Studies in Computational Intelligence 527

Fabrice Guillet
Bruno Pinaud
Gilles Venturini
Djamel Abdelkader Zighed *Editors*

Advances in Knowledge Discovery and Management

Volume 4

Springer

Studies in Computational Intelligence

Volume 527

Series Editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland
e-mail: kacprzyk@ibspan.waw.pl

For further volumes:
http://www.springer.com/series/7092

About this Series

The series "Studies in Computational Intelligence" (SCI) publishes new developments and advances in the various areas of computational intelligence—quickly and with a high quality. The intent is to cover the theory, applications, and design methods of computational intelligence, as embedded in the fields of engineering, computer science, physics and life sciences, as well as the methodologies behind them. The series contains monographs, lecture notes and edited volumes in computational intelligence spanning the areas of neural networks, connectionist systems, genetic algorithms, evolutionary computation, artificial intelligence, cellular automata, self-organizing systems, soft computing, fuzzy systems, and hybrid intelligent systems. Of particular value to both the contributors and the readership are the short publication timeframe and the world-wide distribution, which enable both wide and rapid dissemination of research output.

Fabrice Guillet · Bruno Pinaud
Gilles Venturini · Djamel Abdelkader Zighed
Editors

Advances in Knowledge Discovery and Management

Volume 4

Springer

Editors
Fabrice Guillet
LINA
University of Nantes
France

Bruno Pinaud
LaBRI
University of Bordeaux
France

Gilles Venturini
LI
University François Rabelais of Tours
Tours
France

Djamel Abdelkader Zighed
ERIC
University Lumière of Lyon 2
Bron
France

ISSN 1860-949X ISSN 1860-9503 (electronic)
ISBN 978-3-319-02998-6 ISBN 978-3-319-02999-3 (eBook)
DOI 10.1007/978-3-319-02999-3
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013952936

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The recent and novel research contributions collected in this book are extended and reworked versions of a selection of the best papers that were originally presented in French at the EGC'2012 Conference held in Bordeaux, France, on January 2012. These 9 best papers have been selected from the 29 papers accepted in long format at the conference. These 29 long papers were themselves the result of a peer and blind review process among the 117 papers initially submitted to the conference in 2012 (acceptance rate of 26% for long papers). This conference was the 12^{th} edition of this event, which takes place each year and which is now successful and well-known in the French-speaking community. This community was structured in 2003 by the foundation of the International French-speaking EGC society (EGC in French stands for "Extraction et Gestion des Connaissances" and means "Knowledge Discovery and Management", or KDM). This society organizes every year its main conference (about 200 attendees) but also workshops and other events with the aim of promoting exchanges between researchers and companies concerned with KDM and its applications in business, administration, industry or public organizations. For more details about the EGC society, please consult `http://www.egc.asso.fr`.

Structure of the Book

This book is a collection of representative and novel works done in Data Mining, Knowledge Discovery, Clustering and Classification. It is intended to be read by all researchers interested in these fields, including PhD or MSc students, and researchers from public or private laboratories. It concerns both theoretical and practical aspects of KDM.

The first chapters of this book are related to Knowledge Discovery and Data Mining. Several authors are dealing with the clustering of data. The chapter of F. Queyroi (Chap. 1, page 3) deals with the partitioning of graphs. In the chapter of M. Boullé et al. (Chap. 2, page 15), functional data clustering is studied. The chapter of F de A. T. de Carvalho et al. (Chap. 3, page 37) deals with the clustering of relational data. Three other chapters are related to the mining of frequent sequences

(A. Ben Zakour et al., Chap. 4, page 53), to event extraction from text using SVMs (R. Faiz et al., Chap. 5, page 77), and to discretizing numerical variables for multi-relational data mining (D. Lahbib et al. Chap. 6, page 95).

The three remaining chapters are related to classification and feature extraction or feature selection. In the chapter of H. Chouaib et al. (Chap. 7, page 113), a feature selection approach is proposed using evolutionary algorithms. In the chapter of L. Vézard et al. (Chap. 8, page 133), an evolutionary algorithm is also used to select features and to solve a EEG signal classification task. In the paper of T.-N. Doan and F. Poulet (Chap. 9, page 155) a SVM is used to classify large sets of images.

Acknowledgments

The editors would like to thank the chapter authors for their insights and contributions to this book.

The editors would also like to acknowledge the members of the review committee and the associated referees for their involvement in the review process of the book. Their in depth reviewing, criticisms and constructive remarks have significantly contributed to the high quality of the slected papers.

Finally, we thank Springer and the publishing team, and especially T. Ditzinger and J. Kacprzyk, for their confidence in our project.

Nantes, Bordeaux, Tours, Lyon — *Fabrice Guillet, Bruno Pinaud*
July 2013 — *Gilles Venturini, Djamel Abdelkader Zighed*

Organization

Review Committee

All published chapters have been reviewed by 2 or 3 referees and at least one non-french speaking referee (2 for most papers).

Sadok Ben Yahia	Univ. of Tunis, Tunisia
Paula Brito	Univ. of Porto, Portugal
Francisco de A. T. De Carvalho	Univ. Federal de Pernambuco, Brazil
Gilles Falquet	Univ. of Geneva, Switzerland
Carlos Ferreira	LIAAD INESC Porto LA, Portugal
Fabien Gandon	INRIA, France
Philippe Lenca	Telecom Bretagne, France
Antonio Irpino	Second University of Naples, Italy
Robert Hilderman	Univ. of Regina, Canada
Donato Malerba	Dipartimento di Informatica, Universita' di Bari, Italy
Engelbert Mephu Nguifo	Univ. Clermont-Ferrand 2, France
Francesco Palumbo	Univ. of Naples Federico II, Italy
Jian Pei	Simon Fraser Univ., Canada
Jan Rauch	Univ. of Prague, Czech Republic
Lorenza Saitta	Univ. of Torino, Italy
Ansaf Salleb-Aouissi	Columbia Univ., USA
Gilbert Saporta	CNAM Paris, France
Stefan Trausan-Matu	Univ. of Bucharest, Romania
Rosanna Verde	Univ. of Naples 2, Italy
George Vouros	Univ. of Piraeus, Greece
Jef Wijsen	Univ. of Mons-Hainaut, Belgium

Associated Reviewers

Antonio Balzanella
Marc Boulé
Thierry Despeyroux
Rim Faiz
Gaelle Loosli
Sofian Maabout
François Poulet
Guanting Tang
Ronan Tournier

Contents

List of Contributors

Maha Amami is currently a Ph.D. student in Computer Science at the Higher Institute of Management of Tunis (ISG). Her research interests include Natural Language Processing, Text Mining, Social Information Retrieval.

Asma Ben Zakour is a PhD graduate from the university of Bordeaux I. She is presently a research and development engineer at 2MoRO solutions. Her research interest concerns sequential patterns extraction.

Marc Boullé was born in 1965 and graduated from Ecole Polytechnique (France) in 1987 and Sup Telecom Paris in 1989. Currently, he is a Senior Researcher in the data mining research group of Orange Labs. His main research interests include statistical data analysis, data mining, especially data preparation and modelling for large databases. He developed regularized methods for feature preprocessing, feature selection and construction, correlation analysis, model averaging of selective naive Bayes classifiers and regressors.

Francisco de A. T. de Carvalho is Full Professor at the "Centro de Informatica" of the "Universidade Federal de Pernambuco" (Brazil). His main research interests are symbolic data analysis and clustering analysis. He has authored over 190 technical papers in international journals and conferences. He has served in program committees of Brazilian and international conferences and He has also served as review of international journals. He was elected for the council (2009–2013) of the International Association for Statistical Computing (IASC).

Marie Chavent is Assistant Professor in Statistics at university of Bordeaux Segalen (France). She is member of the Probability and Statistics team at Mathematics Institute of Bordeaux (IMB, UMR CNRS 5251). She is also member of the CQFD research team of Inria Bordeaux Sud-Ouest. Her research interests are dimension reduction, clustering and data analysis.

Hassan Chouaib received his master degree in computer science in artificial intelligence, at Toulouse Paul Sabatier University (France) in 1997. He defended his PhD within the research group *Systèmes Intelligents de Perception* (SIP) at the *Laboratoire d'Informatique Paris Descartes* (LIPADE) in 2011 in Computer Science (Image Processing, Pattern Recognition, Computer Vision) from Paris Descartes University, France. His research is related to feature selection with applications in pattern recognition, image analysis and bioinformatics.

Julien Clauzel is a student in third year in Arts et Métiers school (Bordeaux, France). He did an internship in University of Bordeaux2 in 2011. During this internship, he helped in the acquisition of EEG signals and participated in the development of Matlab codes of genetic algorithms.

Florence Cloppet is assistant professor in computer science at Paris Descartes University (Paris-France) since 1997. She presently belongs to the research group *Systèmes Intelligents de Perception* (SIP) at the *Laboratoire d'Informatique Paris Descartes* (LIPADE). She received her Ph.D. degree in Computer Science (Image Processing, Pattern Recognition, Computer Vision) from René Descartes University, France, in 1996. Her research is related to computer vision and pattern recognition and more precisely to image analysis, extraction and selection of features. She is particularly interested in the use of knowledge in order to improve image analysis both in biomedical or document images.

Thierry Despeyroux is researcher at INRIA. He is a specialist in programming environments and semantics of programming languages and has been deputy leader of the Inria CROAP team for 10 years in Sophia-Antipolis. He then joined the Axis team in Rocquencourt and his research activity includes now data coherence and data mining.

Thanh-Nghi Doan received the M.S. degree in computer science from University of Science, Ho Chi Minh, Viet Nam, in 2003. He is currently a Ph.D. candidate in TEXMEX Research Team, Efficient Exploitation of Multimedia Documents Exploration, Indexing, Navigation, and Access to Very Large Databases, IRISA, France. He worked as a lecturer at An Giang University, Long Xuyen, Viet Nam from 1998 to 2010. His research is mainly focused on machine learning, data mining and high performance computing in computer vision, in particular large scale visual vlassification.

Aymen Elkhlifi graduated in Computer Science at the Higher Institute of Management of Tunis (ISG), and obtained a Ph.D in Computer Science at the University of Paris-Sorbonne, France. His main research interests include Artificial Intelligence, Machine Learning, Natural Language Processing, Information Extraction and Text Mining.

Frédérique Faïta-Aïnseba is assistant professor and researcher in Cognitive Sciences at the Bordeaux Ségalen University in France. She obtained her PhD in Neuroscience in Marseille (France) in 1995. After her graduation, she held a post-doctoral position at Montreal University (Canada). Since her arrival to Bordeaux, she dedicates her researches to the study of cognitive processes involve in language, reading and musical listening.

Rim Faiz obtained his Ph.D. in Computer Science from the University of Paris-Dauphine, in France. She is currently a Professor in Computer Science at the Institute of High Business Study (IHEC), University of Carthage, in Tunisia. Her research interests include Artificial Intelligence, Machine Learning, Natural Language Processing, Information Retrieval, Text Mining, and Semantic Web. She has published several papers and has served as PC member and reviewer for several international conferences and journals. Dr. Faiz is also responsible of the Master "E-Commerce" and Master "Business Intelligence" at IHEC, University of Carthage.

Romain Guigourès was born in 1987 and graduated from EISTI (École Internationale des Science du Traitement de l'Information) and Paris 13 University in 2005. He is currently a PhD candidate in Applied Mathematics at Paris 1 University and works for the data mining research group of Orange Labs. His main research interests include data mining, coclustering and exploratory data analysis.

Dhafer Lahbib was born in 1982 and graduated from The National School of computer Science (Tunisia) in 2006. He got a master Degree from the same school in 2007 and from Paris-sud XI University in 2009. In 2012, he received his doctoral degree from the Cergy-Pontoise University. Since december 2012 he has a post-doctoral position at IBISC laboratory at Evry-Val-d'Essonne University. His research interests include machine learning, multi-relational data mining, structured data mining, ...

Dominique Laurent graduated in 1978 in Mathematics from the University of Orléans (France). He received his doctoral degree in 1987 and then his Habilitation in 1994 from the University of Orléans. From 1987 to 1996, he was Assistant Professor in this same university, and in 1996, he joined the University of Tours as a Professor. Since 2003, he is Professor at University of Cergy-Pontoise (France), where he leads the Graduate school Science et Ingénierie. His research interests include database theory, data mining and data warehousing.

Yves Lechevallier is senior researcher at INRIA (French National Institute for research in computer science and control), within the Paris-Rocquencourt centre, and he belongs to the AxIS research team (Usage-centred design, analysis and improvement of information systems). His research activity mainly concerns data mining: numeric-symbolic methods, clustering, model trees, (multi-)relational data mining, and web usage mining. He has published more than 100 papers in international journals and conference proceedings and has participated to several European and

National research projects. He was a member of numerous committee program in international conferences and workshops on data mining.

Pierrick Legrand received his PhD in applied mathematics from "Ecole centrale de Nantes" and from Nantes university (France) in December 2004. In 2005 and 2006, he received two post-doctoral positions (Evovision group at CICESE research center (Ensenada, México) and INRIA COMPLEX Team (Rocquencourt, France). On September 2006, he became associate professor at the university of Bordeaux2 and researcher at the IMB (Institut de Mathématiques de Bordeaux, UMR CNRS 5251). He is also researcher in the INRIA ALEA team since 2010.

Filipe M. de Melo currently is a graduate student in computer science at Universidade Federal de Pernambuco, Brazil. His research interest is clustering analysis and related methods.

Sofian Maabout is assistant professor at the university of Bordeaux. He is affiliated with both LaBRI and INRIA team CEPAGE. His research activities concern mainly database performance, data mining and parallel algorithms.

Mohamed Mosbah is full Professor at "Institut Polytechnique de Bordeaux". He conducts his research within formal methodology team (LaBRI). His research interest concern Distributed computing, distributed systems, distributed algorithms, Algorithms for mobile and Ad hoc networks, Formal methods and security for distributed systems.

Francois Poulet is assistant Professor in computer science at the University of Rennes I - IRISA. His main research topics include: Information Visualization, Data Analytics, Classification Algorithms, Large Scale Data Mining Algorithms, Big Data.

François Queyroi is a PhD student in computer science and a graduate statistical engineer. He works at the LaBRI (University of Bordeaux, France) in the MaBioVis team. His research interests include network analysis, graph partionning and information visualization.

Fabrice Rossi, born in 1971, is Professor at University Paris 1 since September 2011. He is a former student of the E.N.S. (Ecole Normale Supérieure). He received a Ph.D. in applied mathematics from the Paris-IX Dauphine university in 1996. His research activities focus on data mining and machine learning, especially on methods that support visual exploration of the data, such as the Self Organizing Map. He works in particular on non vector data such as graphs and functional data.

Marc Sistiaga is operation manager at 2MoRO Solutions an aviation software editor where he was project manager and delivery manager. During 5 years, he was development designer at Capgemini for clients such as Airbus or CNES. He has a PhD degree on Image Processing at the LIRMM after spending three years in the Robotics and Positioning Service IFREMER.

Leonardo Trujillo Reyes received an Electronic Engineering (2002) and a Masters in Computer Science (2004) from the Technical Institute of Tijuana in México (ITT). He then received a Ph.D. in Computer Science from CICESE research center, Mexico (2008). He is currently professor at the ITT. Dr. Trujillo is involved in interdisciplinary research in the fields of Evolutionary Computation, Computer Vision, Image Analysis, Pattern Recognition and Autonomous Robotics.

Laurent Vézard is a PhD student at the third year. His advisors are Marie Chavent, Pierrick Legrand and Frédérique Faïta-Aïnseba. He is bio statistical Engineer by the National Institute of Applied Sciences of Toulouse, France (2010). He is also member of the CQFD team of Inria Bordeaux Sud-Ouest. His current main research interests are classification problem, dimension reduction and feature extraction from data using genetic algorithms.

Nicole Vincent is full Professor since 1996. She presently belongs to the research group *Systèmes Intelligents de Perception* (SIP) at the *Laboratoire d'Informatique Paris Descartes* (LIPADE) in the Paris 5 University. After studying in Ecole Normale Supérieure and graduation in Mathematics, Nicole Vincent received a Ph.D. in Computer Science in 1988 from Lyon Insa. She has been involved with several projects in pattern recognition, signal and image processing and video analysis. Her research interest concerns both knowledge and data representation as well as change detection, looking for the best representation space. The main application domains are document image analysis, image retrieval, video sequence analysis and biomedical images.

Part I
Knowledge Discovery and Data Mining

Optimizing a Hierarchical Community Structure of a Complex Network

François Queyroi

Abstract. Many graph clustering algorithms perform successive divisions or aggregations of subgraphs leading to a hierarchical decomposition of the network. An important question in this domain is to know if this hierarchy reflects the structure of the network or if it is only an artifice due to the conduct of the procedure. We propose a method to validate and, if necessary, to optimize the multi-scale decomposition produced by such methods. We apply our procedure to the algorithm proposed by Blondel *et al.* (2008) based on modularity maximization. In this context, a generalization of this quality measure in the multi-level case is introduced. We test our method on random graphs and real world examples.

1 Introduction

A central task of network analysis is the detection of a community structure [Cook and Holder, 2006]. Many graph clustering algorithms have been developed to fulfill this task (see [Fortunato, 2010] for a survey). These methods often rely on the maximization of a quality measure like modularity [Newman, 2006].

Previous works in human sciences [Simon, 1962; Pumain, 2006] suggest however the presence of a hierarchical structure in complex systems such as networks. Several strategies have been used to discover such hierarchies by iteratively grouping or splitting groups. Good examples are algorithms based on a similarity metric. At each iteration the two closest groups (in term of similarity) are merged leading to the construction of a hierarchy. However, the resulting hierarchy is barely relevant for an analyst because a level is the division of only one single group. In a recent paper [Pons and Latapy, 2010], Pons and Latapy provide a procedure to simplify this kind of structure.

François Queyroi
Universiy of Bordeaux, CNRS, LaBRI, France
e-mail: francois.queyroi@labri.fr

F. Guillet et al. (eds.), *Advances in Knowledge Discovery and Management,*
Studies in Computational Intelligence 527,
DOI: 10.1007/978-3-319-02999-3_1,

Other algorithms directly lead to workable hierarchies [Lancichinetti et al., 2011; Rosvall and Bergstrom, 2011]. Blondel *et al.* [Blondel et al., 2008] introduced a flat clustering procedure that relies on the construction of a hierarchy of clusters. The identification of a hierarchy is not the final objective of the algorithm although many clues suggest that this hierarchy is meaningful to analyse the structure of the studied network.

This paper focuses on the validation of such hierarchies. We provide an optimization procedure allowing to filter out undesirable fusion of clusters. We enforce this post-procedure on the results produced by the algorithm of Blondel *et al.*[Blondel et al., 2008]. For this purpose we introduce a generalisation of the modularity quality measure in order to quantify the quality of a hierarchical clustering.

The rest of the paper is organized as follows. In section 2, we introduce the approach used to evaluate the quality of a hierarchical community structure. In section 3, we provide an application of our approach to the algorithm of Blondel *et al.*. In section 4, we show that our procedure is efficient by describing some results obtained on a hierarchically clustered graph benchmark and on real world examples. We compare our results to those produced by two different state-of-the-art procedures[Lancichinetti et al., 2011; Rosvall and Bergstrom, 2011].

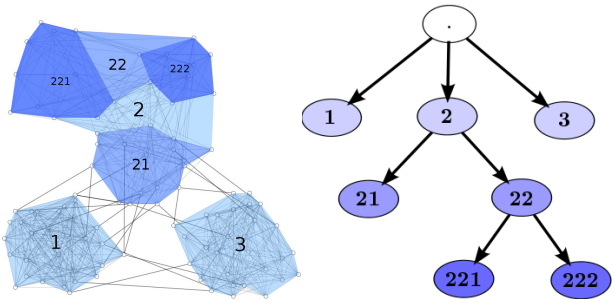
2 Optimizing a Hierarchical Community Structure

2.1 Definitions

Given a graph $G = (V,E)$ where V is the set of vertices and E the set of edges. A flat clustering of G is a partition of the vertices V in several groups (also called *communities* when they are densely connected) defining a set of induced subgraphs of G. In the example provided in Figure 1a, the vertices falling into the hulls labelled $\{1,2,3\}$ correspond to three subgraphs. A hierarchical clustering appears when some of these communities are recursively divided into subgroups. For example, the subgraph labelled 2 is divided into two subgraphs 21 and 22. The nesting between groups of vertices at different levels makes trees an efficient way to model hierarchical clusterings (see Figure 1b). We call *clustering trees* such structures.

Let T be a clustering tree of the vertices set V. It is a rooted tree where each node $t \in T$ can be either an *internal* node if its degree $d(t) \geq 2$ or a *leaf* node if $d(t) = 1$. The set of leaves of T is denoted $\mathscr{F}(T)$. In the previous example we have $\mathscr{F}(T) = \{1,21,221,222,3\}$. Each node $t \in T$ corresponds to a subset $V_t \subset V$. Let $p(t)$ be the direct ancestor of t and $\sigma(t)$ the set of direct successors of t. In the example, we have $p(22) = 2$ and $\sigma(22) = \{221,222\}$. These relations correspond to the following constraints: for each node t, we have $V_t \subseteq V_{p(t)}$ and $V_t = \bigcup_{c\in\sigma(t)} V_c$ if t is *internal*.

We denote by T_t the subtree of T rooted in t and by G_t the subgraph induced by the vertices set V_t. In the example, G_1 is a graph which contains the vertices that fall into the hull labelled 1 and the edges having both extremities in the same set. The *height* of a node t in T is the number of edges between the root of T and t.



(a) A hierarchical clustered graph (b) A labelled clustering tree

Fig. 1 An example of hierarchical clustering of a graph (left) modelled using a clustering tree (right)

We denote by $N_i(T)$ the i-th level of T which is the set of *leaves* in the subtree $T \setminus \{t \in T, h(t) > i\}$. In the example given in Figure 1 we have $N_1(T) = \{1,2,3\}$, $N_2(T) = \{1,21,22,3\}$ et $N_3(T) = \mathscr{F}(T)$. Each level $N_i(T)$ of T is a flat clustering of the set V.

