the function employed to define the distance between two uncertain objects. Despite of the fact that our proposals have been validated with proofs which provided theoretical validity of our intuitions, there are some aspects to improve. For instance, the distance function defined for uncertain objects is a combinations of information theory and

statistical notions. It may be interesting to study a way to formulate a new distance measure that does not need a compound definition. In addition, another interesting issue to investigate consists in improving the prototype representation model for a cluster of uncertain objects; this aspect is crucial since a high-feature rich cluster representative allows to improve both accuracy and efficiency of clustering solutions.

Text data representation and clustering are the newest research topic that I have faced. Research in this field followed the direction of probabilistic model, with particular emphasis on generative models for text representation. In fact, generative models are able to represent documents in a new feature space, which is typically identified as the topic space. Our proposal defines a way to achieve hard clustering of documents modeled according to a generative process, in order to highlight their multi-topic nature. However, generative models are substantially based on syntactic aspects; in this view, we think that it is very interesting to define new generative models that can exploit also semantic facilities, to perform a more accurate model estimation and topic identification. In addition, it is crucial to equip the proposed framework with model-based and soft clustering algorithms, to make a more exhaustive experimental comparison.

Semantics support can be also involved in the framework for collaborative distributed clustering of XML documents, to enrich structural and content aspects of XML documents. In addition, more experiments are needed to make the evaluation robust from both effectiveness and scalability viewpoints. In particular, it would be interesting to know on various application domains, how the collaborative clustering behavior varies in function of different data distributions.

A

Appendix — DDTW and DSA Derivative Estimation Models

Approximating the derivative of a given series plays an essential role in the DDTW method as well as in our DSA. We have described both the DDTW and DSA derivative estimation models in Section 4.2.1 (Eq. 4.1 and Eq. 4.2). Here we present some experimental results which show a comparison of these two models concerning their (i) performance in approximating real derivatives of standard functions and (ii) impact on the performance of DSA and DDTW.

Evaluation of the approximation of real derivative functions.

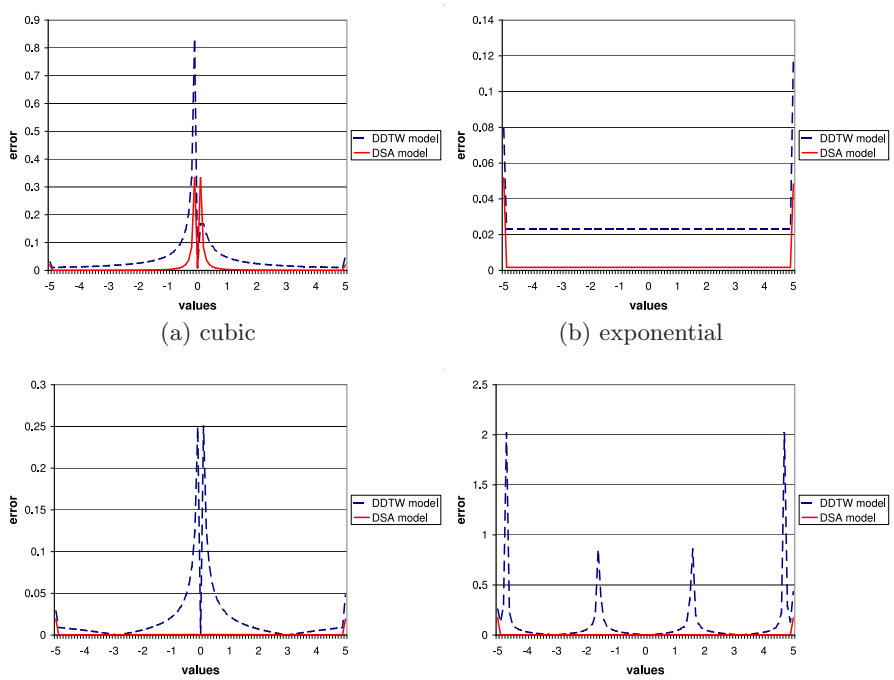
Let $f(t) : \Re \rightarrow \Re$ be a continuous function and $f'(t) : \Re \rightarrow \Re$ be its first derivative. Let $R = [t_1, \ldots, t_n] \in \Re^n$ denote a sequence of real values, over which we suppose to define a sequence T and the corresponding sequence T' of derivative values. Formally, let $T = [x_1, \ldots, x_n]$ and $T' = [x'_1, \ldots, x'_n]$, such that $x_h = f(t_h)$ and $x'_h = f'(t_h), \forall h \in [1..n]$. Given T, we also denote with $\hat{T}' = [\hat{x}'_1, \ldots, \hat{x}'_n]$ the derivative version of T which is obtained by a certain estimation model.

We compute the average approximation error of $\hat{T}'$ (i.e., the estimated derivative sequence) with respect to T' (i.e., the actual derivative sequence) as follows:

$$err(\hat{T}', T') = \frac{1}{n} \sum_{h=1}^{n} \frac{|\hat{x}'_h - x'_h|}{|x'_h|}$$

Figure A.1 shows a comparison between the DDTW and DSA derivative estimation models on four example functions, namely a cubic polynomial, an exponential function, a sine wave, and the Gaussian function.

Table A.1 reports on the average approximation errors (in percentage) obtained by using the two models on the selected functions. It can be noted that the DSA derivative estimation model produces an average error of approximation which is always lower than the error by the DDTW model.



(a) cubic (b) exponential

(c) Gaussian (d) sin wave

Fig. A.1. Approximation errors on derivative estimation: DDTW model vs. DSA model

Table A.1. Average approximation errors on derivative estimation. Each function is valued on 101 points over the range $[-5, +5]$.

function	*DDTW model*	*DSA model*
cubic	4.52%	1.12%
exponential	2.48%	0.26%
Gaussian	1.99%	0.08%
sine	11.48%	0.50%

Impact on the performance of DSA and DDTW.

In the main experimental sections we have presented clustering results obtained on the various data sets by using DSA and DDTW. Since both methods are characterized by a distinct model of derivative estimation, it was also interesting to gain an insight into the impact of this model on the performance of DSA and DDTW. For this purpose, we exchanged the respective models of derivative estimation in DSA and DDTW and then repeated the relative experimental evaluation in clustering frameworks.

Table A.2 shows the clustering results obtained by K-means and UPGMA when DDTW was equipped with the DSA derivative estimation model, and compares these results with those previously reported in Table 4.3 and Table 4.4. The modified version of DDTW led to better performances than the original DDTW method in most cases, in particular Mixed-BagShapes (4% by UPGMA and 2% by K-means), Twopat (3% by UPGMA and 1% by K-means), and ControlChart (2%).

Table A.2. DDTW-based clustering results by varying the derivative estimation model

clustering algorithm	derivative estimation	GunX	Tracedata	ControlChart	CBF	Twopat	Mixed-Bag Shapes	Ovarian Cancer
K-means	DSA (Eq. 4.2)	*.90*	*1*	*.91*	*.95*	*.96*	*.78*	*.63*
	DDTW (Eq. 4.1)	.89	1	.89	.96	.95	.76	.62
UPGMA	DSA (Eq. 4.2)	*.72*	*.78*	*.56*	*.48*	*.67*	*.45*	*.64*
	DDTW (Eq. 4.1)	.72	.76	.54	.49	.64	.41	.63

Analogously, Table A.3 reports the clustering results when DSA was equipped with the DDTW derivative estimation model, and compares them with the original performances of DSA. Again, the DSA derivative estimation model prevailed against the DDTW one in most cases — OvarianCancer (5% by UPGMA and 4% by K-means), CBF and GunX (4% by K-means and 2% by K-means), Mixed-BagShapes (3% by K-means and 2% by UPGMA), and Twopat (2%).

Table A.3. DSA-based clustering results by varying the derivative estimation model

clustering algorithm	derivative estimation	GunX	Tracedata	ControlChart	CBF	Twopat	Mixed-Bag Shapes	Ovarian Cancer
K-means	DSA (Eq. 4.2)	*.92*	*1*	*.90*	*.96*	*.97*	*.78*	*.75*
	DDTW (Eq. 4.1)	.90	1	.91	.92	.95	.75	.71
UPGMA	DSA (Eq. 4.2)	*.73*	*.82*	*.54*	*.60*	*.67*	*.51*	*.73*
	DDTW (Eq. 4.1)	.69	.80	.56	.62	.65	.49	.68

The above results led us to conclude that while the DSA derivative estimation model can enhance the DDTW method in practice; conversely, the DDTW derivative estimation model does not bring any beneficial effect to (in general, it may negatively affect) the performance of the DSA method.