2.2 Evaluating a Hierarchical Community Structure

To identify a community structure in a network, *quality measures* are often used in order to compare different flat clusterings of a graph. A *quality measure* Φ is a function having as domain the set of all flat clusterings and as range a real interval. Evaluating the quality of a hierarchical clustering is far more problematic because we have to take the nesting and the height of the clusters into account. To fulfill this task, Blanc *et al.* [Blanc et al., 2010] introduced a recursively defined measure that generalized the Mancoridis criteria[Mancoridis et al., 1998] to hierarchical clusterings. The same idea is used here for all measures respecting the additivity constraint [Pons and Latapy, 2010].

Definition 1. A **quality measure** $\Phi(G,C)$ of a flat clustering $C = (C_1, \ldots, C_k)$ for the set of vertices V of a G is said to be **additive** if it can be written

$$\Phi(G,C) = \sum_{i=1}^{k} \phi(G,C_i) \tag{1}$$

where the function $\phi(G,C_i) \in [0, \frac{1}{k}]$ is called the *gain* of the community i.

Most of the existing quality measures are additive[Pons and Latapy, 2010]. The idea underlying the extension of quality measures to hierarchical clusterings

is the recursive call of an additive quality measure on each internal node of the clustering tree.

Definition 2. Given $\Phi(G,C)$ an additive quality measure, its extension to a hierarchical clustering tree T rooted in r is denoted $\Phi(G,T;q)$ and is defined as follows:

$$\Phi(G,T;q) = \begin{cases} \sum_{t\in\sigma(r)} \phi(G,V_t)\,(1+q\times\Phi(G_t,T_t;q)) & \text{if } \sigma(r)>0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

for $q \in [0,1]$.

The measure $\Phi(G,T;q)$ is a polynomial with a variable $q \in [0,1]$. On one hand, the weight of an internal node at the bottom of the hierarchy increases when q is close to 1. On the other hand, we have $\Phi(G,T;q) = \Phi(G,N_1(T))$ for $q = 0$.

Note that the quality of a community (a node of T) is weighted by the product of the quality of its ancestors. This weight corresponds to the idea that a badly defined community (with an external density greater than its internal density for example) can only generate badly defined sub communities (see [Blanc et al., 2010] for further details).

Definition 3. We denote by **hierarchical quality index** of a clustering tree T, the function $\Phi(G,T)$ which is the integral of the polynomial $\Phi(G,T;q)$ for $q \in [0,1]$:

$$\Phi(G,T) = \int_0^1 \Phi(G,T;q)dq \tag{3}$$

The value of q to use is an open issue. When there is no reason to promote or penalize deep hierarchies, the criteria $\Phi(G,T)$ shall be used.

2.3 Hierarchy Quality Optimization

The formula 2 and 3 can be used to compare different hierarchical clusterings and select the best one for a given network. Therefore we are able to access the relevance of a modification applied on a hierarchical clustering. We can for example determine whether or not a given node in the clustering tree should be removed. The removal of a node t is the replacement of t by its successors $\sigma(t)$. We denote as $\Delta_t(T) = \Phi(G,T\setminus\{t\}) - \Phi(G,T)$ the quality variation due to this modification. Given an initial clustering tree T, our optimization procedure can be defined as the iterative suppression of an internal node t (if it exists) maximizing $\Delta_t(T)$ with $\Delta_t(T) > 0$.

The removal of a node $t \in T$ results in several modifications in the multilevel quality measure computation. First, the weights of all nodes in the subtree T_t are greater because the depth of the clusters this subtree contains are now smaller in T. Secondly, the nodes of the set $\sigma(t)$ do not longer correspond to a flat clustering of the subgraph G_t but are new parts of the flat clustering of the subgraph $G_{p(t)}$. Looking at the previous example in Figure 1, after the deletion of the node 22, the nodes 221 and 222 are now direct successors of the node 2. Therefore, the number

of edges leaving 221 and 222 increases because the edges between these clusters and the cluster 21 are added.

The complexity of our procedure is $O(|T|^3)$ where $|T|$ is the number of nodes in the clustering tree T. First, we assume that the gain function ϕ can be computed in constant time. This can be achieved by keeping some information into memory (the number of internal/external edges for example). Secondly, the function Φ is computed in $O(|T|)$ as a simple depth-first search over the clustering tree. Finally, the procedure described above lies in the family of greedy algorithms.

3 Application to Modularity Maximization

In this section, we present the algorithm of Blondel *et al.*[Blondel et al., 2008] which produces a hierarchical clustering of a graph. We then illustrate the fact that the hierarchies produced may contain some irrelevant groups. These observations justify the use of our method.

3.1 Algorithm Description

The algorithm of Blondel *et al.* is a modularity maximization heuristic. The modularity can be defined as follows:

$$Q(G,C) = \sum_{t=1}^{k} \frac{e_t}{M} - \left(\frac{d_t}{2M}\right)^2 \tag{4}$$

where e_t is the number of edges having both ends in the cluster t, d_t is the sum of the degrees of nodes belonging to the cluster t and M is the number of edges in G. We can easily prove that $Q(G,C)$ is additive. The gain $\phi(G,V_t)$ is here the difference between the observed proportion of internal edges in V_t and its theoretical value in a random graph with the same degree distribution.

At the beginning of the algorithm, each vertex corresponds to a single community. The algorithm has two major phases. First, we seek for each vertex the communities that lie in its direct neighbourhood and compute the potential increase of modularity resulting of assigning the vertex to each of them. The vertex is then assigned to the community that maximize the gain (ties are broken randomly). This phase is repeated as long as an increase of the modularity is possible and results in a flat clustering of the graph. Secondly, we replace the previous graph by the quotient graph computed using the previous clustering. These two phases are iteratively repeated as long as the modularity increases.

3.2 Discussion on the Resulting Hierarchy

The algorithm of Blondel *et al.* produces a hierarchy T by iteratively applying a flat clustering procedure (the first phase described above) to the quotient graph created at the previous iteration. Each level of T can be seen as a local maximum of the

modularity. The authors suggest that the last level found $N_1(T)$ is the most meaningful since it corresponds to the highest modularity reached.

This algorithm is very popular in social network analysis because it can be applied on very large graphs while providing clusterings with high modularity values. We can however hardly determine whether or not the hierarchy is meaningful to analysis a given network. We provide here two major issues.

First, the hierarchy may contain irrelevant intermediate clusters. Note that the first phase of the algorithm is nondeterministic because the resulting flat clustering depends on the order in which the vertices are taken. We illustrate this issue using the example given in Figure 1. The first iteration of the algorithm leads to the detection of the communities $\{1,21,221,222,3\}$ (the last level of the final hierarchy). At the second iteration the communities 221 and 222 are grouped leaving the others isolated even if grouping 221, 222 and 21 would lead to a greater modularity. Looking at the final clustering tree, we could say that the cluster 22 is just a building step and is therefore irrelevant.

Secondly, the direct optimization of modularity can lead to the excessive aggregation of several communities. This issue is called the *resolution limit* (see a description in [Fortunato and Barthélemy, 2007] and experimental illustrations in [Good et al., 2010]). Blondel *et al.* actually discussed the fact that the hierarchy can be seen as an alternative to this issue. The excessive aggregations occur at the first levels of the hierarchy. While the flat modularity gain may be small (but still positive) remember that the multilevel quality measure we provide takes the whole hierarchy into account. Top level clusters can therefore be removed if their contribution is not strong enough to justify an additional level.

These issues illustrate the usefulness of our method when applied to the hierarchy produced by the algorithm described in this section. We therefore use the procedure described in Section 2.3 using in Eq. 3

$$\phi(G,V_t) = \frac{e_t}{e_{p(t)}} - \left(\frac{d_t}{2e_{p(t)}}\right)^2 \tag{5}$$

as the *gain* of the community indexed by t in T having $p(t)$ as direct ancestor.

4 Results

In this section, we discuss the results of several experiments. First, we show that our procedure is able to detect irrelevant intermediate clusters using benchmark graphs where a two level hierarchical clustering is known. Secondly, we provide results of our procedure when it is applied on real world networks. These results seem reasonable when compared to other state-of-the-art hierarchical clustering algorithms.

4.1 Validation on Random Graphs with a Known Hierarchical Clustering

In order to validate our method we use the LFR-*benchmark* extended to hierarchical clustering [Lancichinetti and Radicchi, 2008; Lancichinetti et al., 2011]. This benchmark is used in order to evaluate the effectiveness of clustering algorithms (see [Rosvall and Bergstrom, 2011] for example).

We generate graphs with a power law degree distribution and a two-level community structure. These levels are denoted *micro*-communities and *macro*-communities. We can decide how well the communities are defined (in term of density). This is achieved by using two parameters μ_1 and μ_2 which correspond to the proportion of edges between *macro*-communities and the proportion of edges between *micro*-communities lying in the same *macro*-communities respectively. The graphs have 10000 vertices with an average degree of 20 and a maximum degree of 100. The size of *macro*-communities and *micro*-communities are in the range $[400, 4000]$ and $[10, 100]$ respectively.

We evaluate how well the given multilevel structure is identified by using the normalized mutual information [Danon et al., 2005]. This measure is used to access the similarity between two partitions of the same set. The result is a score between 0 (the partitions are completely different) and 1 (the partitions are the same).

The results are given in Figure 2. The x-axis corresponds to the value $\mu_1 + \mu_2$ which is the proportion of edges outside *micro*-communities. For four different values of μ_1, we compare the different clusterings for $\mu_2 \in [\mu_1, 1 - \mu_1]$. The y-axis corresponds to the normalized mutual information between the compared clusterings. We compare the real *micro*-communities to the clustering given by $N_2(T)$ and $\mathscr{F}(T)$ (orange and red curves respectively) and the real *macro*-communities to the clustering given by $N_1(T)$ (blue curves). The results reported here correspond to an average on one hundred samples.

First, we analyze the results obtained using the algorithm without optimization (left column in Figure 2). The *micro*-communities seem to be identified when they are well defined theoretically. This situation occurs when $\mu_2 < 0.5$. *Macro*-communities are also identified when the proportion of edges between *micro*-communities is superior to the proportion of edges between *micro*-communities. We can however observe that the clustering trees produced by the algorithm contain additional levels. Indeed the flat clustering $N_2(T)$ should be equal to $\mathscr{F}(T)$ (the *micro*-communities) but it is obviously not always the case here. This last observation confirms the risks outlined in Section 3.2.

Looking now at the results obtained using our method (right column in the Figure 2), we can see that the intermediate levels are removed and that the flat clustering $N_2(T)$ is almost always equal to the *micro*-communities. Moreover, the similarities between $N_1(T)$/*macro*-communities and between $\mathscr{F}(T)$/*micro*-communities do not change. It means that we do not remove wrongly some clusters.

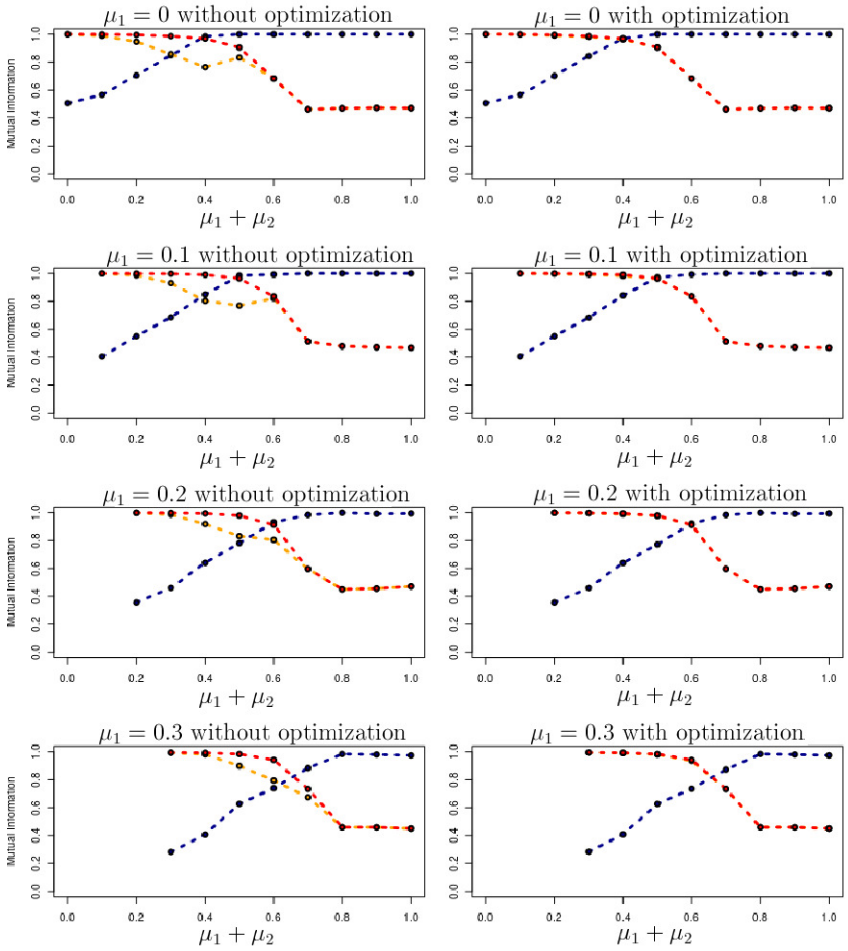


Fig. 2 Evaluation on the multilevel LFR Benchmark for different values of μ_1 and μ_2. The blue line corresponds to the mutual information between $N_1(T)$ and the real *macro*-communities. The red one between $\mathscr{F}(T)$ and the real *micro*-communities. Finally, the orange one between $N_2(T)$ and the real *micro*-communities.

4.2 Real World Examples

We now present some results of our method when applied to real world networks. We compare our resulting hierarchical clustering to the hierarchical clusterings produced by the *Oslom* [Lancichinetti et al., 2011] and *Infomap* [Rosvall and Bergstrom, 2011] algorithms. Note that these algorithms are also nondeterministic.

4.2.1 Co-publication Network

We first look at a co-publication network in social network analysis (see [Fortunato, 2010] for more details). The graph contains 515 authors (vertices), two authors are linked when they are co-authors in at least one paper. The graph contains 1318 edges.

This kind of network is conductive to the presence of some hierarchical community structure. Indeed, the top level of such hierarchy could correspond to people in the same university/institute while the bottom level could correspond to groups formed by Professors/Ph.D. students.

The *Oslom* and *Infomap* algorithms detect big clusters at the top level (with over a hundred people). While these clusters can be easily separated from the rest of the network by removing a couple of edges, they are not densely connected. In particular they contain a lot of biconnected components.

The results obtained using the algorithm of [Blondel et al., 2008] without our method provides similar results. Using our procedure, the first level is removed and contains subgraphs with a small graph diameter that may correspond to close collaboration within same research teams. A visualization of the results is given in Figure 3. The clusterings $N_1(T)$ and $N_2(T)$ are drawn using blue and grey concave hulls respectively.

4.2.2 Migration Network

We now investigate the hierarchical structure which can be found in a migration network. The graph models migration flows in USA (see [Cui et al., 2008] for details on this dataset). The 1650 vertices represent American counties. For each couple of counties we know the number of person who moved from one to the other between 1995 and 2000. There is a total of 6500 positive relations in this network which are represented as weighted directed edges.

A visualisation of the results is provided in Figure 4 where counties are geolocalized. The two first hierarchical levels are drawn using a color mapping. Both levels illustrate the following observation: geographically close counties are more likely to be part of the same clusters. This observation can be also found in the results obtained using *Oslom* and *Infomap* algorithms.

The *Infomap* algorithm does not find any hierarchical structure in this network. The biggest identified communities correspond to California, Texas and the East of the country. On the opposite, the *Oslom* algorithm provides a deep hierarchical clustering tree. The first level contains mostly two very big communities corresponding to the West/Mid-West area and the East. These clusters are then divided over two additional levels. The bottom clustering corresponds to the clustering provided by the *Infomap* algorithm.

The results of our method are a good compromise. The first level (see Figure 4a) contains relatively large clusters. The second level (see Figure 4b) is similar to the clustering provided by the *Infomap* algorithm.

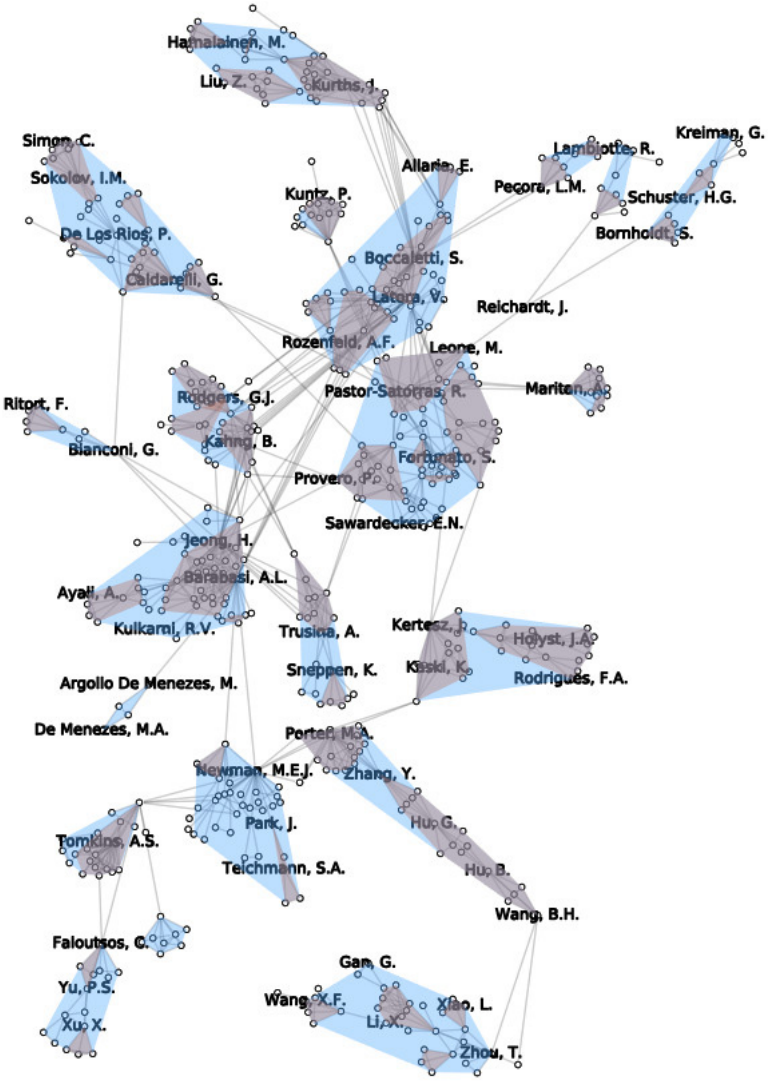
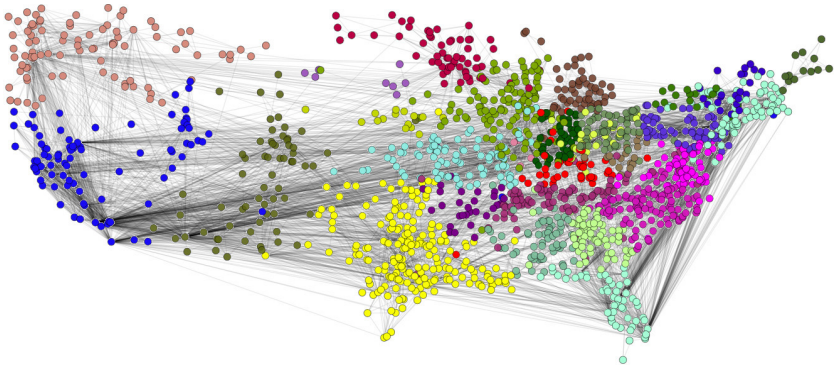


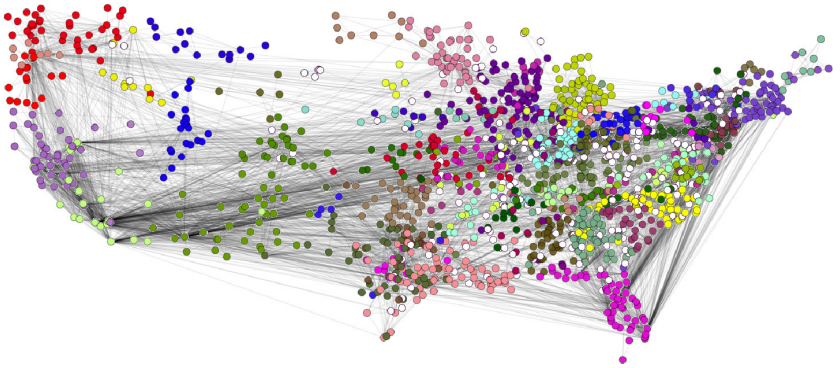
Fig. 3 Results on a scientific co-publication network. The blue and grey hulls correspond to the flat clusterings $N_1(T)$ and $N_2(T)$ respectively.

5 Conclusion

We introduced a post-processing procedure to improve the quality of a hierarchical clustering of a network. This is achieved by iteratively removing the internal clusters that decrease a multilevel quality measure. Our method was applied to the algorithm of Blondel *et al.* [Blondel et al., 2008] by using a generalization of the modularity metric to hierarchical clusterings. The experiments run on random graphs clearly



(a) First level $N_1(T)$



(b) Second level $N_2(T)$

Fig. 4 Results on a migration network (United-States). Vertices position corresponds to the geographical coordinates of the corresponding county. Two vertices belong to the same cluster if they have the same color. White coloured vertices are isolated (cluster of size 1).

show that the hierarchies we provide are very close to the ground truth hierarchies. Results obtained on real networks are also meaningful.

Note that our method does not allow to know whether or not the leaves of the clustering tree should be removed. We can reduce this problem to the following one: is a given clustering better than no clustering at all? One way to overcome this problem is to use a minimal threshold for modularity. However dealing with this kind of clusters is less problematic. Indeed, from an analysis perspective, the first levels of a hierarchical clustering are the most relevant.

As future work, we plan to test the effectiveness of our post-processing procedure when applied with different hierarchical clustering algorithms. The greedy removal of internal clusters is a fast and intuitive method but adding internal clusters to the hierarchy is also a possible modification. We need to investigate the way of

combining these basic operations to explore the space of hierarchical clusterings using a hierarchical quality measure as objective function.

References

[Blanc et al., 2010] Blanc, C., Delest, M., Fédou, J.-M., Mélançon, G., Queyroi, F.: Évaluer la qualité d'une fragmentation de graphe multi-niveaux. In: Journées MARAMI 2010, Toulouse, France (2010)

[Blondel et al., 2008] Blondel, V.D., Guillaume, J.-L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment, P10008 (2008)

[Cook and Holder, 2006] Cook, D.J., Holder, L.B.: Mining Graph Data. Wiley (2006)

[Cui et al., 2008] Cui, W., Zhou, H., Qu, H., Wong, P., Li, X.: Geometry-based edge clustering for graph visualization. IEEE Transactions on Visualization and Computer Graphics 14(6), 1277–1284 (2008)

[Danon et al., 2005] Danon, L., Diaz-Guilera, A., Duch, J., Arenas, A.: Comparing community structure identification. Journal of Statistical Mechanics: Theory and Experiment, P09008 (2005)

[Fortunato, 2010] Fortunato, S.: Community detection in graphs. Physics Reports 486(3-5), 75–174 (2010)

[Fortunato and Barthélemy, 2007] Fortunato, S., Barthélemy, M.: Resolution limit in community detection. Proceedings of the National Academy of Sciences 104(1), 36 (2007)

[Good et al., 2010] Good, B., De Montjoye, Y., Clauset, A.: Performance of modularity maximization in practical contexts. Physical Review E 81(4), 46106 (2010)

[Lancichinetti and Radicchi, 2008] Lancichinetti, A., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Physical Review E 78(4), 046110 (2008)

[Lancichinetti et al., 2011] Lancichinetti, A., Radicchi, F., Ramasco, J.: Finding statistically significant communities in networks. PloS One 6(4), e18961 (2011)

[Mancoridis et al., 1998] Mancoridis, S., Mitchell, B., Rorres, C., Chen, Y., Gansner, E.: Using automatic clustering to produce high-level system organizations of source code. In: Proceedings of the 6th International Workshop on Program Comprehension, pp. 45–52. IEEE (1998)

[Newman, 2006] Newman, M.E.J.: Modularity and community structure in networks. Proceedings of the National Academy of Sciences, USA 103, 8577–8582 (2006)

[Pons and Latapy, 2010] Pons, P., Latapy, M.: Post-processing hierarchical community structures: Quality improvements and multi-scale view. Theoretical Computer Science 412, 892–900 (2010)

[Pumain, 2006] Pumain, D. (ed.): Hierarchy in Natural and Social Sciences. Methodos Series, vol. 3. Springer (2006)

[Rosvall and Bergstrom, 2011] Rosvall, M., Bergstrom, C.: Multilevel compression of random walks on networks reveals hierarchical organization in large integrated systems. PloS One 6(4), e18209 (2011)

[Simon, 1962] Simon, H.: The architecture of complexity. Proceedings of the American Philosophical Society 106(6), 467–482 (1962)

Nonparametric Hierarchical Clustering of Functional Data

Marc Boullé, Romain Guigourès, and Fabrice Rossi

Abstract. In this paper, we deal with the problem of curves clustering. We propose a nonparametric method which partitions the curves into clusters and discretizes the dimensions of the curve points into intervals. The cross-product of these partitions forms a data-grid which is obtained using a Bayesian model selection approach while making no assumptions regarding the curves. Finally, a post-processing technique, aiming at reducing the number of clusters in order to improve the interpretability of the clustering, is proposed. It consists in optimally merging the clusters step by step, which corresponds to an agglomerative hierarchical classification whose dissimilarity measure is the variation of the criterion. Interestingly this measure is none other than the sum of the Kullback-Leibler divergences between clusters distributions before and after the merges. The practical interest of the approach for functional data exploratory analysis is presented and compared with an alternative approach on an artificial and a real world data set.

1 Introduction

In functional data analysis (FDA [Ramsay and Silverman, 2005]), observations are functions (or curves). Each function is sampled at possibly different evaluation

Marc Boullé · Romain Guigourès
Orange Labs
2 av. Pierre Marzin
22300 Lannion, France
e-mail: {marc.boulle,romain.guigoures}@orange.com

Romain Guigourès · Fabrice Rossi
SAMM EA 4543, Université Paris 1
90 rue de Tolbiac
75013 Paris, France
e-mail: romain.guigoures@malix.univ-paris1.fr,
fabrice.rossi@univ-paris1.fr

F. Guillet et al. (eds.), *Advances in Knowledge Discovery and Management,*
Studies in Computational Intelligence 527,
DOI: 10.1007/978-3-319-02999-3_2,

points, leading to variable-length sets of pairs (evaluation point, function value). Functional data arise in many domains, such as daily records of precipitation at a weather station or hardware monitoring where each curve is a time series related to a physical quantity recorded at a specified sampling rate.

Exploratory analysis methods for large functional data sets are needed in practical applications such as e.g. electric consumption monitoring [Hébrail et al., 2010]. They reduce data complexity by combining clustering techniques with function approximation methods, representing a functional data set by a small set of piecewise constant prototypes. In this type of approach, both the number of prototypes and the number of segments (constant parts of the prototypes) are under user control. On a positive side, this limits the risk of cognitive overwhelming as the user can ask for a low complexity representation. Unfortunately, this can also induce under/over-fitting of the model to the data; additionally the number of prototypes and the number of segments both need to be tuned, while they can be adjusted independently in [Hébrail et al., 2010], increasing the risk of over/under-fitting. Other parametric approaches for function clustering and/or function approximation can be found in e.g. [Cadez et al., 2000; Chamroukhi et al., 2010], [Gaffney and Smyth, 2004], [Ramsay and Silverman, 2005]. All those methods make (sometimes implicit) assumptions on the distribution of the functions and/or on the measurement noise.

Nonparametric functional approaches (e.g. [Ferraty and Vieu, 2006]) have been proposed, in particular in [Gasser et al., 1998; Delaigle and Hall, 2010], where the problem of density estimation of a random function is considered. However, those models do not tackle directly the summarizing problem outlined in [Hébrail et al., 2010] and recalled above. Nonparametric Bayesian approaches based on Dirichlet process have also been applied to the problem of curves clustering. They aim at inferring a clustering distribution on an infinite mixture model [Nguyen and Gelfand, 2011; Teh, 2010]. The clustering model is obtained by sampling the posterior distribution using Bayesian inference methods.

The present paper proposes a new nonparametric exploratory method for functional data, based on data grid models [Boullé, 2010]. The method makes assumption neither on the functional data distribution nor on the measurement noise. Given a set of sampled functions defined on a common interval $[a,b]$, with values in $[u,v]$, the method outputs a clustering of the functions associated to partitions of $[a,b]$ and $[u,v]$ in sub-intervals which can be used to summarize the values taken by the functions in each cluster, leading to results comparable to those of [Hébrail et al., 2010]. Both approaches are for that matter compared in this article.

The method has no parameters and obtains in a fully automated way an optimal summary of the functional data set, using a Bayesian approach with data dependent priors. In some cases, especially for large scale data sets, the optimal number of clusters and of sub-intervals may be too large for a user to interpret all the discovered fine grained patterns in a reasonable time. Therefore, the method is complemented with a post-processing step which offers the user a way to decrease the number of clusters in a greedy optimal way. The number of sub-intervals, that is the level of

details kept in the functions, is automatically adjusted in an optimal way when the number of clusters is reduced.

The post-processing technique consists in merging successively the clusters in the least costly way, from the finest clustering model to one single cluster containing all the curves. It appears that the cost of the merge of two clusters is a weighted sum of Kullback-Leibler divergences from the merged clusters to the created cluster which can be interpreted as a dissimilarity measure between the two clusters that have been merged. Thus, the post-processing technique can be considered as an agglomerative hierarchical clustering [Hastie et al., 2001]. Decision-making tools can be plotted using a dendrogram and a Pareto chart of the criterion value as a function of the number of clusters.

The rest of the paper is organized as follows. Section 2 introduces the problem of curves clustering and relates our method to alternative approaches. Next, in Section 3, the clustering method based on joint density estimation is introduced. Then, the post-processing technique is detailed in section 4. In Section 5 the results of experimentations on an artificial data set and on a power consumption data set are shown. Finally Section 6 gives a summary.

2 Functional Data Exploratory Analysis

In this section, we describe in formal terms the data under analysis and the goals of the analysis.

Let $\mathscr{C}$ be a collection of n functions or curves, $c_i, 1 \leq i \leq n$, defined from $[a,b]$ to $[u,v]$, two real intervals. Each curve is sampled at m_i values in $[a,b]$, leading to a series of observations denoted $c_i = (x_{ij}, y_{ij})_{j=1}^{m_i}$, with $y_{ij} = c_i(x_{ij})$.

As in all data exploratory settings, our main goal is to reduce the complexity of the data set and to discover patterns in the data. We are therefore interested in finding clusters of similar functions as well as in finding functional patterns, that is systematic and simple regular shapes in individual functions. In [Chamroukhi et al., 2010; Hébrail et al., 2010] functional patterns are simple functions such as interval indicator functions or polynomial functions of low degree: a function is approximated by a linear combination of such simple functions in [Hébrail et al., 2010] or generated by a logistic switching process based on low degree polynomial functions in [Chamroukhi et al., 2010]. B-splines could also be used as in [Abraham et al., 2003] but with no simplification virtues.

Let us denote k_C the number of curve clusters. Given k_C classes $\mathscr{F}_k$ of "simple functions" used to discover functional patterns (e.g., piecewise constant functions with P segments), the method proposed in [Hébrail et al., 2010] finds a partition $(\mathscr{C}_k)_{k=1}^{k_C}$ of $\mathscr{C}$ and k_C simple functions $(f_k \in \mathscr{F}_k)_{k=1}^{k_C}$ which aim at minimizing

$$\sum_{k=1}^{k_C} \sum_{c_i \in \mathscr{C}_k} \sum_{j=1}^{m_i} (y_{ij} - f_k(x_{ij}))^2 , \tag{1}$$

which corresponds to a form of K-means constrained by the choice of the segments, in the functional space L^2. The approach of [Chamroukhi et al., 2010] optimizes a similar criterion obtained from a maximum likelihood estimation of the parameters of the functional generative model.

Given a specific choice of the simple function classes, the functional prototypes $(f_k)_{k=1}^{k_C}$ obtained by [Chamroukhi et al., 2010; Hébrail et al., 2010] induce k_C partitions of $[a,b]$ into sub-intervals on which functions are roughly constant. Those partitions are the main tool used by the analyst to understand the functional pattern inside each cluster. The general abstract goal of functional data exploration is therefore to build clusters of similar functions associated to sub-intervals of the input space of the functions which summarize the behavior of the functions.

Bayesian Approaches, as described in [Nguyen and Gelfand, 2011], assume that the collection of curves realizations can be represented by a set of canonical curves drawn from a Gaussian Process and organized into clusters. The clusters are described using a label function that is a realization of a multinomial distribution with a Dirichlet prior. Whereas parametric models using a fixed and finite number of parameters may suffer from over- or under-fitting, Bayesian nonparametric approaches were proposed to overcome these issues. By using a model with an unbounded complexity, underfitting is mitigated, while the Bayesian approach of computing or approximating the full posterior over parameters lessens over-fitting [Teh, 2010]. Finally, the parameters distribution is obtained by sampling the posterior distribution using Bayesian inference methods such as Markov Chain Monte Carlo [Neal, 2000] or Variational Inference [Blei and Jordan, 2005]. Then a post-treatment is required for the choice of the clustering parameters among their distribution.

The Dirichlet Process prior requires two parameters: a concentration parameter and a base distribution. For a concentration parameter α and a data set containing n curves, the expected number of clusters $\bar{k}$ is $\bar{k} = \alpha \log(n)$ [Wallach et al., 2010]. Hence, the concentration parameter has a significant impact on the obtained number of clusters. For that matter, according to [Vogt et al., 2010], one should not expect to be able to reliably estimate this parameter.

Our method - named MODL and detailed in Section 3 - is comparable to approaches based on Dirichlet process (DP) in so much as all estimate a posterior probability based on the likelihood and a prior distribution of the parameters. The methods are also nonparametric with an unbounded complexity, since the number of parameters is not fixed and grows with the amount of available data.

Nevertheless, MODL is intrinsically different from the DP based methods. First, approaches based on DP are Bayesian and yield a distribution of clusterings, the final clustering being selected using a post-treatment like chosing the mode of the posterior distribution or by studying the clusters co-occurence matrix. By contrast, MODL is a MAP approach, the most probable model is directly obtained using optimization algorithms. Secondly, MODL is not applied on the values but on the order statistics of the sample. One first benefit is to avoid outliers or scaling problems. By using order statistics, the retrieved models are invariant by any monotonic transformation of the input data, which makes sense since the method aims at modeling the correlations between the variables, not the values directly. Then, DP based methods

consider distributions of the parameters that lie in $\mathbb{R}$ or any continuous space, which measure is consequently infinite. As for MODL, the correlations between the variables are modeled on a sample. In the case of curves clustering, these variables are the location X, the corresponding curve realization Y, and the curve label C. This allows to work on a finite discrete space and thus to simplify the model computation, that mainly comes down to counting problems. Finally, the MODL approach is clearly data dependant. In a first phase, the data sample is used cautiously to build the model space and the prior: only the size of the sample and the values (or empirical ranks) of each variable taken independently are exploited. The correlation model is inferred in a second phase, using a standard MAP approach. Hence, proving the consistency of this data dependant modeling technique is still an open issue. Actually, experimental results with both reliable and fine grained retrieved patterns show the relevancy of the approach.

3 MODL Approach for Functional Data Analysis

In this section, we summarize the principles of data grid models, detailed in [Boullé, 2010], and apply this approach on the functional data.

3.1 Data Grid Models

Data grid models [Boullé, 2010] have been introduced for the data preparation phase of the data mining process [Chapman et al., 2000], which is a key phase, both time consuming and critical for the quality of the results. They allow to automatically, rapidly and reliably evaluate the class conditional probability of any subset of variables in supervised learning and the joint probability in unsupervised learning. Data grid models are based on a partitioning of each variable into intervals in the numerical case and into groups of values in the categorical case. The cross-product of the univariate partitions forms a multivariate partition of the representation space into a set of cells. This multivariate partition, called data grid, is a piecewise constant nonparametric estimator of the conditional or joint probability. The best data grid is searched using a Bayesian model selection approach and efficient combinatorial algorithms.

3.2 Application to Functional Data

We propose to represent the collection $\mathscr{C}$ of n curves as a unique data set with $m = \sum_{i=1}^{n} m_i$ observations and three variables, C to store the curve identifier, X and Y for the point coordinates. We can apply the data grid models in the unsupervised setting to estimate the joint density $p(C,X,Y)$ between the three variable. The curve variable C is grouped into clusters of curves, whereas each point dimension X and Y is discretized into intervals. The cross-product of these univariate partitions forms a data grid of cells, whith a peacewise constant joint density estimation per triplet

of curve cluster, X interval and Y interval. As $p(X,Y|C) = \frac{p(C,X,Y)}{p(C)}$, this can also be interpreted as an estimator of the joint density between the point dimensions, which is constant per cluster of curves. This means that similar curves with respect to the joint density of their point dimensions will tend to be grouped into the same clusters. It is noteworthy that the (X,Y) discretization is optimized globally for the set of all curves and not locally per cluster as in [Hébrail et al., 2010].

We introduce in Definition 1 a family of functional data clustering models, based on clusters of curves, intervals for each point dimension, and a multinomial distribution of all the points on the cells of the resulting data grid.

Definition 1. A functional data clustering model is defined by:

- a number of clusters of curves,
- a number of intervals for each point dimension,
- the repartition of the curves into the clusters of curves,
- the distribution of the points of the functional data set on the cells of the data grid,
- the distribution of the points belonging to each cluster on the curves of the cluster.

Notation.

- $\mathscr{C}$: *collection of curves, size* $n = |\mathscr{C}|$.
- $\mathscr{P}$: *point data set containing all points of* $\mathscr{C}$ *using 3 variables, size* $m = |\mathscr{P}|$.
- C: *curve variable*
- X,Y: *variables for the point dimensions*
- k_C: *number of clusters of curves*
- k_X, k_Y: *number of intervals for variables* X *and* Y
- $k = k_C k_X k_Y$: *number of cells of the data grid*
- n_{i_C}: *number of curves in cluster* i_C
- m_i: *number of points for curve* i
- m_{i_C}: *cumulated number of points for curves of cluster* i_C
- m_{j_X}, m_{j_Y}: *cumulated number of points for intervals* j_X *of* X *and* j_Y *of* Y
- $m_{i_C j_X j_Y}$: *cumulated number of points for cell* (i_C, j_X, j_Y) *of the data grid*

We assume that the numbers of curves n and points m are known in advance and we aim at modeling the joint distribution of the m points on the curve and the point dimensions. In order to select the best model, we apply a Bayesian approach, using the prior distribution on the model parameters described in Definition 2.

Definition 2. The prior for the parameters of a functional data clustering model are chosen hierarchically and uniformly at each level:

- the numbers of clusters k_C and of intervals k_X, k_Y are independent from each other, and uniformly distributed between 1 and n for the curves, between 1 and m for the point dimensions,
- for a given number k_C of clusters, every partitions of the n curves into k_C clusters are equiprobable,

- for a model of size (k_C, k_X, k_Y), every distributions of the m points on the $k = k_C k_X k_Y$ cells of the data grid are equiprobable,
- for a given cluster of curves, every distributions of the points in the cluster on the curves of the cluster are equiprobable,
- for a given interval of X (resp. Y), every distributions of the ranks of the X (resp. Y) values of points are equiprobable.

Taking the negative log of the posterior probability of a model given the data, this provides the evaluation criterion given in Theorem 1, which specializes to functional data clustering the unsupervised data grid model general criterion [Boullé, 2010].

Theorem 1. *A functional data clustering model M distributed according to a uniform hierarchical prior is Bayes optimal if the value of the following criteria is minimal*

$$\begin{aligned} c(M) &= -log(P(M)) - log(P(\mathscr{P}|M)) \\ &= \log n + 2\log m + \log B(n,k_C) \\ &\quad + \log\binom{m+k-1}{k-1} + \sum_{i_C=1}^{k_C} \log\binom{m_{i_C}+n_{i_C}-1}{n_{i_C}-1} \\ &\quad + \log m! - \sum_{i_C=1}^{k_C}\sum_{j_X=1}^{k_X}\sum_{j_Y=1}^{k_Y} \log m_{i_C j_X j_Y}! \\ &\quad + \sum_{i_C=1}^{k_C} \log m_{i_C}! - \sum_{i=1}^{n} \log m_i! \\ &\quad + \sum_{j_X=1}^{k_X} \log m_{j_X}! + \sum_{j_Y=1}^{k_Y} \log m_{j_Y}! \end{aligned} \tag{2}$$

$B(n,k)$ is the number of divisions of n elements into k subsets (with eventually empty subsets). When $n = k$, $B(n,k)$ is the Bell number. In the general case, $B(n,k)$ can be written as $B(n,k) = \sum_{i=1}^{k} S(n,i)$, where $S(n,i)$ is the Stirling number of the second kind [Abramowitz and Stegun, 1970], which stands for the number of ways of partitioning a set of n elements into i nonempty subsets.

As negative log of probabilities are coding lengths, the model selection technique is similar to a minimum description length approach [Rissanen, 1978]. The first line in Formula 2 relates to the prior distribution of the numbers of cluster k_C and of intervals k_X and k_Y, and to the specification of the partition of the curves into clusters. The second line represents the specification of the parameters of the multinomial distribution of the m points on the k cells of the data grid, followed by the specification of the multinomial distribution of the points of each cluster on the curves of the cluster. The third line stands for the likelihood of the distribution of the points on the cells, by the mean of a multinomial term. The last line corresponds to the likelihood of the distribution of the points of each cluster on the curves of the cluster, followed by the likelihood of the distribution of the ranks of the X values (resp. Y values) in each interval.

Algorithm 1: Greedy Bottom Up Merge Heuristic

Require: M (initial solution)
Ensure : M^*; $c(M^*) \leq c(M)$
$M^* \leftarrow M$;
while *solution is improved* **do**
  $M' \leftarrow M^*$;
  forall the *merge u between 2 clusters or adjacent intervals of X or Y* **do**
    $M^+ \leftarrow M^* + u$;
    if $c(M^+) < c(M')$ **then**
      $M' \leftarrow M^+$;
    end if
  end forall
  if $c(M') < c(M^*)$ **then**
    $M^* \leftarrow M'$ (improved solution);
  end if
end while

3.3 Optimization Algorithm

The optimization heuristics have practical scaling properties, with $O(m)$ space complexity and $O(m\sqrt{m}\log m)$ time complexity. The main heuristic is a greedy bottom-up heuristic, which starts with a fine grained model, with a few points per interval on X and Y and a few curves per cluster, considers all the merges between clusters and adjacent intervals, and performs the best merge if the criterion decreases after the merge, as detailed in Algorithm 1.

This heuristic is enhanced with post-optimization steps (moves of interval bounds and of curves across clusters), and embedded into the variable neighborhood search (VNS) meta-heuristic [Hansen and Mladenovic, 2001], which mainly benefits from multiple runs of the algorithm with different initial random solutions.

The optimization algorithms summarized above have been extensively evaluated in [Boullé, 2010], using a large variety of artificial data sets, where the true data distribution is known. Overall, the method is both resilient to noise and able to detect complex fine grained patterns. It is able to approximate any data distribution, provided that there are enough instances in the train data sample.