B

Appendix — Impact of Preprocessing on Similarity Detection

As we have discussed in Section 4.3.1, smoothing was performed prior to the mining tasks in order to handle noise in the raw data, regardless of the particular representation method or distance measure used. Smoothing turned out to be useful for all the methods on every dataset—except for the OvarianCancer case. The intuition that skipping the smoothing stage would cause a decrease in performing similarity detection was supported by experimental evidence when we tried to directly classify the original (i.e., non-smoothed) data. Indeed, as shown for some prominent methods in Table B.1, the decrease would be significantly high in most datasets, with peaks of around 50% on ControlChart, CBF, and Twopat.

Table B.1. K-means clustering performance reduction in case of no smoothing

	GunX	Trace data	Control Chart	CBF	Twopat	Mixed-BagSh.
FTW	-21%	-16%	-41%	-29%	-23%	–
DTW	-16%	-2%	-46%	-48%	-53%	-4%
DDTW	-19%	–	-47%	-57%	-50%	–
DWT	-6%	-17%	-37%	-35%	-15%	–
SD	-5%	-19%	-45%	-46%	-47%	–
PLA	-14%	-1%	-47%	-46%	-35%	-3%
PAA	-7%	-2%	-46%	-46%	-30%	-5%
SAX	-22%	-2%	-43%	-47%	-25%	-4%
APCA	-18%	-5%	-48%	-42%	-46%	-4%
DSA	-19%	–	-48%	-54%	-51%	–

Another important remark is that the relative performances of most of the various methods (including our DSA) do not vary substantially whether or not smoothing is performed. This indicates that the representation model and

Table B.2. Summary of the preprocessing setups providing the best clustering results by K-means

	GunX	Trace data	Control Chart	CBF	Twopat	Mixed-BagSh.	Ovarian Cancer
LCSS	MA δ=9 it=3	MA δ=5 it=3	EXP ω=0.3 it=1	No smooth.	EXP ω=0.1 it=4	No smooth.	No smooth.
EDR	No smooth.	EXP ω=0.3 it=5	EXP ω=0.7 it=1	EXP ω=0.7 it=1	No smooth.	EXP ω=0.1 it=3	No smooth.
ERP	EXP ω=0.1 it=5	EXP ω=0.1 it=1	EXP ω=0.7 it=3	EXP ω=0.1 it=5	MA δ=5 it=3	EXP ω=0.5 it=5	No smooth.
FTW	EXP ω=0.3 it=3	No smooth.	EXP ω=0.7 it=3	EXP ω=0.1 it=3	EXP ω=0.3 it=3	EXP ω=0.5 it=2	No smooth.
DTW	EXP ω=0.1 it=3	No smooth.	EXP ω=0.9 it=1	No smooth.	EXP ω=0.9 it=5	EXP ω=0.1 it=5	No smooth.
DDTW	EXP ω=0.9 it=3	EXP ω=0.1 it=1	EXP ω=0.3 it=5	EXP ω=0.5 it=5	EXP ω=0.1 it=1	EXP ω=0.1 it=1	No smooth.
DFT	EXP ω=0.2 it=4	EXP ω=0.1 it=3	MA δ=9 it=2	EXP ω=0.1 it=1	EXP ω=0.2 it=4	MA δ=5 it=1	No smooth.
DWT	EXP ω=0.1 it=2	EXP ω=0.6 it=2	EXP ω=0.3 it=1	EXP ω=0.1 it=3	EXP ω=0.6 it=3	EXP ω=0.1 it=2	No smooth.
CHEBY	EXP ω=0.1 it=3	EXP ω=0.9 it=5	EXP ω=0.7 it=5	EXP ω=0.5 it=1	MA δ=9 it=3	EXP ω=0.9 it=3	No smooth.
SD	EXP ω=0.7 it=5	MA δ=9 it=5	EXP ω=0.6 it=4	EXP ω=0.3 it=4	EXP ω=0.3 it=5	EXP ω=0.9 it=4	No smooth.
PLA	EXP ω=0.1 it=1	MA δ=5 it=1	EXP ω=0.5 it=1	EXP ω=0.5 it=1	EXP ω=0.3 it=1	EXP ω=0.4 it=1	No smooth.
PAA	EXP ω=0.1 it=1	EXP ω=0.5 it=1	EXP ω=0.7 it=3	EXP ω=0.7 it=1	EXP ω=0.1 it=1	EXP ω=0.1 it=1	No smooth.
SAX	EXP ω=0.3 it=3	EXP ω=1 it=1	EXP ω=0.4 it=3	EXP ω=0.5 it=1	EXP ω=0.1 it=1	EXP ω=0.5 it=3	No smooth.
APCA	EXP ω=0.1 it=3	EXP ω=0.7 it=5	EXP ω=0.7 it=3	No smooth.	EXP ω=0.5 it=1	EXP ω=0.1 it=3	No smooth.
DSA	EXP ω=0.9 it=3	EXP ω=0.1 it=1	EXP ω=0.2 it=2	EXP ω=0.4 it=3	EXP ω=0.2 it=2	EXP ω=0.1 it=2	No smooth.

similarity/distance measure play a more important role than the preprocessing operations in determining the best approach(es) to similarity detection in time series.

A special remark should also be made on the OvarianCancer dataset which is huge-dimensional and largely affected by noisy factors, like most of mass spectra datasets (cf. Section 5.1). On this dataset, DSA performed far better than DDTW, precisely +13% by K-means, +10% by UPGMA and 5% by K-NN, in terms of F_1-measure (cf. Tables 4.3–4.5).

Table B.2 summarizes the best preprocessing setups for DSA and the other methods on the various datasets, using the K-means algorithm; we left the best setups for UPGMA and K-NN out of the presentation, since they resulted fairly similar to those obtained by K-means in most datasets. In the table, term MA (resp., EXP) stands for moving average (resp., exponential smoothing) and is followed by the value set for δ (resp., ω) and the number of iterations.

C

Appendix — Dynamic Kernel Clustering

We briefly present a preliminary investigation on the use of a more general notion of cluster prototype in time series clustering. For this purpose, we considered a *dynamic kernel clustering* algorithm. Dynamic kernel clustering [Did73] falls into the family of partitional clustering, and represents a generalization of centroid-based and medoid-based partitional clustering algorithms. Indeed, a major difference between dynamic kernel clustering and more popular algorithms (such as, e.g., K-means and K-medoids) consists in the notion of cluster prototype, which can be a centroid, a medoid, or even a whole set of points or objects. To conduct an experimental evaluation of dynamic kernel clustering in our context, we chose the following notion of cluster prototype based on a set of medoids: given any cluster C in the current partition, the prototype (or kernel) of C is the set of the H objects in C such that the sum of the distances of all the objects in C to such H objects is minimized.

Table C.1 summarizes the (average) F-measure scores obtained by dynamic kernel clustering and compares them to the corresponding results obtained by K-means on the various datasets. For each dataset and method, the table contains three values separated by the symbol '/':

- the first value refers to dynamic kernel clustering with $H = 2$;
- the second value refers to dynamic kernel clustering with $H = \sqrt{|C|}$, for each cluster C;
- the third value refers to K-means (which is extracted from Table 4.3).

In Table C.1 we can observe that there were no significant differences between dynamic kernel with $H = 2$ and K-Means. In most cases, the obtained F-measure scores were identical or very close ($\pm 1\%$ on average) to those obtained by K-means; in particular, as far as our DSA, the maximum improvement of quality with respect to K-means occurred in Mixed-BagShapes, which was about 2%.