4 Agglomerative Hierarchical Clustering

The model carried out by the method detailed in the section 3 is optimal according to the criterion introduced in Theorem 1. This parameter-free solution allows to track fine and relevant patterns without over-fitting. This provides a suitable initial solution to lead an exploratory analysis. Still, this initial solution may be too fine for an easy interpretation. We propose here a post-processing technique which aims at simplifying the clustering while minimizing the loss of information. This allows to

explore the retrieved patterns at any granularity, up to the finest model, without any user parameter.

We first study the impact of a merge on the criterion, then focus on the properties of the proposed dissimilarity measure and finally describe the agglomerative hierarchical clustering heuristic. It is noteworthy than the same modeling criterion is optimized both for building the initial clustering and for aggregating the clusters in the agglomerative heuristic.

4.1 The Cost of Merging Two Clusters

Let $M_{1_C,2_C}$ and M_{γ_C} be two clustering models, the first one is the model before the merge of the clusters 1_C and 2_C, the second one is the model after the merge, that yields a new cluster $\gamma_C = 1_C \cup 2_C$. We denote $\Delta c(1_C, 2_C)$ the cost of the merge of 1_C and 2_C, defined as:

$$\Delta c(1_C, 2_C) = c(M_{\gamma_C}) - c(M_{1_C,2_C})$$

It results from Theorem 1 that the clustering model M_{γ_C} is a less probable MODL explanation of the data set $\mathscr{P}$ than $M_{1_C,2_C}$ according to a factor based on $\Delta c(1_C, 2_C)$.

$$p(M_{\gamma_C}|\mathscr{P}) = e^{-\Delta c(1_C,2_C)} p(M_{1_C,2_C}|\mathscr{P}) \tag{3}$$

We focus on the asymptotic behavior of $\Delta c(1_C, 2_C)$ when the number of data points m tends to infinity.

Theorem 2. *The criterion variation is asymptotically equal to a weighted sum of the Kullback-Leibler divergences from the clusters 1_C and 2_C to γ_C, estimated on the $k_X \times k_Y$ bivariate discretization.*

$$\Delta c(1_C, 2_C) = m_{1_C} D_{KL}(1_C||\gamma_C) + m_{2_C} D_{KL}(2_C||\gamma_C) + O(log(m_{\gamma_C})) \tag{4}$$

Proof. The full proof is left out for brevity. Mainly, the computation of $\Delta c(1_C, 2_C)$ makes some prior terms (2 first lines of Formula 2) vanish and bounds the other ones by $O(log(m_{\gamma_C}))$ terms. Then, using the Stirling approximation $log(m!) = m(log(m) - 1) + O(log(m))$, the variation of the likelihood (the two last lines of Formula 2) can be rewritten as a weighted sum of Kullback-Leibler divergences. □

4.2 The Cost of a Merge as a Dissimilarity Measure

As the criterion defined in Theorem 1 is used to find the best model, we naturally chose it to evaluate the quality of the clustering. When two clusters are merged, the criterion decreases and its resulting variation can be viewed as dissimilarity between both clusters. When the number of points tends to infinity, the dissimilarity measure asymptotically converges to a weighted sum of Kullback-Leibler divergence

(see Theorem 2). This divergence is a non symmetric measure of the difference between two distributions [Cover and Thomas, 1991]. The variation of the criterion Δc has some interesting properties. First, it is symmetrical, $\Delta(1_C, 2_C) = \Delta(2_C, 1_C)$. Then, $\Delta c(1_C, 2_C)$ is asymptotically non-negative since the Kullback-Leibler divergence is also [Cover and Thomas, 1991]. The weights have an important impact on the merge in the case of unbalanced clusters. A trade-off is achieved between merging two balanced clusters with similar distributions and merging two different clusters, one of them having a tiny weight. The best merge is the one with the least loss of information, as $c(M)$ can be interpreted as the total coding length of the clustering model plus the data points given the model.

4.3 The Agglomerative Hierarchical Classification

The principle of the agglomerative clustering is to merge successively the clusters in order to build a tree called dendrogram. The usual dissimilarity measures for the dendrogram are based on Euclidean distances (Single-Linkage, Complete-Linkage, Ward, ...). Here we build a dendrogram using the criterion variation Δc. Due to the properties of this dissimilarity measure, the resulting dendrogram is well-balanced. Indeed, given the trade-off between merging similarly distributed clusters and merging tiny with large clusters, we obtain clusters with comparable sizes at each level of hierarchy.

Let us notice that during the agglomerative process, the best merge can relate either to the cluster variable C or to the points dimensions X or Y. Therefore, the granularity of the representation of the curves coarsens as the number of clusters decreases. As a consequence, the dissimilarity measure between two clusters of a partition "coarsens" together with the coarsening of the other partitions. This makes sense since fewer clusters in the partition need a less discriminative similarity measure to be distinguished. It is noteworthy that during the agglomerative process, partitions are coarsened but not re-optimized by locally moving the bounds of the intervals. Although this may be sub-optimal, this allows to ease the exploratory analysis by using the same family of nested intervals at any model granularity.

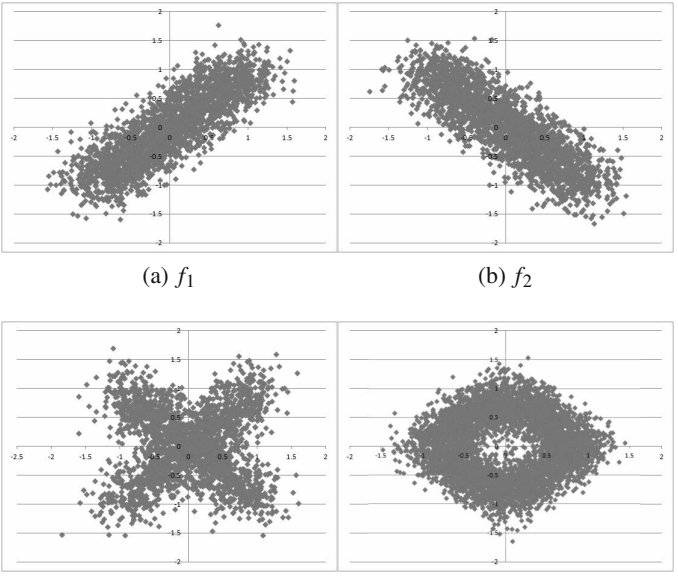
5 Experiments

In this section, we first highlight properties of our approach using an artificial data set and then apply it on a real-life data set, next we successively merge the clusters and finally show what kind of exploratory analysis can be performed.

5.1 Experiments on an Artificial Data Set

A variable z is sampled from an uniform distribution: $Z \sim \mathcal{U}(-1,1)$. ε_i denotes a white Gaussian noise: $E \sim \mathcal{N}(0, 0.25)$. Let us consider the four following distributions:

- $f_1 : x = z + \varepsilon_x,\ y = z + \varepsilon_y$
- $f_2 : x = z + \varepsilon_x,\ y = -z + \varepsilon_y$
- $f_3 : x = z + \varepsilon_x,\ y = \alpha z + \varepsilon_y$
  with $\alpha \in \{-1, 1\}$
  and $p(\alpha = -1) = p(\alpha = 1)$
- $f_4 : x = (0.75 + \varepsilon_x) cos(\pi(1 + z)),$
  $y = (0.75 + \varepsilon_y) sin(\pi(1 + z))$



(a) f_1 (b) f_2

(c) f_3 (d) f_4

Fig. 1 Artificially generated distributions

We generate a collection of 40 curves using each distribution defined previously (10 curves per distribution). We generate a data set $\mathscr{P}$ of 10^5 points. Each point is a triple of values with a randomly chosen curve (among 40), a x and a y value generated according to the distribution related to the curve.

We apply our functional data clustering method introduced in Section 3 on subsets of $\mathscr{P}$ of increasing sizes. The experiment is running 10 times per subset of points that are resampled each time. The graph on Figure 2 displays the average number of clusters and the number of X and Y intervals for a given number of points m. For very small subsets (below 400 data points), there are not enough data to discover significant patterns, and our method produces one single cluster of curves, with one single interval for the X and Y variables. From 400 data points, the numbers of clusters and intervals start to grow. Finally with only 25 points per curve

on average, that is 1000 points in the whole point data set, our method recovers the underlying pattern and produces four clusters of curves related to the f_1, f_2, f_3 and f_4 distributions.

Despite the method retrieved the actual number of clusters, below 2000 data points, the clusters may not be totally pure and some curves misplaced into clusters. In our experiments, for 1000 data points, 2% of the curves are misplaced on average, while for 2000 points, all the curves are systematically placed in their actual cluster.

It is noteworthy that by growing the size of the subset beyond 2000 data points, the number of retrieved patterns is constant and equal to four. By contrast, the number of intervals grows with the number of data points. This shows the good asymptotic behaviour of the method: it retrieves the true number of patterns and exploits the growing number of data to better approximate the pattern shapes.

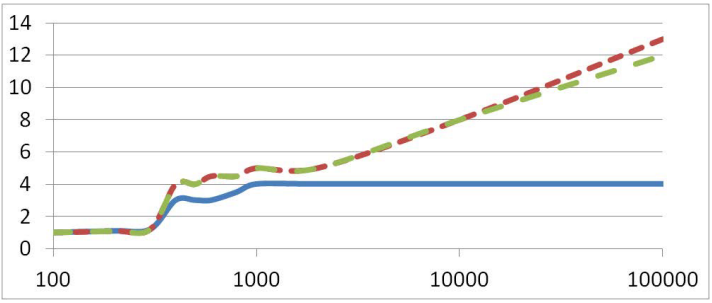


Fig. 2 Number of clusters (solid line), number of X intervals (tight dotted line) and number of Y intervals (spaced dotted line) for a given number of data points m

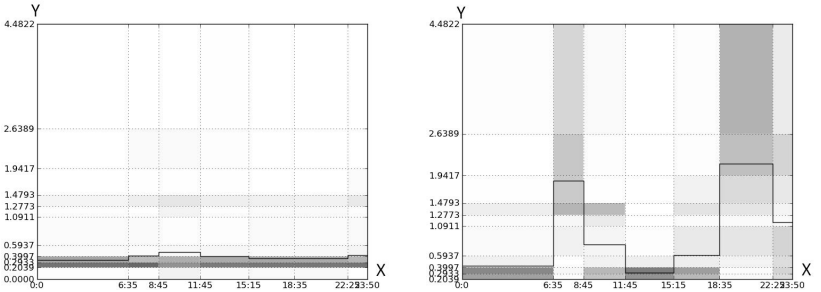
Regarding the results of the experiments on this data set, it is noteworthy that MODL does not require the same point locations for each curve. This may be an usefull property to make a clustering of functionnal data for which the measurement have not been recorded at regular intervals. Moreover, beyond the clustering of functional data, our method is able to deal with distributions. Thus, it is possible to detect clusters of multimodal distributions like the ones generated using f_3 and f_4.

5.2 *Analysis of a Power Consumption Data Set*

We use the data set [Hébrail et al., 2010] which consists in the electric power consumption recorded in a personal home during almost one year (349 days). Each curve consists in 144 measurements which give the power consumption of a day at a 10 minutes sampling rate. There are 50,256 data points and three features: the time of the measure X, the power measure Y and the day identifier C. The study of this data set aims at grouping the days according to the characteristic of the power

consumption of each day. First, the optimal model is computed using the MODL approach. Finally the approach is compared to that of [Hébrail et al., 2010].

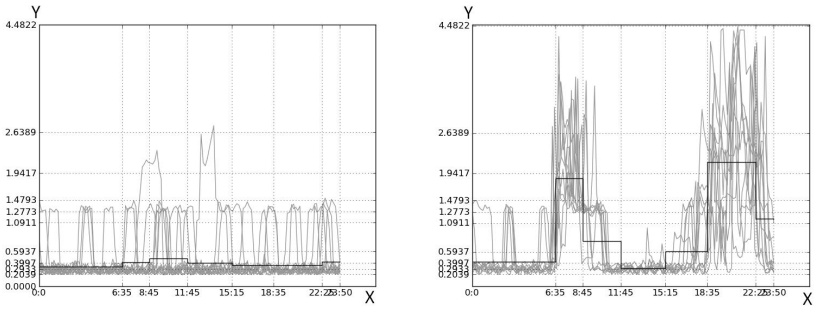
The MODL-Optimal Discretization. The optimal clustering consists in a data grid defined by 57 clusters, 7 intervals on X and 10 on Y. This means that the 349 recorded days have been grouped into 57 clusters, each day has been discretized into 7 time segments and the power measures into 10 power segments. This result highlights some characteristic days, such as the workdays, the days off or the days when nobody is at home. The summarized prototypes, represented by piecewise constant lines, show the average power consumption per time segment. The conditional probabilities of the power segments given the time segments are represented by grey cells, where the grey level shows the related conditional probability. The first representation has been chosen in order to simplify the reading of the curve, and the second to highlight some interesting phenomena such as the multimodal distributions of data points within the time segments.



(a) (b)

Fig. 3 Two examples among the 57 clusters, the plots display the summarized prototypes and the conditional probabilities represented by darkened cells. Figure (a) represents the largest cluster, typifying days where the power consumption is very low and almost constant; the residents were probably not at home. Figure (b), that is the second largest cluster, shows a workday with a low consumption during the night and the office hours, and with peaks in the morning and evening.

Multimodal distributions. In Figure 3.(b), we notice that the prototype is located between two dark cells for the third time segment. This means that the majority of the data points have been recorded in the higher and the lower power segments but rarely in the interval where the prototype is for this time segment. Thus, a multimodal distribution of the data points on this time segment is highlighted, which is confirmed by Figure 4.(b). Let us notice that 3.(a) is another illustration of a multimodal distribution for which the points are more frequent in the lower mode than in



(a) (b)

Fig. 4 Prototypes and stacked curves for the clusters of Figures 3 (a) and (b)

the upper one. Overall, the method extends the clustering of curves to clustering of distributions.

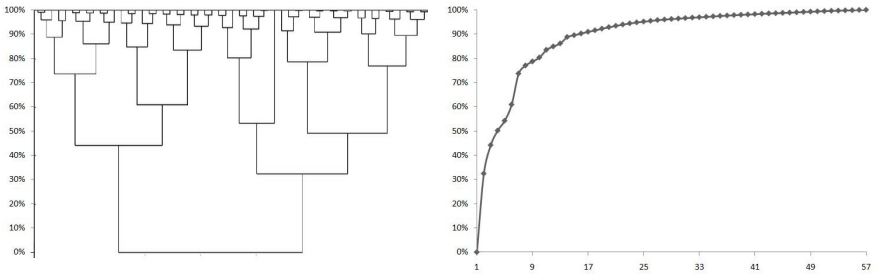
Merging the Clusters. Whereas the finest data grid yields a rich clustering and useful information for some characteristic clusters, a more synthetic and easily interpretable view of the power consumption over the year may be desirable in some applications. That is why agglomerative merges have been performed and represented on Figure 5 by a dendrogram and a Pareto chart presenting the percentage of kept information as a function of the number of clusters. This measure is defined as following:

Definition 3. Let $M_\emptyset$ be the null model with one cluster of curves and one interval per point dimension, whose data grid consists in one cell containing all the points. Its properties are detailed in [Boullé, 2010]. We denote M_{opt} the optimal model according to the optimization of the criterion defined in the Theorem 1 and M_k the model resulting from successive merges until obtaining k clusters. The percentage of kept information for k clusters τ_k is defined as:

$$\tau_k = \frac{c(M_k) - c(M_\emptyset)}{c(M_{opt}) - c(M_\emptyset)}$$

The dendrogram is well-balanced and the Pareto chart is concave, which allows to divide by three the number of clusters while keeping almost 90% of the initial information.

Comparative analysis of the modeling results. In order to highlight the differences between the results retrieved using MODL and the approach of [Hébrail et al., 2010], we propose to study a simplified data grid by coarsening the optimal model until having four clusters, using the post-processing technique detailed in Section 4. By doing this, 50% of the information is kept and the power consumption and the time discretizations are reduced to four intervals. Contrary to MODL, the approach



(a) Dendrogram (b) Pareto chart

Fig. 5 Dendrogram and Pareto chart of kept information per number of clusters

of [Hébrail et al., 2010] requires the user to specify the number of clusters and time segments. We applied therefore their clustering technique with four clusters and a total of sixteen time intervals that are optimally distributed over the four clusters. The clusters retrieved by both approaches are displayed in Figures 6 and 7.

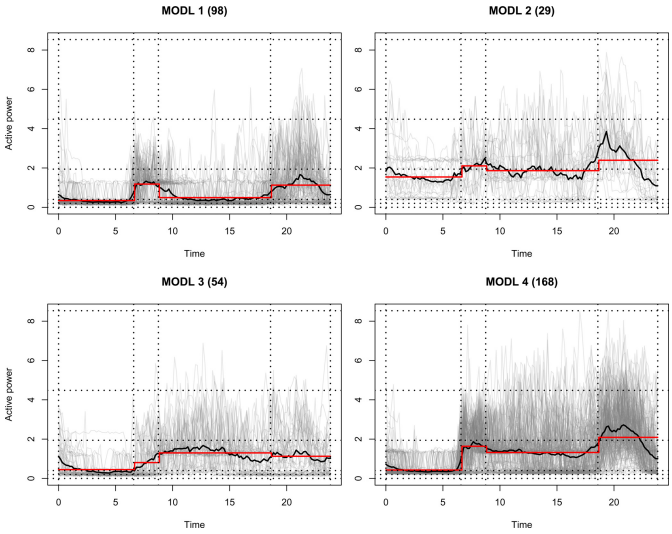


Fig. 6 The four clusters of curves retrieved using MODL with the average (black line) and the prototype (red solid line) curves. The number in parenthesis above each curve refers to the number of curves in the cluster.

MODL computes a global discretization for both the time and the power consumption. Conversely, the approach of [Hébrail et al., 2010] makes a discretization of the temporal variable only, that is different for each cluster of curves. In certain cases like the cluster 3 of the Figure 7, it may be suitable to avoid

over-discretizations, and a few number of time segments is better for a local interpretation. However, having common time segments for all the clusters enables an easier comparison between the clusters. In the context of the daily power consumption, MODL enables the identification of four periods: the *night* (midnight - 6.35 AM), the *morning* (6.35 AM - 8.45 AM), the *day* (8.45 AM - 6.35 PM) and the *evening* (6.35 PM - midnight). We are then able to compare the differences in terms of power consumption between the clusters of curves for each period of the day.

The approach of [Hébrail et al., 2010] is based on the k-means and thus minimizes the variance between the curves locally to each time segment. It is the reason why the prototype are close to the average curves in the clusters obtained by this approach. In MODL, this property is not wanted. As a consequence, the prototype and the average curves seem less correlated. MODL is based on a joint density estimation that yields more complex patterns. To highlight the differences in terms of patterns, we propose to focus on a specific time segment. The first interval (i.e the *night*) found by MODL also exists in the four clusters obtained using the approach of [Hébrail et al., 2010]. Let us focus on this time segment to investigate on the distributions of the power consumption measurements for each cluster of curves. To do that, we compute the probability density function of the power consumption variable locally to the first time segment, using a kernel density estimator [Sheather and Jones, 1991]. The results are displayed in Figures 8 and 9.

The density functions for the power consumption are similar for all the four clusters retrieved by the approach of [Hébrail et al., 2010] during the *night*: for all the four clusters, we observe that the power measurements are very dense around one

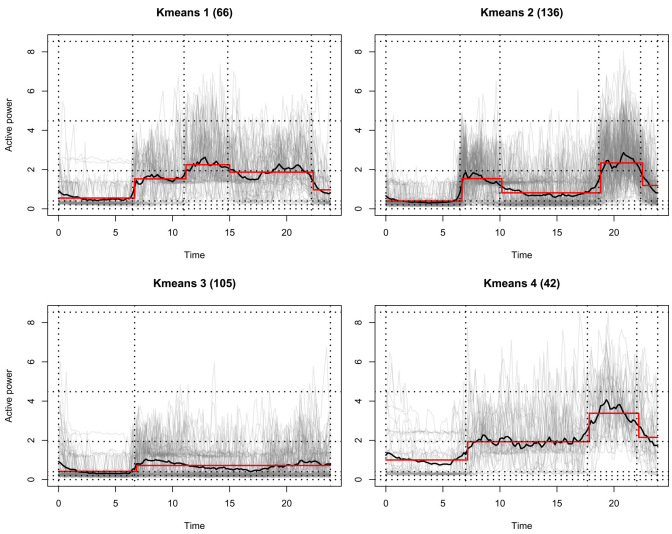


Fig. 7 The four clusters of days retrieved using the approach of [Hébrail et al., 2010] with the average (black line) and the prototype (red solid line) curves. The number in parenthesis above each curve refers to the number of curves in the cluster.

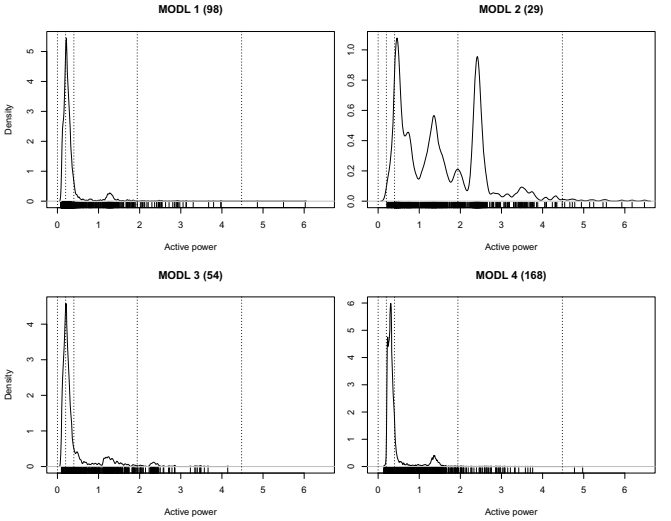


Fig. 8 Kernel density estimation of the power consumption measurements between midnight and 6.35 AM for each cluster of curves retrieved using MODL

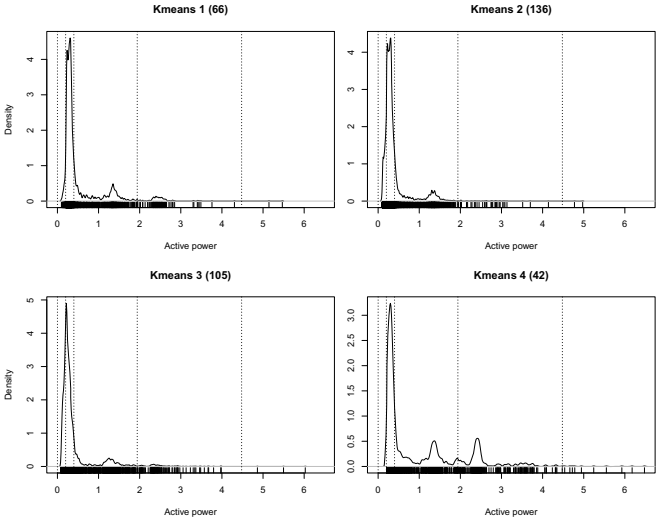


Fig. 9 Kernel density estimation of the power consumption measurements between midnight and 6.35 AM for each cluster of curves retrieved using the approach of [Hébrail et al., 2010]

unique low consumption value that corresponds to the year average power consumption of the studied time segment. As for MODL, the density functions are very similar for the clusters 1 and 3 and also very similar to the ones displayed in Figure 9.

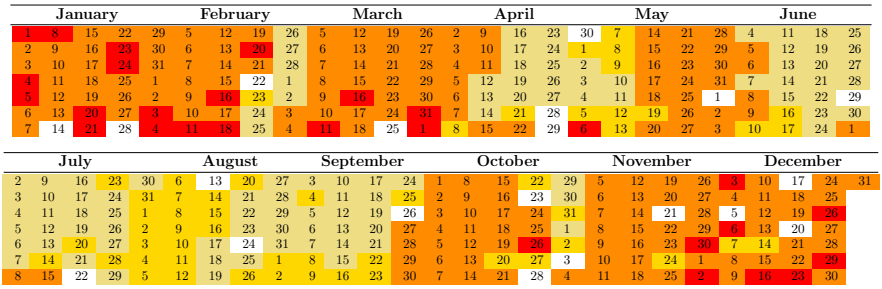


Fig. 10 Calendar of the year 2007 retrieved using MODL. Each line represents a day of the week. There are four colors (one per cluster), the redder the color, the higher the average power consumption of the cluster is. The white days correspond to days with missing data.

However, the cluster 4 is different in that the density peak has been translated to an upper power interval. Finally, the cluster 2 highlights multimodalities with three power values around which the measurements are dense. This complex pattern has been retrieved by MODL since it based on joint density estimation; the competing approach cannot track such patterns.

The curves of Figures 6 and 7 do not clearly highlight the differences between the results. Displaying the calendar with different colors for the 4 clusters gives a more powerful reading of the differences between the results obtained using both methods. This is displayed in Figures 10 and 11.

The calendar of the clusters retrieved using MODL (see Figure 10) emphasizes a certain seasonality. Indeed, the way the curves are grouped highlights a link with the weather and the temperatures in France this year. The summer, from June to September, is a season when the temperatures are usually high. On the calendar, there are two clusters corresponding to this period. The rest of the year, the temperatures are lower and lead to an increase of the power consumption which is materialized by the two other clusters. It appears that in late April and early May, the temperature was exceptionally high this year: these days have been classified into the summer clusters. Interestingly, the cluster shown in Figure 3.(a) where nobody was at home and the power consumption is low, has been included into a summer cluster (periods from the 23^{th} of February to the 2^{nd} of March and from the 29^{th} of October to the 3^{rd} of November).

For its part, the calendar obtained using the approach of [Hébrail et al., 2010] does not show a seasonality as the one retrieved using MODL does. The clusters are more distributed all over the year. The dark blue cluster (i.e the one with the higher average power consumption) groups however only cold winter days and can be compared to the reddest cluster of the Figure 10. The palest cluster (i.e the one with the lower average power consumption) characterizes also the warmer days and the days where there is nobody at home (see Figure 3.(a)). As for the other ones with intermediate average power consumption, they do not show any correlation with the period of the day and thus do not allow an immediate interpretation.

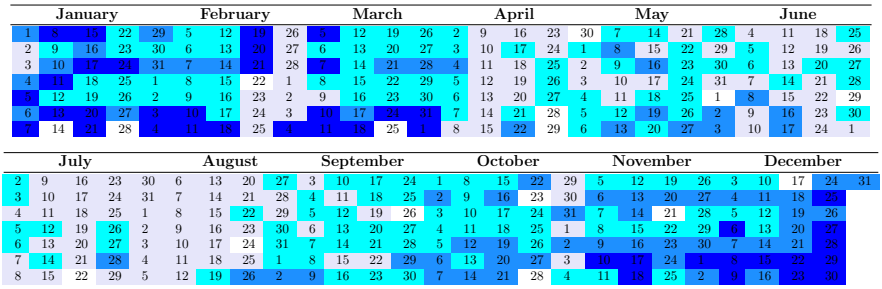


Fig. 11 Calendar of the year 2007 retrieved using the approach of [Hébrail et al., 2010]. Each line represents a day of the week. There are four colors (one per cluster), the bluer the color, the higher the average power consumption of the cluster is. The white days correspond to days with missing data.

All in all, both approaches track different patterns and consequently retrieve different clustering schemes. On the one hand, MODL requires no user-defined parameters and is suitable when there are no prior knowledges of the data. Moreover, the approach is supplemented by powerful exploratory analysis tools allowing a global interpretation of the results at different granularity levels. On the other hand, the approach of [Hébrail et al., 2010] enables a thorough understanding of the clusters by making a time decomposition locally to every cluster. In this practical case study, it appears that both methods are complementary.

6 Conclusion

In this paper, we have focused on functional data exploratory analysis, more particularly on curves clustering. The method that is proposed in this paper does not consider the data set as a collection of curves but rather as a set of data points with three features, two continuous, the point coordinates, and one categorical, the curve identifier. By clustering the curves and discretizing each point variable while selecting the best model according to a Bayesian approach, the method behaves as a nonparametric estimator of the joint density of both the curve and point variables. In case of large data sets, the best model tends to be too fine grained for an easy interpretation. To overcome this issue, a post-processing technique is proposed. This technique aims at merging successively the clusters until obtaining a simplified clustering while losing the least accuracy. This process is equivalent to making a hierarchical agglomerative classification, whose dissimilarity measure is a weighted sum of Kullback- Leibler divergences from the new cluster to the two merged clusters. Experimentations have been conducted on an artificial data set in order to highlight interesting properties of the method and on a real world data set, the power consumption of a home over a year. On the one hand, the finest model highlights interesting phenomena such as multimodal distributions for some time segments among the same cluster. As for the post-processing technique, a well-balanced dendrogram

and a concave Pareto chart emphasize the ability of the finest model to be simplified with few information loss, leading to a more interpretable clustering. An interpretation of these results has been made focusing on the differences with an alternative approach.

Beyond clustering of curves, the proposed method is able to cluster a collection of distributions. In future works, we plan to extend the method to multidimensional distributions by considering more than two point dimensions.

References

[Abraham et al., 2003] Abraham, C., Cornillon, P., Matzner-Løbe, E., Molinari, N.: Unsupervised curve clustering using b-splines. Scandinavian Journal of Statistics 30(3), 581–595 (2003)

[Abramowitz and Stegun, 1970] Abramowitz, M., Stegun, I.: Handbook of mathematical functions. Dover Publications Inc., New York (1970)

[Blei and Jordan, 2005] Blei, D.M., Jordan, M.I.: Variational inference for dirichlet process mixtures. Bayesian Analysis 1, 121–144 (2005)

[Boullé, 2010] Boullé, M.: Data grid models for preparation and modeling in supervised learning. In: Guyon, I., Cawley, G., Dror, G., Saffari, A. (eds.) Hands on Pattern Recognition. Microtome (2010) (in press)

[Cadez et al., 2000] Cadez, I., Gaffney, S., Smyth, P.: A general probabilistic framework for clustering individuals and objects. In: Proc. ACM Sixth Inter. Conf. Knowledge Discovery and Data Mining, pp. 140–149 (2000)

[Chamroukhi et al., 2010] Chamroukhi, F., Samé, A., Govaert, G., Aknin, P.: A hidden process regression model for functional data description. application to curve discrimination. Neurocomputing 73(7-9), 1210–1221 (2010)

[Chapman et al., 2000] Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., Wirth, R.: CRISP-DM 1.0: step-by-step data mining guide (2000)

[Cover and Thomas, 1991] Cover, T., Thomas, J.: Elements of information theory. Wiley-Interscience, New York (1991)

[Delaigle and Hall, 2010] Delaigle, G., Hall, P.: Defining probability density for a distribution of random functions. Annals of Statistics 38(2), 1171–1193 (2010)

[Ferraty and Vieu, 2006] Ferraty, F., Vieu, P.: Nonparametric Functional Data Analysis: Theory and Practice. Springer (2006)

[Gaffney and Smyth, 2004] Gaffney, S., Smyth, P.: Joint probabilistic curve clustering and alignment. In: Advances in Neural Information Processing Systems 17 (2004)

[Gasser et al., 1998] Gasser, T., Hall, P., Presnell, B.: Nonparametric estimation of the mode of a distribution of random curves. Journal of the Royal Statistical Society 60, 681–691 (1998)

[Hansen and Mladenovic, 2001] Hansen, P., Mladenovic, N.: Variable neighborhood search: principles and applications. European Journal of Operational Research 130, 449–467 (2001)

[Hastie et al., 2001] Hastie, T., Tibshirani, R., Friedman, J.: The elements of statistical learning. Springer (2001)

[Hébrail et al., 2010] Hébrail, G., Hugueney, B., Lechevallier, Y., Rossi, F.: Exploratory Analysis of Functional Data via Clustering and Optimal Segmentation. Neurocomputing 73(7-9), 1125–1141 (2010)

[Neal, 2000] Neal, R.M.: Markov chain sampling methods for dirichlet process mixture models. Journal of Computational and Graphical Statistics 9(2), 249–265 (2000)

[Nguyen and Gelfand, 2011] Nguyen, X., Gelfand, A.: The dirichlet labeling process for clustering functional data. Sinica Statistica 21(3), 1249–1289 (2011)

[Ramsay and Silverman, 2005] Ramsay, J., Silverman, B.: Functional Data Analysis. Springer Series in Statistics. Springer (2005)

[Rissanen, 1978] Rissanen, J.: Modeling by shortest data description. Automatica 14, 465–471 (1978)

[Sheather and Jones, 1991] Sheather, S., Jones, M.: A reliable data-based bandwidth selection method for kernel density estimation. Journal of the Royal Statistical Society. Series B (Methodological), 683–690 (1991)

[Teh, 2010] Teh, Y.W.: Dirichlet processes. In: Encyclopedia of Machine Learning. Springer (2010)

[Vogt et al., 2010] Vogt, J.E., Prabhakaran, S., Fuchs, T.J., Roth, V.: The translation-invariant wishart-dirichlet process for clustering distance data (2010)

[Wallach et al., 2010] Wallach, H.M., Jensen, S.T., Dicker, L., Heller, K.A.: An alternative prior process for nonparametric bayesian clustering. In: AISTATS, pp. 892–899 (2010)

Multi-view Clustering on Relational Data

Francisco de A.T. de Carvalho, Yves Lechevallier,
Thierry Despeyroux, and Filipe M. de Melo

Abstract. Clustering is a popular task in knowledge discovery. In this chapter we illustrate this fact with a new clustering algorithm that is able to partition objects taking into account simultaneously their relational descriptions given by multiple dissimilarity matrices. The advantages of this algorithm are threefold: it uses any dissimilarities between objects, it automatically ponderates the impact of each dissimilarity matrice and it provides interpretation tools. We illustrate the usefulness of this clustering method with two experiments. The first one uses a data set concerning handwritten numbers (digitized pictures) that must be recognized. The second uses a set of reports for which we have an expert classification given *a priori* so we can compare this classification with the one obtained automatically.

1 Introduction

Clustering is a popular task in knowledge discovery and it is applied in various fields including data mining, pattern recognition, computer vision, etc. [Gordon, 1999; Jain et al., 1999]. Clustering methods aim at organizing a set of objects into clusters such that items within a given cluster have a high degree of similarity, while items belonging to different clusters have a high degree of dissimilarity. A precise definition of the dissimilarity between objects is thus very important.

Some of the clustering techniques are called partitioning methods. Partitioning methods seek to obtain a single partition of the input data into a given number of clusters. Often, such methods look for a partition that optimizes (locally) an adequacy criterion function.

Francisco de A.T. de Carvalho · Filipe M. de Melo
Centro de Informatica -CIn/UFPE - Av. Prof. Luiz Freire, s/n -Cidade Universitaria - CEP 50740-540, Recife-PE, Brazil
e-mail: {fatc,fmm}@cin.ufpe.br

Yves Lechevallier · Thierry Despeyroux
INRIA, Paris-Rocquencourt, 78153 Le Chesnay Cedex, France
e-mail: {Yves.Lechevallier,Thierry.Despeyroux}@inria.fr

F. Guillet et al. (eds.), *Advances in Knowledge Discovery and Management,*
Studies in Computational Intelligence 527,
DOI: 10.1007/978-3-319-02999-3_3,

Two usual representations of the objects upon which clustering can be based are (usual or symbolic) feature data and relational data. When each object is described by a vector of quantitative or qualitative values the set of vectors describing the objects is called feature data. When each object is described by a vector of sets of categories, intervals or weight histograms, the set of vectors describing the objects can be considered as symbolic (feature) data, according to the Symbolic Data Analysis (SDA) approach[Bock and Diday, 2000]. Alternatively, when each pair of objects is represented by a relationship, then we have relational data. The most common case of relational data is when we have (a matrix of) dissimilarity data, say $R = [r_{il}]$, where r_{il} is the pairwise dissimilarity (often a distance) between objects i and l.

Many methods and algorithms have been proposed in order to cluster (usual or symbolic) feature data[Gordon, 1999; Jain et al., 1999; Kaufman and Rousseeuw, 1990]. However, few clustering models have been proposed for relational data. [Frigui et al., 2007] observed that several applications, as content-based image retrieval, would benefit strongly from clustering methods for relational data. In SDA, many effective dissimilarity measures proposed to the comparison of symbolic data are not differentiable with respect to the prototype parameters and thus, they could not be used in clustering methods for symbolic feature data based on objective functions. For example, in order to cluster constrained symbolic data, [De Carvalho et al., 2009] used the dynamic clustering algorithm for relational data [De Carvalho et al., 2012]. The constraints were taken into account during the computation of a suitable dissimilarity function between the symbolic feature data in order to obtain a relational data set.

In this paper we will focus on relational data. When the representation of an object is not unique, we speak of multi-view data. Multi-view data can be found in many domains such as bioinformatics, marketing, etc. [Cleuziou et al., 2009], and in structural documents. For example, in XML documents with many sections, each of these sections can be interpreted as a different view.

This paper presents a clustering algorithm that is a variant of the one given in [De Carvalho et al., 2012], that is able to partition objects taking simultaneously into account their relational descriptions given by multiple dissimilarity matrices. The main idea is to obtain a collaborative role of the different dissimilarity matrices [Pedrycz, 2002] in order to obtain a final consensus partition [Leclerc and Cucumel, 1987].

The dissimilarity matrices could have been generated using different sets of variables and a fixed dissimilarity function (the final partition gives a consensus between different views (sets of variables) describing the objects), using a fixed set of variables and different dissimilarity functions (the final partition gives the consensus between different dissimilarity functions) or using different sets of variables and dissimilarity functions. Moreover, the influence of the different dissimilarity matrices is not equally important in the definition of the clusters in the final consensus partition. Thus, in order to obtain a central partition from all dissimilarity matrices, it is necessary to learn cluster-dependent relevance weights for each dissimilarity matrix.

[Frigui et al., 2007] proposed CARD, a clustering algorithm that is able to partition objects taking into account multiple dissimilarity matrices and that learns a relevance weight for each dissimilarity matrix in each cluster. CARD is mainly based on the well known fuzzy clustering algorithms for relational data RFCM [Hathaway et al., 1989] and FANNY [Kaufman and Rousseeuw, 1990].

The clustering algorithm proposed in this paper is designed to give a partition and a prototype for each cluster as well as to learn a relevance weight for each dissimilarity matrix by optimizing an adequacy criterion that measures the fit between clusters and their representatives. These relevance weights change at each algorithm iteration and are different from one cluster to another. The method is based on the dynamic hard clustering algorithm for relational data [Lechevallier, 1974; De Carvalho et al., 2008, 2009] and on adaptive distances [Diday and Govaert, 1977; De Carvalho and Lechevallier, 2009]. One of the advantage of the algorithm is that it provides interpretation tools that help in understanding the result.

In order to demonstrate the usefulness of this new clustering algorithm, we apply it on two different applications. The first one concerns the clustering of handwritten digits (0 to 9) that are scanned in binary pictures. The data that are used are available from the "UCI machine learning repository". The second one uses the example given [De Carvalho et al., 2010] taking a document data base for which we have an expert categorization.

2 A Dynamic Clustering Algorithm Based on Multiple Dissimilarity Matrices

In this section, we introduce an extension of the dynamic clustering algorithm for relational data [De Carvalho et al., 2008] which is able to partition objects taking simultaneously into account their relational descriptions given by multiple dissimilarity matrices.

In this new version, the prototype is no more defined as an object, but as a vector of objects from E. For each matrix there is one associated object.

Let $E = \{e_1, \ldots, e_n\}$ be a set of n examples and let p dissimilarity $n \times n$ matrices $(\mathbf{D}_1, \ldots, \mathbf{D}_j, \ldots, \mathbf{D}_p)$ where $\mathbf{D}_j[i,l] = d_j(e_i, e_l)$ gives the dissimilarity between objects e_i and e_l on dissimilarity matrix $\mathbf{D}_j$. Assume that the prototype $g_k = (g_{k1}, \ldots, g_{kp})$ is the prototype vector of cluster C_k, where each component belongs to the set E, *i.e.* , $\mathbf{g}_k \in E^p \, (k = 1, \ldots, K)$, with $g_{kj} \in E \, (j = 1, \ldots, p)$.

The dynamic hard clustering algorithm with relevance weight for each dissimilarity matrix looks for a partition $P = (C_1, \ldots, C_K)$ of E into K clusters and the corresponding prototype vector $\mathbf{g}_k \in E^p$ representing the cluster C_k in P and a weight for each dissimilarity matrix such that the adequacy criterion J is locally optimized. The adequacy criterion is defined as

$$J = \sum_{k=1}^{K} \sum_{e_i \in C_k} d_{\lambda_k}(e_i, \mathbf{g}_k) = \sum_{k=1}^{K} \sum_{e_i \in C_k} \sum_{j=1}^{p} \lambda_{kj} d_j(e_i, g_{kj}) \tag{1}$$

in which

$$d_{\lambda_k}(e_i, \mathbf{g}_k) = \sum_{j=1}^{p} \lambda_{kj} d_j(e_i, g_{kj}) \tag{2}$$

is the dissimilarity between an example $e_i \in C_k$ and the cluster prototype $g_k \in E^p$ parameterized by the relevance weight vector $\lambda_k = (\lambda_{k1}, \ldots, \lambda_{kj}, \ldots, \lambda_{kp})$ where λ_{kj} is the weight for the dissimilarity matrix $\mathbf{D}_j$ for the cluster C_k, and $d_j(e_i, g_{kj}))$ is the local dissimilarity d_j between an example $e_i \in C_k$ and the cluster prototype $g_{kj} \in E$.

Our clustering algorithm alternates the three following steps:

- ***Step 1: Definition of the Best Prototype Vectors***
  In this step, the partition $P = (C_1, \ldots, C_K)$ of E into K clusters and the relevance weight matrix λ are fixed.
  For each cluster C_k we compute the prototype vector $\mathbf{g}_k$ which minimizes the clustering criterion J. This vector contains the components g_{kj}, objects of E, that are obtained using :

$$l = argmin_{1 \leq h \leq n} \sum_{e_i \in C_k} \lambda_{kj} d_j(e_i, e_h) \tag{3}$$

- ***Step 2: Definition of the Best Relevance Weight Matrix***
  In this step, the partition $P = (C_1, \ldots, C_K)$ of E and the vector of prototypes $\mathbf{g} = (\mathbf{g}_1, \ldots, \mathbf{g}_K)$ are fixed.
  The element j of the relevance weight vector $\lambda_k = (\lambda_{k1}, \ldots, \lambda_{kj}, \ldots, \lambda_{kp})$, which minimizes the clustering criterion J under $\lambda_{kj} > 0$ et $\prod_{j=1}^{p} \lambda_{kj} = 1$, is calculated by the following expression:

$$\lambda_{kj} = \frac{\left\{\prod_{h=1}^{p} \left[\sum_{e_i \in C_k} d_h(e_i, g_{kh})\right]\right\}^{\frac{1}{p}}}{\left[\sum_{e_i \in C_k} d_j(e_i, g_{kj})\right]} \tag{4}$$

  Remark The more the examples in the cluster C_k are close to the component g_{kj} of the prototype $\mathbf{g}_k$ considering the matrix of dissimilarity $\mathbf{D}_j$, the higher is the value of the weight λ_{kj}.
- ***Step 3: Definition of the Best Partition***
  In this step, the vector of prototypes $\mathbf{g} = (g_1, \ldots, g_K)$ and the relevance weight matrix λ are fixed.
  The cluster C_k is updated according to the following allocation rule:

$$C_k = \{e_i \in E : d_{\lambda_k}(e_i, \mathbf{g}_k) < d_{\lambda_h}(e_i, \mathbf{g}_h) \forall h \neq k\} \tag{5}$$

  If the minimum is not unique, e_i is assigned to the class having the smallest index.