Setting the kernel with $H = \lceil \sqrt{|C|} \rceil$, for each cluster C, led to further improvements of the clustering quality with respect to K-means in most cases. On average, the observed improvements were about 2-3%, with maximum

Table C.1. Summary of average quality results (F_1-measure) by dynamic kernel clustering and comparison with K-means clustering results

	GunX	Trace data	Control Chart	CBF	Twopat	Mixed BagSh.	Ovarian Cancer
LCSS	.58/.62/.59	.32/.32/.30	.50/.52/.50	.79/.79./.79	.37/.37/.36	.33/.35/.32	.35/.39/.34
EDR	.53/.53/.54	.72/.74/.74	.90/.91/.88	.86/.86/.86	.44/.44/.42	.69/.73/.70	.63/.63/.58
ERP	.73/.73/.72	.64/.65/.62	.77/.77/.76	.59/.59/.58	.38/.41/.39	.49/.53/.48	.34/.37/.34
DTW	.67/.68/.66	.77/.81/.78	.86/.86/.87	.89/.90/.89	.95/.95/.95	.78/.80/.77	.58/.63/.60
DDTW	.90/.91/.89	.99/1/1	.89/.89/.89	.96/.97/.97	.94/.96/.95	.78/.78/.76	.62/.66/.62
FTW	.75/.74/.74	.91/.92/.90	.80/.80/.81	.67/.67/.67	.57/.56/.55	.75/.75/.73	.57/.62/.58
L_2 on DFT	.65/.64/.63	.78/.78/.77	.78/.80/.78	.66/.69/.67	.42/.43/.39	.70/.71/.70	.41/.41/.36
L_2 on DWT	.59/.60/.61	.69/.69/.67	.76/.79/.76	.76/.76/.74	.37/.37/.36	.70/.71/.68	.40/.40/.36
L_2 on CHEBY	.56/.58/.57	.72/.74/.72	.70/.72/.70	.68/.70/.69	.40/.42/.38	.73/.75/.72	.36/.38/.34
DTW on SD	.68/.69/.67	.95/.96/.95	.85/.87/.85	.89/.89/.87	.87/.91/.89	.77/.79/.76	.69/.71/.69
DTW on PLA	.74/.74/.73	.78/.79/.77	.89/.91/.89	.88/.87/.87	.75/.76/.75	.77/.77/.74	.62/.65/.60
DTW on PAA	.69/.69/.68	.80/.80/.78	.87/.88/.87	.88/.89/.86	.72/.75/.73	.76/.78/.75	.60/.61/.59
DTW on SAX	.73/.75/.73	.76/.77/.77	.83/.84/.83	.87/.87/.87	.70/.70/.69	.73/.74/.71	.61/.61/.58
DTW on APCA	.76/.78/.77	.80/.80/.81	.88/.88/.89	.84/.84/.83	.90/.92/.91	.78/.79/.74	.57/.59/.55
DTW on DSA	.91/.94/.92	1/1/1	.89/.91/.90	.95/.95/.96	.97/.98/.97	.80/.83/.78	.76/.79/.75

increases of the quality up to 4-5% in Mixed-BagShapes and OvarianCancer. Indeed, as we expected, the dynamic kernel clustering revealed to be particularly advantageous for those datasets where the separation between the classes is less sharp (e.g., Mixed-BagShapes and OvarianCancer). In this case, using a set of medoids as cluster prototype was effective to better represent the clusters and improve the clustering quality.

However, the improvements in terms of clustering quality provided by the dynamic kernel algorithm appeared to be not enough large to justify a

runtime behavior that is much more costly than K-means. Moreover, the higher computational complexity of the dynamic kernel algorithm represents an efficiency issue that may limit the applicability to large datasets and real use cases.

Bibliography

[AA98] N. L. Anderson and N. G. Anderson. Proteome and Proteomics: new Technologies, new Concepts, and new Words. *Electrophoresis*, 19(11):1853–1861, 1998.

[ABKS99] M. Ankerst, M. M. Breunig, H. P. Kriegel, and J. Sander. OPTICS: Ordering Points To Identify the Clustering Structure. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 49–60, 1999.

[Abn91] S. Abney. Parsing By Chunks. In *Principle-Based Parsing: Computation and Psycholinguistics*. Kluwer Academic Publishers, 1991.

[AG97] A. Apostolico and Z. Galil. *Pattern Matching Algorithms*. Oxford University Press, 1997.

[Agg07] C. C. Aggarwal. On Density Based Transforms for Uncertain Data Mining. In *Proc. IEEE Int. Conf. on Data Engineering (ICDE)*, pages 866–875, 2007.

[AJ56] B. P. Adhikari and D. D. Joshi. Distance Discrimination Resume Exhaustif. *Public Institute of Statistic University Paris 5*, 5:57–74, 1956.

[AKG87] S. Abiteboul, P. Kanellakis, and G. Grahne. On the Representation and Querying of Sets of Possible Worlds. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 34–48, 1987.

[AL04] M. Arenas and L. Libkin. A Normal Form for XML Documents. *ACM Transaction on Database Systems*, 29(1):195–232, 2004.

[ALB+07] D. Apetrei, I. Lungu, F. Batrinu, G. Chicco, R. Porumb, and P. Postolache. Load Pattern Classification and Profiling for A Large Supply Company. In *Proc. Int. Conf. on Electricity Distribution (CIRED)*, pages 1–4, 2007.

[AM03] R. Aebersold and M. Mann. Mass Spectrometry-Based Proteomics. *Nature*, 422:198–207, 2003.

[AS66] S. M. Ali and S. D. Silvey. A General Class of Coefficients of Divergence of One Distribution from Another. *Royal Statistical Society*, 28(1):131–142, 1966.

[AY08] C. C. Aggarwal and P. S. Yu. Outlier Detection with Uncertain Data. In *Proc. SIAM Int. Conf. on Data Mining (SDM)*, pages 483–493, 2008.

[AY09] C. C. Aggarwal and P. S. Yu. A Survey of Uncertain Data Algorithms and Applications. *IEEE Trans. on Knowledge and Data Engineering (TKDE)*, 21(5):609–623, 2009.

[Bas89] M. Basseville. Distance Measures for Signal Processing and Pattern Recognition. *Signal Processing*, 18(4):349–369, 1989.

[BC94] D. J. Berndt and J. Clifford. Using Dynamic Time Warping To Find Patterns in Time Series. In *Proc. AAAI Workshop on Knowledge Discovery in Databases*, pages 359–370, 1994.

[BC09] A. Benis and M. Courtine. Biomarker Discovery in Medical Genomics Data. In *Proc. Int. Conf. on Bioinformatics & Computational Biology (BIOCOMP)*, pages 265–274, 2009.

[BCR05] B. Botte, V. Cannatelli, and S. Rogai. The Telegestore Project in ENEL's Metering System. In *Proc. Int. Conf. on Electricity Distribution (CIRED)*, 2005.

[Bel98] J. R. Bellegarda. Exploiting both Local and Global Constraints for Multi-Spanstatistical Language Modeling. *Acoustics, Speech and Signal Processing*, 2:677–680, 1998.

[Ber06] P. Berkhin. A Survey of Clustering Data Mining Techniques. *Grouping Multidimensional Data*, pages 25–71, 2006.

[Bez81] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, 1981.

[BF98] P. S. Bradley and U. M. Fayyad. Refining Initial Points for K-means Clustering. In *Proc. Int. Conf. on Machine Learning (ICML)*, pages 91–99, 1998.

[BGG97] C. S. Burrus, R. A. Gopinath, and H. Guo. *Introduction to Wavelets and Wavelet Transforms: A Primer*. Prentice Hall, 1997.

[Bha43] A. Bhattacharyya. On a Measure of Divergence Between Two Statistical Populations Defined by their Probability Distributions. *Bulletin of the Calcutta Mathematical Society*, 35:99–110, 1943.

[Bla99] W. Blackstock. Proteomics: Quantitative and Pphysical Mapping of Cellular Proteins. *Trends in Biotechnology*, 17(3):121–127, 1999.

[BM93] G. P. Babu and M. N. Murty. A Near-Optimal Initial Seed Value Selection in K-means Algorithm using a Genetic Algorithm. *Pattern Recognition Letters*, 14(10):763–769, 1993.

[BM04] P. V. Biron and A. Malhotra. XML Schema Part 2: Datatypes Second Edition, 2004. W3C Recommendation, http://www.w3.org/TR/2004/REC-xmlschema-2-20041028.

[BNJ03] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[BP03] A. Bouchachia and W. Pedrycz. A Semi-supervised Clutsering Algorithm for Data Exploration. In *Proc. Int. Fuzzy Systems Association World Congress (IFSA)*, pages 328–337, 2003.

[BPSM98] T. Bray, J. Paoli, and C. M. Sperberg-McQueen. Extensible Markup Language (XML) 1.0, 1998. W3C Recommendation, http://www.w3.org/TR/1998/REC-xml-19980210.

[BPSM+04] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau. Extensible Markup Language (XML) 1.0, 3rd edition, 2004. W3C Recommendation, http://www.w3.org/TR/2004/REC-xml-20040204.

[Bri90] E. H. Bristol. Swinging Door Trending: Adaptive Trending Recording. In *Proc. ISA National Conf.*, pages 749–753, 1990.

[BYRN99] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press Books. Addison Wesley, 1999.

[BZ04] J. Bi and T. Zhang. Support Vector Classification with Input Data Uncertainty. In *Proc. Int. Conf. on Neural Information Processing Systems (NIPS)*, pages 483–493, 2004.