It's easy to demonstrate that each previous step decreases the criterion J.

The dynamic hard clustering algorithm with relevance weight for each dissimilarity matrix sets an initial partition and alternates three steps until convergence, when the criterion $J(P, \lambda, \mathbf{g})$ reaches a stationary value representing a local minimum.

Comparing with the initial algorithm [De Carvalho et al., 2012], using a vector prototype allows to optimize the choice of the prototype and of the weight locally, by class and by dissimilarity matrix. The clustering criterion J is decomposed according to the dissimilarity matrices, and according to classes and dissimilarity matrices simultaneously, allowing to interpret the classes against matrices.

3 Interpreting Clusters and Partition

Let T be the criterion corresponding to the criterion J applied to a clustering in a unique class of E. The tools that help to interpret the classes and the partition are based on the decomposition of the criterion T in two parts. The first one corresponds to the dispersion intra-classes W (W corresponds to the clustering criterion J) and the second one corresponds to the dispersion inter-classes B. We use the approach given by [Chavent et al., 2006] that permits to compute this decomposition even if computing the inter-classes dispersion B is impossible (see [De Carvalho et al., 2012]).

Let $P = (C_1, \ldots, C_K)$ the final partition $E = \{e_1, \ldots, e_n\}$ in K classes. Let $\mathbf{g}_k$ the prototype and λ_k the vector of relevance weight of $C_k\,(k = 1, \ldots, K)$. Suppose also that the global prototype is the vector $\mathbf{g} = (g_1, \ldots, g_p)$ where $g_j \in E\,(j = 1, \ldots, p)$.

The global dispersion T of the partition $P = (C_1, \ldots, C_K)$ is defined by

$$T = \sum_{k=1}^{K} \sum_{e_i \in C_k} d_{\lambda_k}(e_i, \mathbf{g}) = \sum_{k=1}^{K} \sum_{e_i \in C_k} \sum_{j=1}^{p} \lambda_{kj} d_j(e_i, g_j) \tag{6}$$

where the global prototype $\mathbf{g}$, that minimizes the global dispersion T, is composed of $g_j = e_l \in E$ computed using :

$$l = argmin_{1 \leq h \leq n} \sum_{k=1}^{K} \sum_{e_i \in C_k} \lambda_{kj} d_j(e_i, e_h) \tag{7}$$

The global dispersion is decomposed in

a) $T = \sum_{k=1}^{K} T_k$ with $T_k = \sum_{e_i \in C_k} \sum_{j=1}^{p} \lambda_{kj} d_j(e_i, g_j)$;
b) $T = \sum_{k=1}^{K} \sum_{j=1}^{p} T_{kj}$ with $T_{kj} = \sum_{e_i \in C_k} \lambda_{kj} d_j(e_i, g_j)$;
c) $T = \sum_{j=1}^{p} T_j$ with $T_j = \sum_{k=1}^{K} \sum_{e_i \in C_k} \lambda_{kj} d_j(e_i, g_j)$

The dispersion intra-classes W is given by the clustering criterion J(see 1):

a) $J = \sum_{k=1}^{K} J_k$ with $J_k = \sum_{e_i \in C_k} \sum_{j=1}^{p} \lambda_{kj} d_j(e_i, g_{kj})$;
b) $J = \sum_{j=1}^{p} J_j$ with $J_j = \sum_{k=1}^{K} \sum_{e_i \in C_k} \lambda_{kj} d_j(e_i, g_{kj})$;
c) $J = \sum_{k=1}^{K} \sum_{j=1}^{p} J_{kj}$ with $J_{kj} = \sum_{e_i \in C_k} \lambda_{kj} d_j(e_i, g_{kj})$

One can easily show that

i) $T \geq J$;
ii) $T_k \geq J_k\,(k = 1, \ldots, K)$;

iii)$T_j \geq J_j\,(j = 1,\ldots,p)$;
iv)$T_{kj} \geq J_{kj}\,(k = 1,\ldots,K; j = 1,\ldots,p)$.

Given the global dispersion, the intra-classes dispersion and their decomposition, the indexes for the help to interpretation of classes and partition introduced by [Chavent et al., 2006] can be easily be adapted to the new algorithm.

The global quality of the final partition is $Q(P) = 1 - \frac{J}{T}$. An index $Q(P)$ close to 1 indicates a partition of better quality (more homogeneous classes).

The global quality of the final partition according to each dissimilarity matrix is given by $Q_j(P) = 1 - \frac{J_j}{T_j}$. A value for $Q_j(P)$ close to 1 indicates a good quality of the partition P according to the dissimilarity matrix $\mathbf{D}_j$. The comparison between $Q_j(P)$ and $Q(P)$ shows that the discriminant power of the dissimilarity matrix $\mathbf{D}_j$ is greater than the average discriminant power of all the dissimilarity matrices.

4 Applications

To illustrate the usefulness of our new algorithm, we use it on two different data sets. The first one is a set of digitized handwritten digits, the second a set of scientific activity reports.

4.1 Handwritten Digits Dataset

Our first example concerns the clustering of "multiple features" data available in the "UCI machine learning repository". This set of data contains handwritten digits (0 to 9) that are scanned in binary pictures. The 2000 handwritten digits (objects) are described by 649 numerical variables. These variables are partitioned in 6 different sets (views):

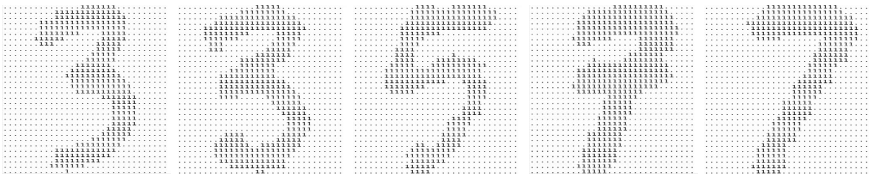


Fig. 1 Digitized handwritten digit '3', '3', '5', '7', '7'

- 76 Fourier coefficients describing the shape of the digits
- 64 Karhunen-Love coefficients
- 240 pixels average in 2 x 3 windows
- 47 Zernike moments
- 6 Morphological characteristics

Theses data are structured in 10 *a priori* classes containing 200 objects, each class corresponding to one digit.

We first consider 7 data tables: one in which the objects are described by all the 649 variables (table "mfeat") and 6 other tables in which the objects are described by one of the 6 different "views", each "view" having respectively 76 (table "mfeatFou"), 216 (table "mfeatFac"), 64 (table "mfeatKar"), 240 (table "mfeatPix"), 47 (table "mfeatZer"), and 6 (table "mfeatMor") variables.

Then 7 relational data tables are obtained from theses 7 data tables using the Euclidean distance. All these tables are then normalized according to their global dispersion [Chavent, 2005] to have the same dispersion. This means that each dissimilarity $d(\mathbf{x}_i,\mathbf{x}'_i)$ in a given relational data table has been normalized as $\frac{d(\mathbf{x}_i,\mathbf{x}'_i)}{T}$ where $T=\sum_{i=1}^{n} d(e_i,g)$ is the global dispersion and $g=e_l\in E=\{e_1,\ldots,e_n\}$ is the global prototype, which is computed according to $l=argmin_{1\leq h\leq n}\sum_{i=1}^{n} d(e_i,e_h)$.

Our clustering algorithm has been performed first on the relational data table "mfeat" and then simultaneously in the 6 relational data tables "mfeatFou", "mfeatFac", "mfeatKar", "mfeatPix", "mfeatZer", and "mfeatMor", corresponding to the 6 different "views" to obtain a partition in 10 clusters. The clustering algorithm is run 100 times and the best result according to the adequacy criterion J is selected. Our goal is to compare the partition obtain by our clustering algorithm with the partition in 10 clusters given *a priori*. The comparison criterion that we have chosen are the overall (global) error rate of classification (*OERC*)[Breiman et al., 1984], the corrected Rand index (*CR*)[Hubert and Arabie, 1985], and the F-measure[Van Rijisbergen, 1976].

Results

The values of the *CR*, F-measure and *OERC* indexes, obtained from the final partition computed by our clustering algorithm applied to the relational data table "mfeat", are respectively 0.518, 0.674, and 37.75%.

The values of the same indexes obtained from the final partition computed by our clustering algorithm applied simultaneously to the 6 relational data tables corresponding to the 6 different "views" are respectively 0.762, 0.879 et 12.10%. The table 1 shows the relevance weight matrix of the relational data tables in the clusters.

The table 2 shows the confusion matrix into 10 cluster computed for the final partition.

We can see that the dissimilarity matrix "mfeatMor" is the most pertinent one for defining all the clusters. We also see that the dissimilarity matrix "mfeatFac" has a relevance weight as important that the one for the dissimilarity matrix "mfeatMor" for the cluster 3.

The global quality of the final partition is $Q(P)=1-\frac{J}{T}=0.919$. Closer is the index $Q(P)$ to 1 better is the partition quality (with more homogeneous clusters).

The global quality of the final partition relative to each dissimilarity matrix $Q_j(P)=1-\frac{J_j}{T_j}$ $(j=1,\ldots,6)$ is shown in Table 3. A value of $Q_j(P)$ close to 1 is an indication of a good quality of the partition P relative to the dissimilarity matrix $\mathbf{D}_j$. Comparing $Q_j(P)$ with $Q(P)$ shows that the discriminant power of the

Table 1 Relevance Weight Matrix of the Relational Data Tables in the Clusters

	Relevance Weight of Dissimilarity Matrices					
Clusters	1-mfeatMor	2-mfeatZer	3-mfeatPix	4-mfeatKar	5-mfeatFou	6-mfeatFac
1	6.728	0.713	0.562	0.595	0.533	1.165
2	12.543	0.615	0.515	0.546	0.434	1.059
3	2.891	0.919	0.612	0.646	0.454	2.091
4	3.412	1.083	0.526	0.562	0.513	1.778
5	5.318	0.828	0.573	0.640	0.454	1.361
6	135.631	0.338	0.236	0.252	0.318	1.147
7	54.559	0.484	0.270	0.290	0.393	1.223
8	5.276	0.794	0.547	0.596	0.421	1.733
9	8.163	0.749	0.504	0.559	0.383	1.505
10	8367.671	0.199	0.124	0.134	0.097	0.363

Table 2 Confusion Matrix

	Clusters (Handwritten Digits)									
Clusters	'7'	'1'	'5'	'2'	'4'	'0'	'8'	'3'	'6'	'9'
1	193	15	4	16	6	0	0	30	2	0
2	1	170	0	0	4	0	3	1	5	0
3	0	0	149	1	0	2	0	27	0	0
4	1	0	6	178	0	0	1	3	0	0
5	1	2	1	1	183	0	1	2	3	0
6	0	0	0	0	0	188	18	0	0	0
7	0	11	0	0	0	9	174	0	3	0
8	4	0	40	3	1	1	2	137	1	0
9	0	2	0	1	6	0	1	0	186	0
10	0	0	0	0	0	0	0	0	0	200

Table 3 Global Quality of the Partition P relatively to each Dissimilarity Matrix (%)

	Dissimilarity Matrices					
	1-mfeatMor	2-mfeatZer	3-mfeatPix	4-mfeatKar	5-mfeatFou	6-mfeatFac
$Q_j(P)$	98.44	47.28	43.58	47.16	35.09	65.69

dissimilarity matrix "mfeatMor" is greater that the average discriminant power of all the dissimilarity matrices.

Table 4 shows the heterogeneity index $J(k) = \frac{J_k}{J}$ and the quality index $Q(k) = 1 - \frac{J_k}{T_k}$ for each cluster $k = 1, \ldots, 10$. One can sea, for example, that the cluster 10 (digit '9') is more homogeneous while the cluster 6 (digit '0') is of best quality.

Table 5 shows the index $Q_j(k) = 1 - \frac{J_{kj}}{T_{kj}}$, that gives the quality of the cluster $C_k\,(k = 1, \ldots, 10)$ in the dissimilarity matrix $\mathbf{D}_j\,(j = 1, \ldots, 6)$. Closer to 1 is the value of this index, better is the quality of this cluster in this dissimilarity matrix. While $Q(P)$ is a global index, $Q_j(k)$ is a local one for a given cluster and a given dissimilarity matrix. More, the comparison between the indices $Q_j(k)$ and $Q(k)$

Table 4 Heterogeneity Index and Quality Index of a Cluster(%)

	Cluster k									
	1	2	3	4	5	6	7	8	9	10
Cardinal	266	184	179	189	194	206	197	189	196	200
$J(k)$	17.52	10.20	12.67	10.34	14.07	3.75	6.14	12.38	10.39	2.48
$Q(k)$	88.63	84.80	93.36	93.42	89.42	97.70	93.70	93.43	84.18	88.73

Table 5 Quality of Clusters in the Dissimilarity Matrices (%)

	Dissimilarity Matrix					
Classes	1-mfeatMor	2-mfeatZer	3-mfeatPix	4-mfeatKar	5-mfeatFou	6-mfeatFac
1	97.69	38.36	49.84	52.10	39.84	50.57
2	96.80	32.34	45.94	46.65	30.16	34.07
3	98.79	23.15	34.46	39.43	35.03	39.12
4	98.76	61.18	47.37	52.09	41.66	53.72
5	97.93	13.31	42.37	46.97	20.34	56.26
6	99.58	81.82	60.47	65.07	69.41	77.24
7	98.85	42.33	22.75	26.86	41.98	51.96
8	98.81	25.05	30.37	34.73	20.25	23.20
9	96.50	00.00	45.99	50.50	18.19	68.99
10	03.77	91.28	55.00	53.25	42.25	97.11

gives the dissimilarity matrices that characterize the cluster k. For example, the dissimilarity matrix "mfeatMor" is characteristic of the clusters 1 to 9, while the matrices "mfeatZer" and "mfeatFac" are characteristic of the cluster 10 (digit '9').

4.2 Document Data Base Categorization

As a second application of our algorithm, we use it to categorize a document data base. The document data base is a collection of scientific activity reports produced by each INRIA (The French National Institute for Research in Computer Science and Control) research team in 2007. Theses deliverables are send to the French parliament for public funding assessing and are also made available to its industrial and research partners.

Research teams are grouped into scientific *themes* that do not correspond to an organizational structure (such as departments or divisions), but act as a virtual structure for the purpose of presentation, communication and evaluation. Figure 2 gives a view of this categorization. The choice of the themes and the allocation of the teams are mostly related to strategic objectives and scientific closeness between existing teams, however some geographical constraints, such as the desire for a theme to be representative of most INRIA centers are taken into account. Our aim is to compare the *a priori* categorization given by INRIA of the reports with that induced by the clustering algorithm here proposed.

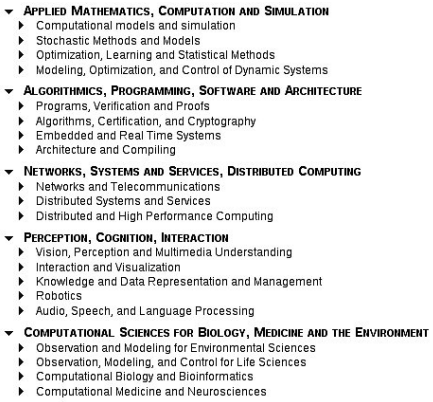


Fig. 2 INRIA research categorization

Each report (RA) is written in English and using LaTeX, it is automatically translated into XML, then to HTML and published on the Web. In the rest of the paper we implicitly refer to the XML version of the Activity Report. The logical structure of the RA is defined by an XML DTD with a few mandatory sections and some optional parts.

In this application we consider activity reports from 164 INRIA research teams in 2007. The XML version of these documents contains 173 files, a total of 613 000 lines, more than 40 Mbytes of data. Figure 3 gives an example of an activity report summary.

Fig. 3 Example of an activity report summary

In these activity reports, four sections have been selected to describe a research team: *overall objectives*, *scientific foundations*, *dissemination* and *new results*. The *overall objectives* part defines the research objectives, *scientific foundations* provides the scientific background followed by potential applications of the research domain, *Dissemination* includes any teaching activity, involvement with the research community (program committees, editorial boards, conference and workshop organization) and seminars, while the *new results* includes the principal results obtained during that year.

In a first step all the texts are preprocessed. Stop-words are removed, and the texts are annotated with part-of-speech and lemma information using treetagger.

Four feature data tables are build, each with 164 objects (the research teams) described by the frequent words (categories) present in one of the four sections. The numbers of frequent words in the sections *overall objectives*, *scientific foundations*, *dissemination*, and *new results* are respectively 220, 210, 404, and 547. Each cell on a data table gives the frequency of a word for the considered activity report section and research team.

Then, four relational data tables have been obtained from the 4 feature data tables through a dissimilarity measure derived from the affinity coefficient [Bacelar-Nicolau, 2000]. We assume that each individual is described by one set-valued variable ("presentation", etc.) which has m_j modalities (or categories) $\{1,\ldots,m\}$. An individual e_i is described by $\mathbf{x}_i = (n_{i1},\ldots,n_{im})$ where n_{ij} is the frequency of modality j. The dissimilarity between a pair of individuals e_i and $e_{i'}$ is given by:

$$d(\mathbf{x}_i,\mathbf{x}'_i) = 1 - \sum_{j=1}^{m} \sqrt{\frac{n_{ij}}{n_{i\bullet}}\frac{n_{i'j}}{n_{i'\bullet}}} \quad where \quad n_{i\bullet} = \sum_{j=1}^{m} n_{ij}.$$

All these relational data tables were normalized according to their global dispersion [Chavent, 2005]: each dissimilarity $d(\mathbf{x}_i,\mathbf{x}'_i)$ in a relation data table has been normalized as $\frac{d(\mathbf{x}_i,\mathbf{x}'_i)}{T}$ where $T = \sum_{i=1}^{n} d(e_i,g)$ is the global dispersion and $g = e_l \in E = \{e_1,\ldots,e_n\}$ is the global prototype, which is computed according to $l = argmin_{1\leq h\leq n} \sum_{i=1}^{n} d(e_i,e_h)$.

Results

The clustering algorithm has been performed simultaneously on these 4 relational data tables ("presentation", "foundation", "dissemination" and "bibliography") in order to obtain a partition in $K \in \{1,\ldots,15\}$. For a fixed number of clusters K, the clustering algorithm is run 100 times and the best result according to the adequacy criterion is selected.

Determining the appropriate number of clusters in a partition is a classical problem but no good solution exists[Milligan and Cooper, 1985]. To choose the right number of cluster, our strategy is those of the SPAD software[1]. It consists in choosing the best couple (inter-classes inertia, number of classes). The decrease of the number of classes increases the intra-classes inertia, so to get a partition with a

[1] `http://eng.spad.eu/`

good quality we must identify an important jump of the index. This peak can be found using the second order differences of the clustering criterion [Da Silva, 2009; Charrad et al., 2010].

The discrete first derivative of J according to k is $Df(x) = (f(x+h) - f(x))/h$ and the second one is $D2f(x) = (f(x+h) - 2f(x) + f(x-h))/h^2$. When h tends to 0 this is equivalent to the usual derivative.

A partition in 4 clusters is chosen because at this spot the second derivative is maximal (see Fig. 4). A partition in 11 clusters would also be possible as it is a local maximum.

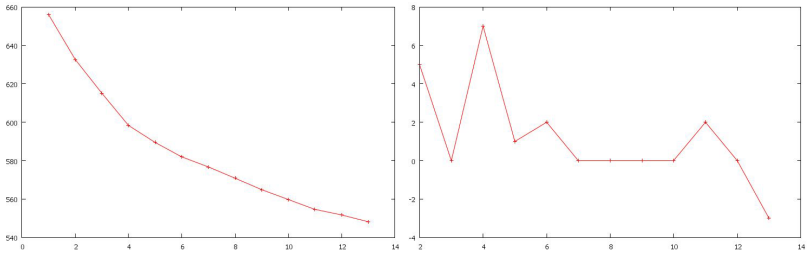


Fig. 4 *J* criteria, second derivative

The 4-clusters partition obtained with the here proposed algorithm was compared with the *a priori* 5-class partition of the INRIA in 2008. INRIA *a priori* categorization is as follows: "Applied Mathematics, Computation and Simulation (M)", "Algorithmics, Programming, Software and Architecture (A)", "Networks, Systems and Services, Distributed Computing (N) ", "Perception, Cognition, Interaction (P)" and "Computational Sciences for Biology, Medicine and the Environment (C)".

The activity reports refer to year 2007, and the expert classification by INRIA has been done in 2008. Between these two years some research teams have been closed and others has evolved. For this reason, only 154 activity reports has been used in the comparison between our automatic clustering and the expert classification done by INRIA.

Table 6 shows that the 4-clusters partition obtained with the clustering algorithm is quite consistent with the *a priori* 5-class categorization, except for the M and C class.

Category 5, Computational Sciences for Biology, Medicine and the Environment, is artificial and is distributed (considering the vocabulary that is used) in two clusters, depending on the fact that the subject is more mathematical or more cognitive. Thus, the cluster C3 could be labelled "Simulation/control/modelisation", and the cluster C4 "Data processing".

The relevance weight matrix for the for variables (sections) used in the activity reports is shown in table 7.

Table 6 Distribution table of 154 reports (2007) in 5 *a priori* categories (2008) (rows) in the 4 clusters (columns)

	C1	C2	C3	C4
M - Applied Mathematics, Computation and Simulation	1	1	**20**	6
A - Algorithmics, Programming, Software and Architecture	**17**	3	1	9
N - Networks, Systems and Services, Distributed Computing	1	**28**	2	2
P - Perception, Cognition, Interaction	5	1	2	**35**
C - Computational Sciences for Biology, Medicine and the Environment	0	0	**11**	**9**

Table 7 Relevance Weight Matrix of the Dissimilarity matrices in the classes

	Relevance Weight of Dissimilarity Matrices			
Clusters	overall objectives	scientific foundations	new results	dissemination
1	0.969026	0.979387	1.000909	1.052727
2	1.019705	0.934093	1.073774	0.977738
3	0.966223	1.068582	1.073115	0.902545
4	0.976156	0.993158	1.026519	1.004837

The values of the *CR*, F-measure and *OERC* indexes, obtained from the final partition computed by our clustering algorithm are respectively 0.360, 0.657 and 27.92%.