[CAV01] S. Castano, V. De Antonellis, and S. Vimercati. Global Viewing of Heterogeneous Data Sources. *IEEE Journal on Knowledge and Data Engineering*, 13(2):277–297, 2001.

[CBM07] K. R. Coombes, K. A. Baggerly, and J. S. Morris. *Pre-Processing Mass Spectrometry Data*. Fundamentals of Data Mining in Genomics and Proteomics. Kluwer, 2007.

[CCKN06] M. Chau, R. Cheng, B. Kao, and J. Ng. Uncertain Data Mining: An Example in Clustering Location Data. In *Proc. Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD)*, pages 199–204, 2006.

[CF99] K. Chan and A. Fu. Efficient Time Series Matching by Wavelets. In *Proc. Int. Conf. on Data Engineering (ICDE)*, pages 126–133, 1999.

[CGS00] I. V. Cadez, S. Gaffney, and P. Smyth. A general probabilistic framework for clustering individuals and objects. In *Proc. ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 140–149, 2000.

[Che52] H. Chernoff. A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the Sum of Observations. *Annals of Mathematical Statistics*, 23(4):493–507, 1952.

[CKH07] C. K. Chui, B. Kao, and E. Hung. Mining Frequent Itemsets from Uncertain Data. In *Proc. Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD)*, pages 47–58, 2007.

[CKMP02] K. Chakrabarti, E. Keogh, S. Mehrotra, and M. Pazzani. Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases. *ACM Transactions on Database Systems*, 27(2):188–228, 2002.

[CKP03] R. Cheng, D. V. Kalashnikov, and S. Prabhakar. Evaluating Probabilistic Queries Over Imprecise Data. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 551–562, 2003.

[CLL06] V. Cantoni, L. Lombardi, and P. Lombardi. Challenges for Data Mining in Distributed Sensor Networks. In *Proc. Int. Conf. on Pattern Recognition (ICPR)*, pages 1000–1007, 2006.

[CMOT04] G. Costa, G. Manco, R. Ortale, and A. Tagarelli. A Tree-based Approach to Clustering XML Documents by Structure. In *Proc. European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 137–148, 2004.

[CN04a] Y. Cai and R. Ng. Indexing Spatio-Temporal Trajectories with Chebyshev Polynomials. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 599–610, 2004.

[CN04b] L. Chen and R. Ng. On The Marriage of Lp-norms and Edit Distance. In *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pages 792–803, 2004.

[CNP+05] G. Chicco, R. Napoli, F. Piglione, P. Postolache, M. Scutariu, and C. Toader. Emergent Electricity Customer Classification. *IEEE Proceedings Generation, Transmission & Distribution*, 152(2):164–172, 2005.

[CNP06] G. Chicco, R. Napoli, and F. Piglione. Comparisons Among Clustering Techniques for Electricity Customer Classification. *IEEE Transactions on Power Systems*, 21(2):933–940, 2006.

[CÖO05] L. Chen, M. T. Özsu, and V. Oria. Robust and Fast Similarity Search for Moving Object Trajectories. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 491–502, 2005.

[CRZ03] A. B. Chaudhri, A. Rashid, and R. Zicari. *XML Data Management: Native XML and XML-Enabled Database Systems.* Addison-Wesley, 2003.

[CS96] P. Cheeseman and J. Stutz. Bayesian Classification (AutoClass): Theory and Results. In *Advances in Knowledge Discovery and Data Mining*, pages 153–180. 1996.

[CT06] T. M. Cover and J. A. Thomas. *Elements of Information Theory (2nd Edition).* Wiley, 2006.

[CTT05] L. Candillier, I. Tellier, and F. Torre. Transforming XML Trees for Efficient Classification and Clustering. In *INitiative for the Evaluation of XML Retrieval (INEX) Workshop*, pages 469–480, 2005.

[DB75] L. Doyle and J. Becker. *Information Retrieval and Processing.* Melville, 1975.

[DDF+90] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.

[DDH01] A. Doan, P. Domingos, and A. Y. Halevy. Reconciling Schemas of Disparate Data Sources: A Machine-Learning Approach. In *Proc. Int. Conf. on Management of Data (SIGMOD)*, pages 509–520, 2001.

[DGM+05] A. Deshpande, C. Guestrin, S. Madden, J. M. Hellerstein, and W. Hong. Model-based approximate querying in sensor networks. *VLDB Journal*, 14(4):417–443, 2005.

[Did73] E. Diday. The Dynamic Clusters Method in Nonhierarchical Clustering. *Journal on Computer and Information Science*, 2(1):61–88, 1973.

[DL06] A. Doucet and M. Lehtonen. Unsupervised Classification of Text-Centric XML Document Collections. In *INitiative for the Evaluation of XML Retrieval (INEX) Workshop*, 2006.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–38, 1977.

[DM01] I. S. Dhillon and D. S. Modha. Concept Decompositions for Large Sparse Data Using Clustering. *Machine Learning*, 42(1):143–175, 2001.

[DM02] A. Doucet and H. A. Myka. Naive Clustering of a Large XML Document Collection. In *Proc. Annual ERCIM Workshop of the Initiative for the Evaluation of XML retrieval (INEX)*, pages 81–88, 2002.

[Dră03] S. Drăghici. *Data Analysis Tools for DNA Microarrays*. Chapman & Hall/CRC, 2003.

[DS04] N. N. Dalvi and D. Suciu. Efficient Query Evaluation on Probabilistic Databases. In *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pages 864–875, 2004.

[DS07] N. N. Dalvi and D. Suciu. Management of probabilistic data: foundations and challenges. In *Proc. Symposium on Principles of Database Systems (PODS)*, pages 1–12, 2007.

[Dub87] R. C. Dubes. How many clusters are best? — an experiment. *Pattern Recognition*, 20(6):645–663, 1987.

[EKSX96] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proc. ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 226–231, 1996.

[EMH03] M. Eisenhardt, W. Muller, and A. Henrich. Documents by Distributed P2P Clustering. In *Jahrestagung der Gesellschaft für Informatik e.V. (GI) (2)*, pages 286–291, 2003.

[FD95] R. Feldman and I. Dagan. Knowledge Discovery in Textual Databases (KDT). In *Proc. of Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 112–117, 1995.

[FFGZ03] S. Flesca, F. Furfaro, S. Greco, and E. Zumpano. Repairs and Consistent Answers for XML Data with Functional Dependencies. In *Proc. Int. XML Database Symposium (XSym)*, pages 238–253, 2003.

[FGB02] A. Faradjian, J. Gehrke, and P. Bonnet. GADT: A Probability Space ADT for Representing and Querying the Physical World. In *Proc. IEEE Int. Conf. on Data Engineering (ICDE)*, pages 201–211, 2002.

[Fis87] D. H. Fisher. Improving Inference through Conceptual Clustering. In *Proc. Int. Conf. on Artificial Intelligence (AAAI)*, pages 461–465, 1987.

[FPSS96a] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From Data Mining to Knowledge Discovery: An Overview. In *Advances in Knowledge Discovery and Data Mining*, pages 1–34. AAAI/MIT Press, 1996.

[FPSS96b] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. Knowledge Discovery and Data Mining: Towards a Unifying Framework. In *Proc. Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 82–88, 1996.

[FRVG05] V. Figueiredo, F. Rodrigues, Z. Vale, and J. B. Gouveia. An Electric Energy Consumer Characterization Framework based on Data Mining Techniques. *IEEE Transactions on Power Systems*, 20(2):596–602, 2005.

[FW04] D. C. Fallside and P. Walmsley. XML Schema Part 0: Primer Second Edition, 2004. W3C Recommendation, `http://www.w3.org/TR/2004/REC-xmlschema-0-20041028`.

[GGPT09] S. Greco, F. Gullo, G. Ponti, and A. Tagarelli. Collaborative Clustering of XML Documents. In *Proc. Int. Workshop on Distributed XML Processing: Theory and Practice*, 2009.

[GH88] D. E. Goldberg and J. H. Holland. Genetic Algorithms and Machine Learning. *Machine Learning*, 3:95–99, 1988.

[GLF89] J. H. Gennari, P. Langley, and D. H. Fisher. Models of Incremental Concept Formation. *Artificial Intelligence*, 40(1–3):11–61, 1989.

[GM00] D. Guillaume and F. Murtagh. Clustering of XML Documents. *Computer Physics Communications*, 127:215–227, 2000.

[GMW07] G. Gan, C. Ma, and J. Wu. *Data Clustering: Theory, Algorithms, and Applications*. ASA-SIAM Series on Statistics and Applied Probability, 2007.

[GPT+07] F. Gullo, G. Ponti, A. Tagarelli, G. Tradigo, and P. Veltri. A Time Series Based Approach for Classifying Mass Spectrometry

Data. In *Proc. IEEE Int. Symposium on Computer-Based Medical Systems (CBMS)*, pages 412–417, 2007.

[GPT08a] F. Gullo, G. Ponti, and A. Tagarelli. Clustering Uncertain Data Via K-Medoids. In *Proc. Int Conf. on Scalable Uncertainty Management (SUM)*, pages 229–242, 2008.

[GPT+08b] F. Gullo, G. Ponti, A. Tagarelli, G. Tradigo, and P. Veltri. MSPtool: A Versatile Tool for Mass Spectrometry Data Preprocessing. In *Proc. of the IEEE Int. Symposium on Computer-Based Medical Systems (CBMS)*, pages 209–214, 2008.

[GPT+09a] F. Gullo, G. Ponti, A. Tagarelli, S. liritano, M. Ruffolo, and D. Labate. Low-voltage electricity customer profiling based on load data clustering. In *Proc. Int. Database Engineering & Applications Symposium (IDEAS)*, pages 330–333, 2009.

[GPT+09b] F. Gullo, G. Ponti, A. Tagarelli, G. Tradigo, and P. Veltri. Hierarchical Clustering of Microarray Data with Probe-level Uncertainty. In *Proc. Int. Symposium on Computer-Based Medical Systems (CBMS)*, pages 1–6, 2009.

[GPT+09c] F. Gullo, G. Ponti, A. Tagarelli, G. Tradigo, and P. Veltri. MaSDA: A System for Analyzing Mass Spectrometry Data. *Computer Methods and Programs in Biomedicine*, 95(2):S12–S21, 2009. available online at http://dx.doi.org/10.1016/j.cmpb.2009.02.011.

[GPTG07] F. Gullo, G. Ponti, A. Tagarelli, and S. Greco. Accurate and Fast Similarity Detection in Time Series. In *Proc. Italian Symposium on Advanced Database Systems (SEBD)*, pages 172–183, 2007.

[GPTG08] F. Gullo, G. Ponti, A. Tagarelli, and S. Greco. A Hierarchical Algorithm for Clustering Uncertain Data via an Information-Theoretic Approach. In *Proc. IEEE Int. Conf. on Data Mining (ICDM)*, pages 821–826, 2008.

[GPTG09] F. Gullo, G. Ponti, A. Tagarelli, and S. Greco. A Time Series Representation Model for Accurate and Fast Similarity Detection. *Pattern Recognition*, 42(11):2998–3014, 2009.

[GSSC95] M. Garcia-Solaco, F. Saltor, and M. Castellanos. A Structure Based Schema Integration Methodology. In *Proc. Int. Conf. on Information and Data Engineering (ICDE)*, pages 505–512, 1995.

[GT06] T. Green and V. Tannen. Models for Incomplete and Probabilistic Information. *IEEE Data(base) Engineering Bulletin*, 29(1):17–24, 2006.

[GUP06] J. Galindo, A. Urrutia, and M. Piattini. *Fuzzy Databases: Modeling, Design, and Implementation.* Idea Group Publishing, 2006.

[Gus97] D. Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology.* Cambridge University Press, 1997.

[GV03] G. L. Glish and R. W. Vachet. The Basics of Mass Spectrometry in the Twenty-First Century. *Nature Reviews*, 2:140–150, 2003.

[Har75] J. A. Hartigan. *Clustering Algorithms*. Applied Statistics. John Wiley & Sons, 1975.

[Has70] W. K. Hastings. Monte Carlo Sampling Methods using Markov Chains and their Applications. *Biometrika*, 57(1):97–109, April 1970.

[HBV01] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On Clustering Validation Techniques. *Journal of Intelligent Information Systems*, 17(2):107–145, 2001.

[HBV02] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. Cluster Validity Methods: Part I. *SIGMOD Record*, 31(2):40–45, 2002.

[Hea99] M. A. Hearst. Untangling Text Data Mining. In *Proc. of Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3–10. Morgan Kaufmann, 1999.

[HK98] A. Hinneburg and D. A. Keim. An Efficient Approach to Clustering in Large Multimedia Databases with Noise. In *Proc. ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 58–65, 1998.

[HK00] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann. Addison-Wesley, 2000.

[HK06] K. Hammouda and M. Kamel. Collaborative document clustering. In *Proc. SIAM Int. Conf. on Data Mining (SDM)*, pages 211–214, 2006.

[HNP05] A. Hotho, A. Nürnberger, and G. Paass. A Brief Survey of Text Mining. *LDV Forum*, 20(1):19–62, 2005.

[HO82] C. M. Hoffmann and M. J. O'Donnell. Pattern Matching in Trees. *Journal of the ACM*, 29(1):68–95, 1982.

[Hof99] T. Hofmann. Probabilistic Latent Semantic Indexing. In *Proc. Annual ACM Conf. on Research and Development in Information Retrieval (SIGIR)*, pages 50–57, 1999.

[Hof01] T. Hofmann. Unsupervised Learning by Probabilistic Latent Semantic Analysis. *Machine Learning*, 42(1-2):177–196, 2001.

[HPM+03] S. R. Hingorani, E. F. Petricoin 3rd, A. Maitra, V. Rajapakse, C. King, M. A. Jacobetz, S. Ross, T. P. Conrads, T. D. Veenstra, B. A. Hitt, Y. Kawaguchi, D. Johann, L. A. Liotta, H. C. Crawford, M. E. Putt, T. Jacks, C. V. Wright, R. H. Hruban, A. M. Lowy, and D. A. Tuveson. Preinvasive and invasive ductal pancreatic cancer and its early detection in the mouse. *Cancer Cell*, 4(6):437–450, 2003.

[HRC+05a] A. M. K. Hein, S. Richardson, H. C. Causton, G. K. Ambler, and P. J. Green. BGX: a fully Bayesian integrated approach to the analysis of Affymetrix GeneChip data. *Biostatistics*, 6:349–373, 2005.

[HRC+05b] A. M. K. Hein, S. Richardson, H. C. Causton, G. K. Ambler, and P. J. Green. BGX: a fully Bayesian integrated approach to the

analysis of Affymetrix GeneChip data. *Biostatistics*, 6:349–373, 2005.

[HSS03] A. Hotho, S. Staab, and G. Stumme. Ontologies improve text document clustering. In *Proc. IEEE Int. Conf. on Data Mining (ICDM)*, pages 541–544, 2003.

[HY93] T. W. Hutchens and T. T. Yip. New Desorption Strategies for the Mass Spectrometric Analysis of Macromolecules. *Rapid Communications in Mass Spectrometry*, 7(7):576–580, 1993.

[IL84] T. Imielinski and W. Lipski Jr. Incomplete Information in Relational Databases. *Journal of the ACM (JACM)*, 31(4):761–791, 1984.

[ISS07] S. A. Imtiaz, M. A. A. Shoukat Choudhury, and S. L. Shah. Building Multivariate Model from Compressed Data. *Industrial Engineering Chemistry Research and Development*, 46(2):481–491, 2007.

[JD88] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, 1988.

[Jef05] N. O. Jeffries. Algorithms for Alignment of Mass Spectrometry Proteomic Data. *Bioinformatics*, 21(14):3066–3073, 2005.

[JH01] E. Jeong and C.-N. Hsu. Induction of Integrated View for XML Data with Heterogeneous DTDs. In *Proc. ACM Conf. on Information and Knowledge Management (CIKM)*, pages 151–158, 2001.

[JMF99] A. K. Jain, M. N. Murty, and P. J. Flynn. Data Clustering: A Review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[JTJ04] J. Johansson, R. Treloar, and M. Jern. Integration of Unsupervised Clustering, Interaction and Parallel Coordinates for the Exploration of Large Multivariate Data. In *Proc. Int. Conf. on Information Visualisation (IV)*, pages 52–57, 2004.

[JTZ04] D. Jiang, C. Tang, and A. Zhang. Cluster analysis for gene expression data: a survey. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1370–1386, 2004.

[Kai67] T. Kailath. The Divergence and Bhattacharyya Distance Measures in Signal Selection. *IEEE Transactions on Communication Technology*, 15(1):52–60, 1967.

[KAS98] K. V. Kanth, D. Agrawal, and A. Singh. Dimensionality Reduction for Similarity Searching in Dynamic Databases. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 166–176, 1998.

[KCPM01] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases. *Knowledge and Information Systems*, 3(3):263–286, 2001.