5 Conclusion

This paper introduced a new clustering algorithm that is able to partition objects taking into account simultaneously their relational descriptions given by multiple dissimilarity matrices. These matrices could have been generated using different sets of variables and dissimilarity functions.

This algorithm provides a partition and a prototype for each cluster as well as a relevance weight for each dissimilarity matrix by optimizing an adequacy criterion that measures the fit between clusters and their representatives. These relevance weights are automatically computed at each algorithm iteration and are different from one cluster to another. We also provide tools for the interpretation of the clusters and the partition provided by the algorithm.

Two experiments demonstrate the usefulness of this clustering method.

References

[Bacelar-Nicolau, 2000] Bacelar-Nicolau, H.: The affinity coefficient. In: Bock, H.H., Diday, E. (eds.) Analysis of Symbolic Data, pp. 160–165. Springer, Heidelberg (2000)

[Bock and Diday, 2000] Bock, H., Diday, E.: Analysis of Symbolic Data. Springer, Heidelberg (2000)

[Breiman et al., 1984] Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A.: Classification and Regression Trees. Chapman and Hall/CRC, Boca Raton (1984)

[Charrad et al., 2010] Charrad, M., Lechevallier, Y., Ahmed, M.B., Saporta, G.: On the number of clusters in block clustering algorithms. In: Guesgen, H.W., Murray, R.C. (eds.) FLAIRS Conference. AAAI Press (2010)

[Chavent, 2005] Chavent, M.: Normalized k-means clustering of hyper-rectangles. In: Proceedings of the XIth International Symposium of Applied Stochastic Models and Data Analysis (ASMDA 2005), Brest, France, pp. 670–677 (2005)

[Chavent et al., 2006] Chavent, M., De Carvalho, F.A.T., Lechevallier, Y., Verde, R.: New clustering methods for interval data. Computational Statistics 21(2), 211–229 (2006)

[Cleuziou et al., 2009] Cleuziou, G., Exbrayat, M., Martin, L., Sublemontier, J.-H.: Cofkm: A centralized method for multiple-view clustering. In: ICDM 2009 Ninth IEEE International Conference on Data Mining, Miami, USA, pp. 752–757 (2009)

[Da Silva, 2009] Da Silva, A.: Analyse de données évolutives: application aux données d'usage Web. PhD thesis, Université Paris-IX Dauphine (2009)

[De Carvalho et al., 2009] De Carvalho, F.A.T., Csernel, M., Lechevallier, Y.: Clustering constrained symbolic data. Pattern Recognition Letters 30(11), 1037–1045 (2009)

[De Carvalho and Lechevallier, 2009] De Carvalho, F.A.T., Lechevallier, Y.: Partitional clustering algorithms for symbolic interval data based on single adaptive distances. Pattern Recognition 42(7), 1223–1236 (2009)

[De Carvalho et al., 2012] De Carvalho, F.A.T., Lechevallier, Y., De Melo, F.M.: Partitioning hard clustering algorithms based on multiple dissimilarity matrices. Pattern Recognition 45(1), 447–464 (2012)

[De Carvalho et al., 2008] De Carvalho, F.A.T., Lechevallier, Y., Verde, R.: Clustering methods in symbolic data analysis. In: Diday, E., Noirhomme-Fraiture, M. (eds.) Symbolic Data Analysis and the SODAS Software, pp. 181–204. Wiley-Interscience, San Francisco (2008)

[De Carvalho et al., 2010] De Carvalho, F.A.T., Despeyroux, T., De Melo, F.M., Lechevallier, Y.: Utilisation de matrices de dissimilarit multiples pour la classification de documents. In: EGC-M 2010, Extraction et Gestion des Connaissances, Alger, Algérie, pp. 1–10 (2010)

[Diday and Govaert, 1977] Diday, E., Govaert, G.: Classification automatique avec distances adaptatives. R.A.I.R.O. Informatique Computer Science 11(4), 329–349 (1977)

[Frigui et al., 2007] Frigui, H., Hwang, C., Rhee, F.C.: Clustering and aggregation of relational data with applications to image database categorization. Pattern Recognition 40(11), 3053–3068 (2007)

[Gordon, 1999] Gordon, A.: Classification. Chapman and Hall/CRC, Boca Raton, Florida (1999)

[Hathaway et al., 1989] Hathaway, R.J., Davenport, J.W., Bezdek, J.C.: Relational duals of the c-means algorithms. Pattern Recognition 22, 205–212 (1989)

[Hubert and Arabie, 1985] Hubert, L., Arabie, P.: Comparing partitions. Journal of Classification 2(1), 193–218 (1985)

[Jain et al., 1999] Jain, A., Murty, M., Flyn, P.: Data clustering: A review. ACM Comput. Surv. 31(3), 264–323 (1999)

[Kaufman and Rousseeuw, 1990] Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data. Wiley, New York (1990)

[Lechevallier, 1974] Lechevallier, Y.: Optimisation de quelques critères en classification automatique et application a l'étude des modifications des protéines sériques en pathologie clinique. PhD thesis, Université Paris-VI (1974)

[Leclerc and Cucumel, 1987] Leclerc, B., Cucumel, G.: Concensus en classification: une revue bibliographique. Mathématique et Sciences Humaines 100, 109–128 (1987)
[Milligan and Cooper, 1985] Milligan, G.W., Cooper, M.C.: An examination of procedures for determining the number of clusters in a data set. Psychometrika 50, 159–179 (1985)
[Pedrycz, 2002] Pedrycz, W.: Collaborative fuzzy clustering. Pattern Recognition Lett. 23, 675–686 (2002)
[van Rijisbergen, 1976] van Rijisbergen, C.J.: Information retrieval. Butterworth-Heinemann, London (1976)

Relaxing Time Granularity for Mining Frequent Sequences

Asma Ben Zakour, Sofian Maabout, Mohamed Mosbah, and Marc Sistiaga

Abstract. In an industrial context application aiming at performing aeronautic maintenance tasks scheduling, we propose a frequent *Interval Time Sequences* (ITS) extraction technique from discrete temporal sequences using a sliding window approach to relax time constraints. The extracted sequences offer an interesting overview of the original data by allowing a temporal leeway on the extraction process. We formalize the ITS extraction under classical time and support constraints and conduct some experiments on synthetic data to validate our proposal.

1 Introduction

Sequential patterns mining is an important data mining task. It handles several kind of sequential information like network intrusion detection [Srinivasulu et al., 2010], identification of behavior trends [Rabatel et al., 2009] and tandem repeat DNA sequences [Ceci et al., 2011]. According to the target application, different forms of sequences may be extracted e.g., timestamped (see [Yi-Cheng et al., 2010] and [Fournier-Viger et al., 2008]), summarized [Pham et al., 2009], composite (see [Ceci et al., 2011]) and multidimensional sequences (see [Rabatel et al., 2009] and [Plantevit et al., 2007]). Considering timestamped patterns, time representation and time granularity are both more or less relevant regarding the application domain. We introduce through the work presented in this paper a new form of timestamped sequences and propose *ITS-PS*: an algorithm enabling their extraction. The sequences we want to extract aim to characterize aeronautic usage behaviors with respect to their impact on maintenance tasks application. They are intended to be used to predict maintenance application in order to perform its scheduling process. For example, for aircraft lives data, let V_i refer to the flight i,

Asma Ben Zakour · Sofian Maabout · Mohamed Mosbah · Marc Sistiaga
LaBRI, University of Bordeaux, CNRS UMR 5800, France and
2MoRO Solutions, Bidart, France

F. Guillet et al. (eds.), *Advances in Knowledge Discovery and Management,*
Studies in Computational Intelligence 527,
DOI: 10.1007/978-3-319-02999-3_4,

M_j refer to a maintenance task j and $\mathscr{S} = \{S_1, S_2\}$ be a set of historic sequences where $S_1 = \langle (0,V_1)(2,V_2)(3,V_3)(5,M_1)\rangle$ and $S_2 = \langle (0,V_1)(2,V_3)(3,V_2)(6,M_1)\rangle$. Let the minimal support constraint be equal to 2. Our method returns the sequence: $\langle ([0,0]V_1)([2,3]V_2\,V_3)([5,6]M_1)\rangle$. Its meaning is as follows: "If flight V_1 occurs, followed by both flights V_2 and V_3 *in any order* but in a time interval $[2,3]$ after V_1 then, maintenance task M_1 is performed in a time lying in the interval $[5,6]$ after V_1". Such a pattern allows to group V_2 and V_3 in the same relevant behavior.

For this propose, extracted patterns must convey three criteria: (1) the frequency of events chronology in order to describe frequent usage behaviors, (2) timestamped sequences to ensure relevance and precision of maintenance prediction and (3) relaxation of local order of events, i.e., if two events occur in a close time interval, then the chronology of their respective occurrences could be considered as irrelevant, then they may be considered as co-occurring.

To fulfill those three criteria we propose to *merge* temporally close and consecutive events (associated to a discrete timestamp) into an unique set of simultaneous events associated with an interval timestamp. This interval reflects an uncertainty on the occurrence time of events. The "closeness" of events is managed via a user defined maximal sliding window size. In the previous example, events V_2 and V_3 have been merged and timestamped with $[2,3]$ following the application of a sliding window size equal to 1. Note that V_1 cannot be merged with them because its two occurrences are "far" from those of V_2 and V_3. Related methods (some of them are presented in section 2) do not allow extracting such information. For instance, the *GSP* algorithm proposed in [Agrawal and Srikant, 1996] applied on the sequences of the previous example with the same minimal support constraint and window size, extracts the sequence: $\langle (V_1)(V_2\,V_3)(M_1)\rangle$. Even if it contains the same events chronology as ours, it does not provide any temporal information. Hence, the only information it carries is: "flight V_1 is followed by flights V_2 and V_3, in any order, and they are themselves followed by the application of the maintenance task M_1". This frequent sequence cannot be efficiently used by an aeronautic expert who needs to reduce maintenance costs and aircraft interruptions by forecasting the most precise maintenance task application moment since the sequence does not provide any temporal information.

Paper Organization

The following section presents a concise overview of related work, especially the difference between our method and other approaches extracting interval timestamped sequences. Then, we formally define the semantics of sequences with uncertainty time intervals. Section 4 details the extraction process. We conclude our work by comparing our approach with an existing method, the *GSPM* algorithm proposed in [Hirate and Yamana, 2006]). Finally, we present some avenues for future work.

2 Related Work

Several works found in the literature deal with grouping events and frequent interval sequences extraction. The approach proposed in [Pham et al., 2009] merges some sequences events by using a sliding window. Grouping is performed during a pre-processing step, and then an extraction algorithm is applied. The resulting grouped events are however timestamped with a discrete time reference which is an arbitrary choice motivated by treatment simplicity. This represents an information loss w.r.t events occurring times. Moreover, applying the sliding window during a pre-processing phase increases the size of initial sequences since several grouping possibilities for the same sequence. Hence this introduces an ambiguity on the support counting.

Other approaches consider the initial data as timestamped with intervals. These intervals represent *duration* times not uncertainty about the exact discrete occurrence time. [Giannotti et al., 2006] extracts frequent sequences by using an *Apriori* like algorithm [Agrawal and Srikant, 1996]. It first identifies frequent patterns apart from timestamps. Then, for an extracted frequent pattern, it intersects intervals events occurrences in order to provide a succession of intervals associated with the frequent sequence.

In [Guyet and Quiniou, 2011], a timestamped sequence is represented by a hypercube whose axes are the sequence events. The similarity between sequences is expressed by hypercubes intersection volume. Sequences are grouped using this similarity. If there are enough grouped sequences, then a representative one is extracted and considered as a interesting pattern.

Extraction algorithms presented in [Wu and Chen, 2007], [Yi-Cheng et al., 2010] use Allen's interval relationships [Allen, 1983]. A *PrefixSpan*-like [Pei et al., 2001] algorithm is applied on interval timestamped sequences. Both algorithms results presented in [Wu and Chen, 2007] and in [Yi-Cheng et al., 2010] consist in relationships sequences between events and not on timestamped sequences. To the best of our knowledge, the closest work to ours is the one of [Hirate and Yamana, 2006]. Authors extract frequent sequences with interval timestamps from discrete timestamped sequences. They use a *level function* which is actually a non sliding window. Intuitively, events belonging to the same time interval are merged. But since close events may belong to consecutive but different intervals, they cannot be merged. This is due to the fact that the window is fixed.

Concerning the extraction technique itself, we find two main procedures in the literature: the first one is level wise, or breadth first, like *A priori* technique [Agrawal et al., 1994; Agrawal and Srikant, 1995]. This method has been used in many works, for instance, in [Rabatel et al., 2009] and [Agrawal and Srikant, 1996] it was applied to discrete temporal sequences and in [Giannotti et al., 2006] was applied to interval temporal sequences. The principal limitations of the Apriori extraction approach are (1) the number of generated candidates and (2) the number of the whole database scanning which is equal to the number of the levels used during the extraction process. Its principal advantage is its relatively low memory consumption since it maintains only one copy of the database in the main memory. The breadth

first method uses a *divide and conquer* strategy by progressively reducing the search space and selecting at each step 1-patterns (candidates of size 1). Each such selected 1-pattern P is actually a witness of an $i+1$-pattern $Q.P$ where Q is previously evaluated. Evaluating P, e.g its support, is equivalent to that of $Q.P$ because the former is performed in a *projected* data set, i.e., the part that we already know that it contains Q. Hence, intuitively, during the computation advancement, the underlying data is progressively reduced while the patterns keep their size equal to 1 making the evaluation as simple as possible. This second strategy has been used in e.g., [Hirate and Yamana, 2006; Pei et al., 2001; Yi-Cheng et al., 2010; Fournier-Viger et al., 2008; Wu and Chen, 2007; Guyet and Quiniou, 2008].

[Li et al., 2012] used it in order to approximate the set of close patterns extracted from a long sequence by introducing the gap constraint.The main limitation of this approach is its relative great memory consumption since at each pattern extension a physical projection of the database is created. Its advantages are (1) the fewer number of candidates generated at each step and (2) the increasingly reduced data to be scanned during the extraction process. We adopt this second method for applying our algorithm which is inspired by *PrefixSpan* proposed in [Pei et al., 2001].

3 Preliminaries

We first recall some of the standard definitions regarding simple temporal sequences as formulated in previous works, e.g., [Hirate and Yamana, 2006], [Pei et al., 2001] and [Fournier-Viger et al., 2008]. Let $\omega = \{e_1, e_2, \ldots, e_k\}$ be a set of events. A transaction is defined a set of events supposed as occurring simultaneously. A temporal sequence is a succession of chronologically ordered transactions. Each transaction in a temporal sequence is associated with a discrete timestamp, it is denoted by $S = \langle (t_1, I_1), (t_2, I_2) \ldots (t_n, I_n) \rangle$, $n \in \mathbb{N}$ where $\forall 1 \leq i \leq n$, I_i is a transaction and t_i its timestamp. A timed sequences database is a set of temporal sequences where each of them is identified by a unique identifier denoted by *id_sequence*.

Definition 1. Subsumption Let $S' = \langle (t'_1, I'_1), (t'_2, I'_2) \ldots (t'_m, I'_m) \rangle$ and $S = \langle (t_1, I_1), (t_2, I_2) \ldots (t_n, I_n) \rangle$ be two temporal sequences. S subsumes S' iff there exist $1 \leq i_1 \leq \cdots \leq i_m \leq n$ with $I'_1 \subset I_{i_1} \ldots I'_m \subset I_{i_m}$ and $t'_1 = t_{i_1}, \ldots, t'_m = t_{i_m}$.

If S subsumes S' we also say that S is a super-sequence of S'. The support of a sequence S in a database sequences D is the percentage of sequences from D which are super-sequences of S. It is denoted by $support_D(S)$. S is said frequent if its support is greater than a fixed minimum threshold *minsupp*.

Now, we define uncertainty interval temporal sequences. We recall that sequences with interval timestamps consider the transaction's interval as an uncertainty during which the transactions events do occur. If we consider for example the 1-sequence $S = \langle ([t_b, t_e], e) \rangle$, intuitively S means that: "the event e occurs punctually between times t_d and t_e".

Definition 2 (uncertainty Interval Temporal Sequences (ITS)). An n length interval temporal sequence S (ITS) is denoted by:

$$S = \langle([m_1, M_1], I_1), ([m_2, M_2], I_2) \ldots ([m_n, M_n], I_n)\rangle$$

where $([m_i, M_i], I_i)$ is a transaction with interval timestamp such that:

- $m_i \leq occurrence\_time(e_j) \leq M_i$. for all $e_j \in I_i$;
- An interval temporal sequence is consistent if, $m_i \leq m_{i+1}$ and $M_i \leq M_{i+1}$;
- $m_1 = 0$, i.e., timestamps in S are relative to m_1.

Example 1. Let $S1 = \langle([0,1], A)([2,2], BC)\rangle$. It means: "$A$ occurs between time points 0 and 1, B and C occur simultaneously between 1 and 2 temporal units after A". The lower bound of the interval associated to A is the time reference of $S1$. Each of B and C, occur *at most* $2\ (2-0)$ time units after A and *at least* $1\ (2-1)$ time unit after A.

$S2 = \langle([0,3], A)([1,2], B)([2,5], C)\rangle$ is not consistent since the upper bound of the second interval is lower than the upper bound of the first interval$(2 < 3)$.

One may note that Interval Temporal Sequences are special cases of classical temporal sequences. Indeed, $S = \langle(t_1, I_1) \ldots (t_n, I_n)\rangle$ is equivalent to:

$$ITS(S) = \langle([t_1, t_1], I_1), \ldots, ([t_n, t_n], I_n)\rangle$$

In order to fit temporal parameters on extracted sequences and patterns formulation needs, we consider temporal constraints. They aim to: (i) set a maximum threshold of uncertainty, (ii) control minimum and maximum temporal delays between successive transactions and (iii) control the minimum and maximum temporal whole pattern length.

- *Gap* controls the earlier delay between two successive transactions and fix two thresholds: (1) the *mingap* represents the minimum delay below which the successive transactions are considered as too close to represent significant dissociated events and (2) the *maxgap* is the maximum delay between two transactions above which they are considered as too far to be directly correlated (strictly consecutive). Let *SI* be an n length ITS. *SI* satisfies the temporal constraints: *mingap*, *maxgap* if and only if $\forall\ 2 \leq i \leq n$:

$$mingap \leq (m_i - M_{i-1}) \leq maxgap$$

- *Whole_interval* controls the whole length time of a sequence by fixing two thresholds: (1) the *min_whole_interval* fixing the minimum temporal extent of the sequence below which the behavior conveyed by the sequence is not considered as complete and (2) the *max_whole_interval* fixing the maximal temporal extent of the sequence above which the behavior conveyed by the sequence is considered as more than a single one. The *Whole_interval* regulates the time length of a sequence such that for *SI* an n length ITS, *SI* satisfies the temporal constraints: *min_whole_interval*, *max_whole_interval* iff:

$$min\_whole\_interval \leq |m_1 - M_n| \leq max\_whole\_interval$$

- *Sliding Window* enables grouping successive transactions into a single one timestamped with an interval. The size of the sliding window fixes a maximum group spreading and so the maximum interval width. The window size regulates a maximum uncertainty threshold. Let SI be an n length ITS. Then SI satisfies the window size ws iff $\forall\ 1 \leq i \leq n$

$$|M_i - m_i| \leq ws$$

Example 2. Consider the ITS $S = \langle([0,1],A)([2,3],BC)([6,10],D)\rangle$ and the time constraints $mingap$ and $maxgap$ respectively equal to 2 and 3. SI does not satisfy $mingap$ because $m_2 - M_1 = 2-1 = 1 \leq 2$. On the other hand, S satisfies $maxgap$ since for all its successive transactions the maxgap constraint is satisfied ($m_2 - M_1 = 2-1 \leq 3$; $m_3 - M_2 = 6-3 \leq 3$). For a sliding window size equal to 3, S is not valid since $M_3 - m_3 = 10-6 > 3$. For a sliding window constraint fixed to 4, S satisfies it by all its timestamps.

The temporal constraints allow us to manage the temporal parameter into an ITS; they control minimum (respectively maximum) temporal leeway between two successive transactions since the correlation between both of them can be meaningful. Actually, minimum (respectively maximum) gap avoids considering too close (respectively too far) transactions as successive. These constraints are used in several algorithms, e.g., [Agrawal and Srikant, 1996; Fournier-Viger et al., 2008; Rabatel et al., 2009; Li et al., 2012]. In the same way, the *whole_interval* constraint fixes a minimum (respectively maximum) threshold for the whole sequence duration in order to guarantee a meaningful correlation between the transactions belonging to the same sequence[Hirate and Yamana, 2006; Fournier-Viger et al., 2008]. On the other hand, the sliding window manages a balance between the events grouping and uncertainty of their occurrences [Agrawal and Srikant, 1996; Rabatel et al., 2009].

In order to combine these temporal constraints in a consistent manner, we fix the following relationships between them:

$$ws < mingap;\ \ mingap \leq maxgap$$

Temporal constraints enumerated above are fixed by the user in order to extract relevant ITS. In the rest of this section we focus our work on the application of the *ws* constraint.