[KDK03] K. Kummamuru, A. Dhawale, and R. Krishnapuram. Fuzzy co-clustering of documents and keywords. In *Proc. of IEEE Int. Conf. on Fuzzy Systems (FS)*, volume 2, pages 772–777, 2003.

[Keo02] E. Keogh. Exact Indexing of Dynamic Time Warping. In *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pages 406–417, 2002.

[KH88] M. Karas and F. Hillenkamp. Laser Desorption Ionization of Proteins with Molecular Masses Exceeding 10.000 Daltons. *Analytical Chemistry*, 60:259–280, 1988.

[KHS97] R. Kargupta, I. Hanzaoglu, and B. Stafford. Distributed data mining using an agent based architecture. In *Proc. European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 211–214, 1997.

[Kil92] P. Kilpeläinen. *Tree Matching Problems with Applications to Structured Text Databases*. PhD thesis, University of Helsinki, 1992.

[KJF97] F. Korn, H. V. Jagadish, and C. Faloutsos. Efficiently Supporting Ad Hoc Queries in Large Datasets of Time Sequences. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 289–300, 1997.

[KKPR05] H. P. Kriegel, P. Kunath, M. Pfeifle, and M. Renz. Approximated Clustering of Distributed High-Dimensional Data. In *Proc. Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD)*, pages 432–441, 2005.

[KL51] S. Kullback and R. A. Leibler. On Information and Sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[KLC+08] B. Kao, S. D. Lee, D. W. Cheung, W. S. Ho, and K. F. Chan. Clustering Uncertain Data using Voronoi Diagrams. In *Proc. IEEE Int. Conf. on Data Mining (ICDM)*, pages 333–342, 2008.

[Kod99] Y. Kodratoff. Knowledge Discovery in Texts: A Definition, and Applications. In *Proc. Int. Symposium on Methodologies for Intelligent Systems (ISMIS)*, pages 16–29, 1999.

[Koh82] T. Kohonen. Self-Organized Formation of Topologically Correct Feature Maps. *Biological Cybernetics*, 43(1):59–69, 1982.

[Kol01] E. Kolatch. Clustering Algorithms for Spatial Databases: A Survey, 2001.

[KP98] E. Keogh and M. Pazzani. An Enhanced Representation of Time Series which allows Fast and Accurate Classification, Clustering and Relevance Feedback. In *Proc. Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 239–241, 1998.

[KP00] E. Keogh and M. Pazzani. Scaling up Dynamic Time Warping for Datamining Applications. In *Proc. Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 285–289, 2000.

[KP01] E. Keogh and M. Pazzani. Derivative Dynamic Time Warping. In *Proc. SIAM Int. Conf. on Data Mining*, 2001.

[KP05a] H. P. Kriegel and M. Pfeifle. Density-Based Clustering of Uncertain Data. In *Proc. ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 672–677, 2005.

[KP05b] H. P. Kriegel and M. Pfeifle. Hierarchical Density-Based Clustering of Uncertain Data. In *Proc. IEEE Int. Conf. on Data Mining (ICDM)*, pages 689–692, 2005.

[KPAG08] Y. M. Kim, J. F. Pessiot, M. R. Amini, and P. Gallinari. An Extension of PLSA for Document Clustering. In *Proc. ACM Int. Conf. on Information and Knowledge Management (CIKM)*, pages 1345–1346, 2008.

[KPC01] S. W. Kim, S. Park, and W. W. Chu. An Indexed-Based Approach for Similarity Search Supporting Time Warping in Large Sequence Databases. In *Proc. Int. Conf. on Data Engineering (ICDE)*, pages 607–614, 2001.

[KR87] L. Kaufmann and P. J. Rousseeuw. Clustering by means of medoids. In *Proc. Int. Conf. on Statistical Data Analysis based on the L_1 Norm and Related Methods*, pages 405–416, 1987.

[KR90] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis.* John Wiley & Sons, 1990.

[KS96] V. Kashyap and A. Sheth. Semantic and Schematic Similarities between Database Objects: A Context-Based Approach. *The VLDB Journal*, 5(4):276–304, 1996.

[Kul59] S. Kullback. *Information theory and statistics.* Wiley, 1959.

[KXWR] E. Keogh, X. Xi, L. Wei, and C. A. Ratanamahatana. The UCR Time Series Classification/Clustering Homepage. http://www.cs.ucr.edu/∼eamonn/time_series_data/.

[LC03] E. L. Lehmann and G. Casella. *Theory of Point Estimation.* Springer, 2003.

[LCMY04] W. Lian, D. W. Cheung, N. Mamoulis, and S.-M. Yiu. An Efficient and Scalable Algorithm for Clustering XML Documents by Structure. *IEEE Transaction on Knowledge Data Engineering*, 16(1):82–96, 2004.

[LD97] T. K. Landauer and S. T. Dumais. A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. *Psychological Review*, 104(2):211–240, April 1997.

[Lee92] S. K. Lee. An Extended Relational Database Model for Uncertain and Imprecise Information. In *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pages 211–220, 1992.

[LHY04] Y. Li, J. Han, and J. Yang. Clustering Moving Objects. In *Proc. ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 617–622, 2004.

[LKC07] S. D. Lee, B. Kao, and R. Cheng. Reducing UK-means to K-means. In *Workshop Proc. IEEE Int. Conf. on Data Mining (ICDM)*, pages 483–488, 2007.

[LKLC03] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A Symbolic Representation of Time Series, with Implications for Streaming Algorithms. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 2–11, 2003.

[LLAR07] X. Liu, K. K. Lin, B. Andersen, and M. Rattray. Including probe-level uncertainty in model-based gene expression clustering. *Bioinformatics*, 8:98, 2007.

[LLRS97] L. V. S. Lakshmanan, N. Leone, R. B. Ross, and V. S. Subrahmanian. ProbView: A Flexible Probabilistic Database System. *ACM Transactions on Database Systems (TODS)*, 22(3):419–469, 1997.

[LMLR05a] X. Liu, M. Milo, N. D. Lawrence, and M. Rattray. A Tractable Probabilistic Model for Affymetrix Probe-Level Analysis Across Multiple Chips. *Bioinformatics*, 21(18):3637–3644, 2005.

[LMLR05b] X. Liu, M. Milo, N. D. Lawrence, and M. Rattray. A tractable probabilistic model for Affymetrix probe-level analysis across multiple chips. *Bioinformatics*, 21(18):3637–3644, 2005.

[LO009] Clinical Genomics. In L. Liu and M. Tamer Özsu, editors, *Encyclopedia of Database Systems*, page 356. Springer US, 2009.

[LOW02] W. Lin, M. A. Orgun, and G. J. Williams. An Overview of Temporal Data Mining. In *Proc. of Australian Data Mining Workshop (ADM)*, 2002.

[LSS96] E.-P. Lim, J. Srivastava, and S. Shekhar. An Evidential Reasoning Approach to Attribute Value Conflict Resolution in Database Integration. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 8(5):707–723, 1996.

[Lu79] S. Lu. A Tree-to-Tree Distance and its Application to Cluster Analysis. *IEEE Journal on Pattern Analysis and Machine Intelligence*, 1(2):219–224, 1979.

[Luh58] H. P. Luhn. The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development*, 2(2):159–165, 1958.

[LW67] G. N. Lance and W. T. Williams. A General Theory of Classificatory Sorting Strategies: I. Hierarchical Systems. *The Computer Journal*, 9(4):373–380, 1967.

[LYR+04] D. D. Lewis, Y. Yang, T. G. Rose, G. Dietterich, and F. Li. RCV1: A new Benchmark Collection for Text Categorization Research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[MAG+97] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, and J. Widom. Lore: A Database Management System for Semistructured Data. *SIGMOD Record*, 26(3):54–66, 1997.

[MCK+05] J. S. Morris, K. R. Coombes, J. Koomen, K. A. Baggerly, and R. Kobayashi. Feature Extraction and Quantification for Mass Spectrometry in Biomedical Applications using the Mean Spectrum. *Bioinformatics*, 21(9):1764–1775, 2005.

[McQ67] J. B. McQueen. Some methods for classification and analysis of multivariate observations. In *Proc. Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[MFNL03a] M. Milo, A. Fazeli, M. Niranjan, and N. D. Lawrence. A Probabilistic Model for the Extraction of Expression Levels from Oligonucleotide Arrays. *Biochemical Society Transactions*, 31:1510–1512, 2003.

[MFNL03b] M. Milo, A. Fazeli, M. Niranjan, and N. D. Lawrence. A probabilistic model for the extraction of expression levels from oligonucleotide arrays. *Biochemical Society Transactions*, 31:1510–1512, 2003.

[MH03] J. C. Mason and D. Handscomb. *Chebyshev Polynomials*. Chapman & Hall, 2003.

[Mit97] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[MP80] W. Masek and M. Paterson. A Faster Algorithm Computing String Edit Distances. *ournal of Computer and System Sciences*, 20(1):18–31, 1980.

[MP00] G. McLachlan and D. Peel. *Finite Mixture Models*. Wiley Series in Probability and Statistics, 2000.

[MS99] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[Mur83] F. Murtagh. A Survey of Recent Advances in Hierarchical Clustering Algorithm. *The Computer Journal*, 26(4):354–359, 1983.

[Mur85] F. Murtagh. Multidimensional Clustering Algorithms. *Physica-Verlag*, 1985.

[NA08] T. T. T. Nguyen and G. J. Armitage. A Survey of Techniques for Internet Traffic Classification Using Machine Learning. *IEEE Communications Surveys and Tutorials*, 10(1–4):56–76, 2008.

[NDJR06] A. H. Nizar, Z. Y. Dong, M. Jalaluddin, and M. J. Raffles. Load Profiling Method in Detecting non-Technical Loss Activities in a Power Utility. In *Proc. IEEE Int. Power and Energy Conf. (PECon)*, pages 82–87, 2006.

[NJ02] A. Nierman and H. V. Jagadish. Evaluating Structural Similarity in XML Documents. In *Proc. ACM SIGMOD Int. Workshop on the Web and Databases (WebDB)*, pages 61–66, 2002.

[NKC+06] W. K. Ngai, B. Kao, C. K. Chui, R. Cheng, M. Chau, and K. Y. Yip. Efficient Clustering of Uncertain Data. In *Proc. IEEE Int. Conf. on Data Mining (ICDM)*, pages 436–445, 2006.

[NX06] R. Nayak and S. Xu. XCLS: A Fast and Effective Clustering Algorithm for Heterogeneous XML Documents. In *Proc. Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD)*, pages 292–302, 2006.

[Ols95] C. F. Olson. Parallel Algorithms for Hierarchical Clustering. *Parallel Computing*, 21(8):1313–1325, 1995.

[PAH+02] E. F. Petricoin 3rd, A. M. Ardekani, B. A. Hitt, P. Levine, V. A. Fusaro, and S. Steinberg. Use of proteomic patterns in serum to identify ovarian cancer. *Lancet*, 9306(359):572–577, 2002.

[PG02] N. Polyzotis and M. Garofalakis. Structure and Value Synopses for XML Data Graphs. In *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pages 466–477, 2002.

[PH74] T. Pavlidis and S. L. Horowitz. Segmentation of Plane Curves. *IEEE Transactions on Computers*, 23(8):860–870, 1974.

[PKS+04] J. Prados, A. Kalousis, J. C. Sanchez, L. Allard, O. Carrette, and M. Hilario. Mining Mass Spectra for Diagnosis and Biomarker Discovery of Cerebral Accidents. *Proteomics*, 4(6):2320–2332, 2004.

[POP+02] E. F. Petricoin 3rd, D. K. Ornstein, C. P. Paweletz, A. Ardekani, P. S. Hackett, B. A. Hitt, A. Velassco, C. Trucco, L. Wiegand, K. Wood, C. B. Simone, P. J. Levine, W. M. Linehan, M. R. Emmert-Buck, S. M. Steinberg, E. C. Kohn, and L. A. Liotta. Serum Proteomic Patterns for Detection of Prostate Cancer. *Journal of the National Cancer Institute*, 94(20):1576–1578, 2002.

[PRH+04] E. F. Petricoin 3rd, V. Rajapaske, E. H. Herman, A. M. Arekani, S. Ross, D. Johann, A. Knapton, J. Zhang, B. A. Hitt, T. P. Conrads, T. D. Veenstra, L. A. Liotta, and F. D. Sistare. Toxicoproteomics: Serum proteomic pattern diagnostics for early detection of drug induced cardiac toxicities and cardioprotection. *Toxicologic Pathology*, 32 Suppl 1:122–130, 2004.

[PT98] D. Pfoser and N. Tryfona. Requirements, Definitions, and Notations for Spatiotemporal Application Environments. In *Proc. ACM Int. Workshop on Advances in Geographic Information Systems (GIS)*, pages 124–130, 1998.

[PT09] G. Ponti and A. Tagarelli. Topic-Based Hard Clustering of Documents Using Generative Models. In *Proc. Int. Symposium on Methodologies for Intelligent Systems (ISMIS)*, pages 231–240, 2009.

[Ria] Rialto. Exeura S.r.l., http://www.exeura.com/rialto, 2009 edition. Easy Analytics, Ready for Decision Making.

[RJ93] L. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition*. Englewood Cliffs, New Jersey, 1993.

[RJFD99] A. F. Ruckstuhl, M. P. Jacobson, R. W. Field, and J. A. Dodd. Baseline Subtraction using Robust Local Regression Estimation. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 68:179–193, 1999.

[RM97] D. Rafiei and A. Mendelzon. Similarity-Based Queries for Time Series Data. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 13–25, 1997.

[RM98] D. Rafiei and A. Mendelzon. Efficient Retrieval of Similar Time Sequences Using DFT. In *Proc. Int. Conf. on Foundations of Data Organization and Algorithms (FODO)*, 1998.

[Rob81] S. E. Robertson. Term frequency and term value. *SIGIR Forum*, 16(1):22–29, 1981.

[Rog07] S. Rogai. The Telegestore Project: Progress and Results. In *Proc. IEEE Int. Symposium on Power Line Communications and Its Applications (ISPLC)*, 2007.

[RS76] S. E. Robertson and K. Sparck Jones. Relevance Weighting of Search Terms. *Journal of the Association for Information Systems*, 27(3):129–146, 1976.

[Sad91] F. Sadri. Modeling Uncertainty in Databases. In *Proc. IEEE Int. Conf. on Data Engineering (ICDE)*, pages 122–131, 1991.

[Sal89] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, 1989.

[SBHW06] A. D. Sarma, O. Benjelloun, A. Halevy, and J. Widom. Working Models for Uncertain Data. In *Proc. IEEE Int. Conf. on Data Engineering (ICDE)*, pages 7–18, 2006.

[SEKX98] J. Sander, M. Ester, H. P. Kriegel, and X. Xu. Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1998.

[SGF+07] F. M. Smith, W. M. Gallagher, E. Fox, R. B. Stephens, E. Rexhepaj, E. F. Petricoin 3rd, L. Liotta, M. J. Kennedy, and J. V. Reynolds. Combination of SELDI-TOF-MS and Data Mining Provides Early-Stage Response Prediction for Rectal Tumors Undergoing Multimodal Neoadjuvant Therapy. *Annals of Surgery*, 2(245):259–266, 2007.

[SGM00] A. Strehl, J. Ghosh, and R. Mooney. Impact of Similarity Measures on Web-page Clustering. In *Proc. AAAI Workshop on AI for Web Search*, pages 58–64, 2000.

[Sil86] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

[SJW97] K. Spark-Jones and P. Willet. *Readings in Information Retrieval*. Morgan Kaufmann, 1997.

[SL68] G. Salton and M. E. Lesk. Computer Evaluation of Indexing and Text Processing. *Journal of the ACM*, 15(1):8–36, 1968.

[Smy96] P. Smyth. Clustering Sequences with Hidden Markov Models. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 648–654, 1996.

[SN07] I. Sato and H. Nakagawa. Knowledge Discovery of Multiple-Topic Document using Parametric Mixture Model with Dirichlet Prior. In *Proc. ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 590–598. ACM, 2007.

[Spa73] K. Sparck Jones. Index Term Weighting. *Information Storage and Retrieval*, 9:619–633, 1973.

[SQL+03] J. Sander, X. Qin, Z. Lu, N. Niu, and A. Kovarsky. Automatic Extraction of Clusters from Hierarchical Clustering Representations. In *Proc. Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD)*, pages 75–87, 2003.

[SR62] R. R. Sokal and F. J. Rohlf. The Comparison of Dendrograms by Objective Methods. *Taxon*, 11:33–40, 1962.

[SS04] A. C. Sauve and T. P. Speed. Normalization, Baseline Correction and Alignment of High-Throughput Mass Spectrometry Data. In *Proc. Conf. on Genomic Signal Processing and Statistics*, 2004.

[SWY75] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[SYF05] Y. Sakurai, M. Yoshikawa, and C. Faloutsos. FTW: Fast Similarity Search under the Time Warping Distance. In *Proc. ACM Symposium on Principles of Database Systems (PODS)*, pages 326–337, 2005.

[SYY75] G. Salton, C. S. Yang, and C. T. Yu. A Theory of Term Importance in Automatic Text Analysis. *Journal of the Association for Information Systems*, 26(1):33–44, 1975.

[TA77] A. N. Tikhonov and V. Y. Arsenin. *Solution of Ill-posed Problems*. John Wiley & Sons, 1977.

[Tag05] A. Tagarelli. *Information and Knowledge Extraction form Semistructured Text Sources*. PhD thesis, System Engineering and Computer Science, 2005.

[Tai79] K. C. Tai. The Tree-to-Tree Correction Problem. *Journal of the ACM*, 26(3):422–433, 1979.

[TBMM04] H. S. Thompson, D. Beech, M. Maloney, and N. Mendelsohn. XML Schema Part 1: Structures Second Edition, 2004. W3C Recommendation, `http://www.w3.org/TR/2004/REC-xmlschema-1-20041028`.

[TG06] A. Tagarelli and S. Greco. Toward Semantic XML Clustering. In *Proc. SIAM Int. Conf. on Data Mining (SDM)*, pages 188–199, 2006.

[TG09] A. Tagarelli and S. Greco. Semantic Clustering of XML Documents. *ACM Transactions on Information Systems*, 2009. To appear.

[THD07] G. J. Tsekouras, N. D. Hatziargyriou, and E. N. Dialynas. Two-Stage Pattern Recognition of Load Curves for Classification and Electricity Customers. *IEEE Transactions on Power Systems*, 22(3):1120–1128, 2007.

[TSS04] N. F. Thornhill, M. A. A. Shoukat Choudhury, and S. L. Shah. The Impact of Compression on Data-Driven Process Analyses. *Process Control*, 14(4):389–398, 2004.

[TSTK08] G. J. Tsekouras, A. D. Salis, M. A. Tsaroucha, and I. S. Karanasiou. Load Time-Series Classification Based on Pattern Recognition Methods. In Peng-Yeng Yin, editor, *Pattern Recognition Techniques, Technology and Applications*, pages 361–432. IN-TECH, 2008.

[TXC07] Y. Tao, X. Xiao, and R. Cheng. Range Search on Multidimensional Uncertain Data. *ACM Transactions on Database Systems (TODS)*, 32(3):15–62, 2007.

[UCCA09] M. Ullman-Cullere, E. Clark, and S. Aronson. Implications of Genomics for Clinical Informatics. In *Encyclopedia of Database Systems*, pages 1400–1404. 2009.

[US02] N. Ueda and K. Saito. Parametric Mixture Models for Multi-Labeled Text. In *Proc. Annual Conf. on Neural Information Processing Systems (NIPS)*, pages 721–728, 2002.

[VFGL05] A. M. Vercoustre, M. Fegas, S. Gul, and Y. Lechevallier. A Flexible Structured-based Representation for XML Document Mining. In *INitiative for the Evaluation of XML Retrieval (INEX) Workshop*, pages 443–457, 2005.

[VGK02] M. Vlachos, D. Gunopulos, and G. Kollios. Discovering Similar Multidimensional Trajectories. In *Proc. Int. Conf. on Data Engineering (ICDE)*, pages 673–684, 2002.

[vR79] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, 1979.

[VTM09] G. Valentini, R. Tagliaferri, and F. Masulli. Computational Intelligence and Machine Learning in Bioinformatics. *Artificial Intelligence in Medicine*, 45(2–3):91–96, 2009.

[WAA00] Y. Wu, D. Agrawal, and A. Abbadi. A Comparison of DFT and DWT Based Similarity Search in Time-Series Databases. In *Proc. ACM Int. Conf. on Information and Knowledge Management (CIKM)*, pages 488–495, 2000.

[War05] T. Warren Liao. Clustering of Time Series Data—A Survey. *Pattern Recognition*, 38:1857–1874, 2005.

[WCC05] J. W. H. Wong, G. Cagney, and H. M. Cartwright. SpecAlign - Processing and Alignment of Mass Spectra Datasets. *Bioinformatics*, 21(9):2088–2090, 2005.

[WCD+05] B. Williams, S. Cornett, B. M. Dawant, A. Crecelius, B. Bodenheimer, and R. Caprioli. An Algorithm for Baseline Correction of MALDI Mass Spectra. In *Proc. Annual ACM Southeast Regional Conference (SE)*, pages 137–142, 2005.

[WF74] R. A. Wagner and M. J. Fischer. The String-to-String Correction Problem. *Journal of the ACM*, 21(1):168–173, 1974.

[WH09] M. Weinstein and D. Horn. Dynamic quantum clustering: a method for visual exploration of structures in data. *Computing Research Repository (CoRR)*, abs/0908.2644, 2009.

[Wil97] Y. Wilks. Information Extraction as a Core Language Technology. In *Summer School on Information Extraction (SCIE)*, pages 1–9. Springer, 1997.

[WKG04] W. E. Wallace, A. J. Kearsley, and C. M. Guttman. An Automated Peak Identification/Calibration Procedure for High-Dimensional Protein Measures From Mass Spectrometers. *Analytical Chemistry*, 76(9):2446–2452, 2004.

[WNP03] M. Wagner, D. Naik, and A. Pothen. Protocols for Disease Classification from Mass Spectrometry Data. *Proteomics*, 3(9):1692–1698, 2003.

[WSR+98] M. B. W. Wolfe, M. E. Schreiner, B. Rehder, D. Laham, P. W. Foltz, W. Kintsch, and T. K. Landauer. Learning from Text: Matching Readers and Texts by Latent Semantic Analysis. *Discourse Processes*, 25(2/3):309–336, 1998.

[WZL00] X. Wang, X. Zhou, and S. Lu. Spatiotemporal Data Modeling and Management: A Survey. In *Proc. Int. Conf. on Technology of Object-Oriented Languages and Systems (TOOLS)*, pages 202–211. IEEE Computer Society, 2000.

[XCCH02] F. Xiaodong, C. Changling, L. Changling, and S. Huihe. An Improved Process Data Compression Algorithm. In *Proc. Congress on Intelligent Control and Automation (WCICA)*, pages 2190–2193, 2002.

[XEKS98] X. Xu, M. Ester, H. P. Kriegel, and J. Sander. A Distribution-Based Clustering Algorithm for Mining in Large Spatial Databases. In *Proc. Int. Conf. on Data Engineering ICDE*, pages 324–331, 1998.

[XH07] L. Xiao and E. Hung. An Efficient Distance Calculation Method for Uncertain Objects. In *Proc. IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 10–17, 2007.

[YF00] B. K. Yi and C. Faloutsos. Fast Time Sequence Indexing for Arbitrary Lp Norms. In *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pages 385–394, 2000.

[YJF98] B. K. Yi, H. V. Jagadish, and C. Faloutsos. Efficient Retrieval of Similar Time Sequences Under Time Warping. In *Proc. Int. Conf. on Data Engineering (ICDE)*, pages 201–208, 1998.

[YMA+03] Y. Yasui, D. McLerran, B. L. Adam, M. Winget, M. Thornquist, and Z. Feng. An Operator-Independent Approach to Mass Spectral Peak Identification and Integration. *Journal of Biomedicine and Biotechnology*, 4:242–248, 2003.

[YRCK01] J. P. Yoon, V. Raghavan, V. Chakilam, and L. Kerschberg. BitCube: A Three-Dimensional Bitmap Indexing for XML Documents. *Journal of Intelligent Information Systems*, 17(1):241–252, 2001.

[Zad65] L. A. Zadeh. Fuzzy Sets. *Information and Control*, 8(3):338–353, 1965.

[ZG03] S. Zhong and J. Ghosh. A Unified Framework for Model-Based Clustering. *Journal of Machine Learning Research*, 4:1001–1037, 2003.

[ZG05] S. Zhong and J. Ghosh. Generative Model-Based Document Clustering: a Comparative Study. *Knowledge and Information Systems*, 8(3):374–384, 2005.

[Zip49] G. K. Zipf. *Human Behavior and the Principle of Least Effort: an Introduction to Human Ecology.* Addison-Wesley, 1949.

[ZK04] Y. Zhao and G. Karypis. Soft Clustering Criterion Functions for Partitional Document Clustering: a Summary of Results. In *Proc. ACM International Conference on Information and Knowledge Management (CIKM)*, pages 246–247, 2004.

[ZLPZ08] W. Zhang, X. Lin, J. Pei, and Y. Zhang. Managing Uncertain Data: Probabilistic Approaches. In *Proc. Int. Conf. on Web-Age Information Management (WAIM)*, 2008.