3.1 Merging Sequences

In this section, we define a $\diamondsuit$ operator that merges successive transactions belonging to the same sequence. For an ITS, $\diamondsuit$ starts from a position j and merges spreading transactions covered by a window size until the last transaction of the ITS. Hence, $\diamondsuit$ has three parameters: a sequence S, a position j in S and a window size ws. More formally:

Definition 3. Let $S = \langle([m_1,M_1],I_1)([m_2,M_2],I_2)\dots([m_n,M_n],I_n)\rangle$ be an ITS, $j < n$ an integer and a window size ws. Then the $\diamondsuit_{ws}$ operator is defined by:

$$\diamondsuit_{ws}(S,j) = S' = \langle([m'_1,M'_1],I'_1)([m'_2,M'_2],I'_2)\dots([m'_n,M'_n],I'_k)\rangle$$

- where $\forall 1 \leq i < j : ([m'_i,M'_i],I'_i) = ([m_i,M_i],I_i)$;
- $\exists\, j \leq l_j \leq l_{j+1},\dots l_i \cdots \leq l_{k-1} \leq n$ such that:
  - $I'_j = \cup_{p=j}^{l_j} I_p$; $\dots$ $I'_i = \cup_{p=l_{i-1}+1}^{l_i} I_p$; $\dots$ $I'_k = \cup_{p=l_{k-1}+1}^{l_n} I_p$,
  - $m'_j = m_j, M'_j = M_{l_j}, \dots, m'_i = m_{l_{i-1}+1}, M'_i = M_{l_i}, \dots m'_k = m_{l_{k-1}+1}, M'_k = M_n$
  - $|m_j - M_{l_j}| \leq ws$; $\dots |m_{l_{i-1}+1} - M_{l_i}| \leq ws$; $\dots |m_{l_{k-1}+1} - M_n| \leq ws$.

Example 3. Consider $SI = \langle([0,2],A)([1,2],B)([3,5],C)([4,6],D)\rangle$ and a window size $ws = 3$. Then $\diamondsuit_3(SI,1) = \langle([0,2],AB)([3,6],CD)\rangle$, it applies the grouping operator $\diamondsuit$ w.r.t ws =3 from the first transaction of SI to the last one. Events from the first (respectively last) couple of transactions are grouped and their intervals merged. Since both transactions are spread into the window size and $(2-0) \leq 3$ (respectively $(6-3) \leq 3$)). We note that $\diamondsuit_3(SI,2) = SI$ it applies the grouping operator $\diamondsuit$ w.r.t ws =3 from the second transaction of SI to the last one. Actually, for the start grouping position 2, the second and third transactions cannot be merged since their unified interval is too large regarding the window size. Finally, $\diamondsuit_3(SI,3) = \langle([0,2],A)([1,2],B)([3,6],CD)\rangle$ and $\diamondsuit_3(SI,4) = SI$.

Remark 1. The resulting sequencing of the application of $\diamondsuit_{ws}$ operator may merge transactions containing the same item. Indeed, it may happen that an item appears in two successive transactions of the initial sequence. If these transactions are merged, then the item will appear only once (transactions are sets). However, because of the window size condition, we know that both occurrences of the same item take place in a time interval at most equal to ws. Hence, the time interval associated to the item in the new sequence does include both initial timestamps. Moreover, the fact that it appears only once does not affect its support in the database since the support is the number of sequences of the database that support an item. Therefore, replacing two occurrences by one it the same sequence does not incur any loss of information regarding the support measure.

Now we define the $\overbrace{\diamondsuit}$ operator which for an n length ITS and a sliding window of size ws, provides a set of ITS's. It is the set of results of all applications of the $\diamondsuit$ operator on a n length sequence (applied by successively starting the merge from the first transaction to the last one , $j \in [1,n-1]$). Intuitively $\overbrace{\diamondsuit}$ merges successive transactions by sliding the window size along the input sequence. It provides the set of all summarized sequences that represent the input one.

Definition 4. Let $S = \langle([m_1,M_1],I_1)\dots([m_n,M_n],I_n)\rangle$, ws be a window size and $S_j = \diamondsuit_{ws}(S,j) \forall 1 \leq j < n$. Then:

$$\overbrace{\diamondsuit}^{ws}(S) = \{S_1, S_2, \dots, S_{n-1}\}$$

Example 4. Let $S = \langle([0,2],A)([1,2],B)([3,4],C)([4,6],D)\rangle$ and $ws = 3$. $\overbrace{\diamondsuit_3}(S) = \{\langle([0,2],AB)([3,6],CD)\rangle, \langle([0,2],A)([1,4],BC)([4,6]D)\rangle, \langle([0,2],A)\ ([1,2],B)\ ([3,6]CD)\rangle\}$.

3.2 Support

Now we define the supporting relationship between ITSs. Intuitively, S supports S' also said S' is a sub-sequence of S iff events of each transaction of S' are contained in one (or successive) transaction(s) of S and transactions interval of S' imply (a combination of) S transaction(s) interval(s). Note that the transactions chronology order must be preserved. More precisely,

Definition 5. Let S and S' be two ITS. Let $S = \langle([m_1,M_1],I_1),\ldots,([m_n,M_n],I_n)\rangle$ and $S' = \langle([m'_1,M'_1],I'_1)\ \ldots\ ([m'_k,M'_k],I'_k)\rangle$. S is a super-sequence of S' denoted by $S \sqsupseteq S'$ (equiv. S' is a sub-sequence of S denoted $S' \sqsubseteq S$) if and only if: $\forall([m'_j,M'_j]I'_j) \in S'$ and $e \in I'_j$ there exists $([m_k,M_k]I_k) \subset S$ such that:

- $e \in I_k$
- $[m_k,M_k] \subseteq [m'_j,M'_j]$ (we say that $[m_k,M_k]$ implies $[m'_j,M'_j]$)

Example 5. Let $SI_1 = \langle([0,2]A)([3,4],B)([5,6]C)\rangle$, $SI_2 = \langle([0,4]AB)\rangle$ and $SI_3 = \langle([0,2]A)([3,6]BC)\rangle$. $SI_1 \sqsupseteq SI_2$ since $[0,4]$ implies $[0,2]$ and $[3,4]$ and $SI_1 \sqsupseteq SI_3$. However, $SI_1 \not\sqsupseteq SI_4 = \langle([0,3]A)([2,6]BC)\rangle$ since $[0,2]$ does not imply $[0,3]$.

The support of an ITS w.r.t. a sequence database is the number of sequences in the collection that support the interval sequence.

Definition 6. The support of a ITS SI in a collection D is defined by:

$$supp_D(SI) = |\{S \in D \,|\, S \sqsupseteq SI\}|$$

Recall that temporal sequences are a special case of interval sequences. Thus, $S \sqsupseteq SI$.

4 ITS Extraction

This section describes the extraction process of ITS patterns from discrete temporal sequences by considering a frequency threshold *minsupp*, and the time constraints: *ws*, *mingap*, *maxgap*, *min_whole_interval* and *max_whole_interval*. We detail the *ITS-PS* (uncertainty interval temporal sequences-PrefixSpan) algorithm. It gradually groups frequent close events into a single transaction by applying a sliding window.

The algorithm applies a *pattern growth* approach [Pei et al., 2001] by performing a depth first extraction based on database projections. First, *ITS-PS* identifies the set of 1-patterns (frequent events) from the initial database *SDB* denoted by $L_1 = \{S; S = \langle([m = 0, M = 0], e)\rangle; support(e) \geq minsupp\}$. Then, recursively $i+1$-patterns are identified by extending an i-pattern. Each recursive step i applies two tasks:

- The first task identifies L_1 the set of frequent 1-ITS from the search space. A 1-ITS is considered frequent if: (1) the event of its transaction appears in a sufficient number of sequences of the search space, and (2) the maximum delay between its occurrences timestamps is at most equal to *ws*. Each 1-ITS is concatenated to the pattern extracted at the $i-1$ iteration to provide a frequent *i*-pattern. Then, a new iteration is executed. This step is detailed in the section 4.1
- The second task computes a new projection of the current data on each frequent 1-ITS computed at iteration *i*. Each new search space is a summary of the initial one such that it resumes each sequence that is a super-sequence of the $i+1$-pattern by selecting only sub-sequences considered as continuity of the $i+1$-pattern. The $i+1$-pattern is the concatenation of the *i*-pattern with the 1-ITS. This procedure is detailed in the section 4.2.

The recursive process continues until one of the two following conditions is satisfied: (1) No frequent 1-ITS is identified or (2) the projection procedure provides an empty search space.

Table 1 Example of sequences database *SDB*

SDB	
S_1	$\langle(0,A)(1,B)(2,CD)\rangle$
S_2	$\langle(0,A)(2,D)(3,B)(4,F)\rangle$

Table 2 SDB_A: the projection of *SDB* over $\langle([0,0]A)\rangle$

SDB_A	
S_1	$\langle(1,B)(2,CD)\rangle$
S_2	$\langle(2,D)(3,B)(4,F)\rangle$

Example 6. Let *SDB* be the data described in Table 1, $minsupp = 2$ and $ws = 2$. First, *ITS-PS* identifies frequent events and associates to each one the null interval. In the sample data, it identifies $L_1 = \{\langle([0,0]A)\rangle, \langle([0,0]B), \langle([0,0]D)\rangle\}$. It is the set of first transactions of all other extended frequent ITS.

Let us consider the frequent 1-ITS $\langle([0,0]A)\rangle$ the projection step summarizes *SDB* by retaining only the continuations of *A*, i.e. sub-sequences with *A* as a prefix. Table 2 shows the resulting projection. The extraction process continues and identifies in the new search space the frequent 1-ITS in order to extract longer patterns. Then, SDB_A is projected over each 1-pattern found frequent in it. When all extensions of $\langle([0,0]A)\rangle$ are identified those extending $\langle([0,0]B)\rangle$ will be explored and finally those extending $\langle([0,0]D)\rangle$.

4.1 Selecting Frequent 1-Sequences

Concerning the 1-ITS identification, the application of the sliding window allows us to associate shifted occurrences of the same event occurring in different sequences. This association is made under the condition that the delay between their two farthest occurrences is less or equal to the window size.

Example 7. Let's consider SDB_A from the previous example. B appears twice. These 2 occurrences are counted in the support because the maximal delay between them is less or equal to ws ($3-1=2\leq 2$). Thus, the 1-ITS $\langle([1,3]B)\rangle$ is frequent. In the same manner $\langle([2,2]D)\rangle$ is frequent because D appears in the two sequences of SDB_A and the time delay between the timestamps of its occurrences is less than ws ($2-2=0\leq ws$).

We now introduce the $\oplus$ operator for the concatenation of an ITS and a 1-ITS. Intuitively, when an i-ITS is extended by an 1-ITS, there exist two possibilities of concatenating the second at the end of the first: a T-extension and a S-extension. We describe them as follows:

- The T-extension merges the 1-ITS with the last transaction of the i-ITS. The timestamp of the resulting transaction is the union of both initial intervals. Considering ws, this kind of concatenation is possible when one of the intervals *implies* the other. It is denoted by $\oplus_T$
- The S-extension adds the 1-ITS as the $(i+1)$ transaction of the i-pattern. This extension is possible if the gap constraints and the coherence of the sequence are satisfied. It is denoted by $\oplus_S$

Finally, $\oplus$ defines the concatenation operator through $\oplus_T$ and $\oplus_S$. Both kinds of concatenation depend on the combination of upper bounds and lower bounds of the concerned intervals. Fig. 1 illustrates intervals relationship with respect to the concatenation type.

Definition 7. Let given ws, $S=\langle([m_1,M_1]I_1)\dots([m_n,M_n]I_n)\rangle$ and $S'=([t_b,t_e]I)$ both satisfying ws. The extension of S by S' is defined as follows:

$$S\oplus S'=\begin{cases} S\oplus_T S' & \text{if } [t_b,t_e] \text{ implies } [m_n,M_n] \\ & or [m_n,M_n] \text{ implies } [t_b,t_e] \\ S\oplus_S S' & \text{if } m_n\leq t_b \text{ and } M_n\leq t_e \\ S & \text{otherwise} \end{cases}$$

Example 8. Let $S=\langle([0,1]A)([2,3]B)\rangle$ and $S'=\langle([4,5]C)\rangle$. $S\oplus S'=S\oplus_S S'=\langle([0,1]A)([2,3]B)([4,5]C)\rangle$. If we consider $ws=3$ and $S''=\langle([1,3]D)\rangle$ then $S\oplus S''=S\oplus_T S''=\langle([0,1]A)([1,3]BD)\rangle$.

4.2 Projection

Concerning search space projection, we extend the classical process by using a restricted (to the window size) backward projection. Such projection allows to take

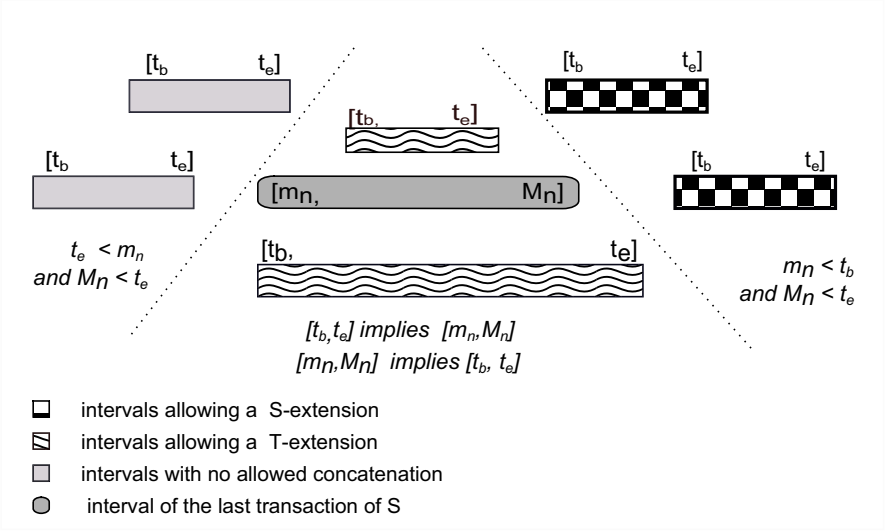


Fig. 1 Illustration of the intervals relationship w.r.t. the extensions types

into account the slide of the window and to consider locally (with regard to the window size) disordered events. This backward exploration permits the selection as an extension of a pattern events occurring frequently around (before or after) its last transaction. Intuitively the new projection holds both of the T-extensions and the S-extensions of the pattern to extend. T-extensions represent close events w.r.t the last transaction of the pattern and to the window size. T-extensions are located in a time delay at most equal to *ws* before and after the last event of the pattern. S-extensions represent events occurring after the pattern in the underlying sequences. The pattern backward analysis has already been used for patterns extension for instance in [Li et al., 2012] to prune the set of close extracted patterns.

Table 3 Projection of the sequences database SDB_A by $\langle([1,3]B)\rangle$

$SDB_{A,B}$	
S_1	$\langle(1,CD)\rangle$
S_2	$\langle(-1,D)(1,F)\rangle$

Example 9. Let us continue the execution of Example 6. Continuations of $\langle([0,0]A)([1,3]B)\rangle$ are identified in the projection of SDB_A by $\langle([1,3]B)\rangle$ denoted by $SDB_{A,B}$ (illustrated in Table 3). The first sequence represents only events appearing after B because there is no event occurring before it (their timestamps w.r.t B are positive). However, in the second sequence, D appears close to and before B (its timestamp w.r.t B is negative). So it is considered as one of its T-extensions and is time stamped

with -1: its time delay w.r.t B. In this new search space, D appears twice and the time delay between its occurrences is less than ws $(1-(-1)=2\leq 2)$. Therefore, the 1-ITS $\langle([-1,1]D)\rangle$ is frequent. Actually, the backwardness of the projection allows us to consider this event frequent despite the fact that in the two sequences it appears on both sides of B. In order to concatenate it to the last extracted pattern, we have first to adjust the temporal reference of $\langle([-1,1]D)\rangle$ w.r.t A. D appears earlier 1 temporal unit after B which turns to appear 3 units after A $(3-1=2)$. It appears at most 1 unit after B which itself appears 1 $(1--1=2)$ units after A. Hence, D is referenced by $[2,2]$ w.r.t the occurrences of A. $\langle([0,0]A)([1,3]B)\rangle \oplus \langle([2,2]D)\rangle = \langle([0,0]A)([1,3]B)\rangle \oplus_T \langle([2,2]D)\rangle = \langle([0,0]A)([1,3]BD)\rangle$.

Let us now consider the extension of $\langle([0,0]A)\rangle$ by the 1-ITS $\langle([2,2]D)\rangle$ frequent in the sequences presented in table 2. $\langle([0,0]A)\rangle \oplus \langle([2,2]D)\rangle = \langle([0,0]A)\rangle \oplus_S \langle([2,2]D)\rangle = \langle([0,0]A)([2,2]D)\rangle$. The projection of SBD_A by $\langle([2,2]D)\rangle$ is illustrated in Table 4. The 1-ITS $\langle([-1,1]B)\rangle$ is frequent and extends the last extracted pattern. We have first to adjust the time reference of the interval associated to B, (-1 +2 =1) for the lower bound and (1+2=3) for the upper bound, then $\langle([0,0]A)([2,2]D)\rangle \oplus \langle([1,3]B)\rangle = \langle([0,0]A)([2,2]D)\rangle \oplus_T \langle([1,3]B)\rangle = \langle([0,0]A)([1,3]BD)\rangle$

Table 4 Projection of SDB_A by $\langle([2,2]D)\rangle$

$SDB_{A,D}$	
S_1	$\langle(-1,B)(0,C)\rangle$
S_2	$\langle(1,B)(2,F)\rangle$

Using the simple backward projection provides in some cases the problem of multiple extractions of the same pattern. This case happens when the proximity between two events is analyzed several times. Actually, when close events can be merged on the same transaction, the order by which the events are considered does not matter because the result is always the same.

Property 1. Let $S=\langle([m_1,M_1]I_1)\dots([m_n,M_n]I_n)\rangle$, ws and $\alpha=\{([m_1,M_1]I_1),\dots,([m_p,M_p]I_p)\}$ the set of the T-extensions of S. Let $m=min(m_1\dots m_p)$ and $M=max(M_1\dots M_p)$ such that $M-m\leq ws$. The ITS provided by successive concatenations of S with all 1-ITS from α in any order are equivalent.

Proof. Let $S_p=([m_p,M_p]I_p)$ and $S_k=([m_k,M_k]I_k)$ two T-extensions of S. S_p and S_k are close to the last transaction of S and then close also from each other. We propose to evaluate the patterns $S\oplus([m_p,M_p]I_p)\oplus([m_k,M_k]I_k)=S\oplus_T([m_p,M_p]I_p)\oplus_T([m_k,M_k]I_k)$ and $S\oplus_T([m_k,M_k]I_k)\oplus_T([m_p,M_p]I_p)$

- Let us first consider the extension of S by $S_p=([m_p,M_p]I_p)$ and then by S_k to obtain: $S\oplus S_p=S\oplus_T S_p=\langle([m_1,M_1]I_1)\dots([m'_n,M'_n]I'_n)$ such that

$$\begin{cases} m'_n=min(m_n,m_p)\\ M'_n=max(M_n,M_p)\\ I'_n=I_n\cup\{I_p\}\end{cases}$$

Then the second concatenation provides: $S \oplus S_p \oplus S_k = S \oplus_T S_p \oplus_T S_k = \langle([m_1, M_1] I_1) \ldots ([m''_n, M''_n] I''_n)$ such that:

$$\begin{cases} m''_n = min(m'_n, m_k) = min(m_n, m_p, m_k) \\ M''_n = max(M'_n, M_k) = max(M_n, M_p, M_k) \\ I''_n = I'_n \cup \{e_k\} = I_n \cup \{I_p, I_k\} \end{cases}$$

- Now we consider the concatenation of S first with S_k and then with S_p: $S \oplus S_k = S \oplus_T S_k = \langle([m_1, M_1] I_1) \ldots ([m'''_n, M'''_n] I'''_n)$ such that:

$$\begin{cases} m'''_n = min(m_n, m_k) \\ M'''_n = max(M_n, M_k) \\ I'''_n = I_n \cup \{I_k\} \end{cases}$$

Then the second concatenation provides: $S \oplus S_k \oplus S_p = S \oplus_T S_k \oplus_T S_p = \langle([m_1, M_1] I_1) \ldots ([m^1_n, M^1_n] I^1_n)$ such that

$$\begin{cases} m^1_n = min(m'''_n, m_k) = min(m_n, m_p, m_k) \\ M^1_n = max(M'''_n, M_k) = max(M_n, M_p, M_k) \\ I^1_n = I'''_n \cup \{e_p\} = I_n \cup \{I_p, I_k\} \end{cases}$$

We can then conclude that $S \oplus_T S_p \oplus_T S_k = S \oplus_T S_k \oplus_T S_p$.

Let S_1 be the result pattern, consider two other *T-extension* $S_u = ([m_u, M_u] e_u)$ and $S_v([m_v, M_v] I_v)$ such that $\{S_v, S_u\} \in \alpha$. By the same manner, if we extend twice (1) by concatenating first S_u to S_1 and then S_v to the obtained sequence and (2) by concatenating first S_v to S_1 and then S_u to the obtained sequence. Then, it is clear that $S \oplus S_v \oplus S_u = S \oplus_T S_v \oplus_T S_u$.

We can finally conclude that whatever the number of T-extensions is, if we extend a sequence S with the same set of T-extensions by considering different orders, the result is always the same. □

In order to avoid multiple extractions of the same ITS k-pattern from an ITS $(i-1)$-pattern, Property 1 is useful. Indeed, we assume that backward exploration does not take into account events already processed as the last element of a i-pattern from the same $(i-1)$-pattern. To cope with this consideration, we suppose a total order $\lhd$ between events such that event e_1 is lower than e_2 w.r.t $\lhd$ (noted $e_1 \lhd e_2$). Let $I = \{e_1, \ldots, e_n\}$. For notation convenience, we note $e \lhd I$ iff $e \lhd e_i$ for $1 \leq i \leq n$ and generalize it to sets of events, i.e., $I_1 \lhd I_2$ iff $\exists e_j \in I_2$ such that $\forall e_i \in I_1$ $e_i \lhd e_j$.

Example 10. Let $\omega = \{A, B, C, D, E, F\}$ and $A \lhd B \lhd C \lhd D \lhd E \lhd F$, then $A \lhd EF$

Considering $\lhd$, we define the prefix and suffix of a sequence. Intuitively, the prefix of S w.r.t S' is the set of sub-sequences of S starting at the beginning of S and supporting S'. The suffix of S w.r.t S' is the set of sub-sequences of S containing the possible continuations of S'.

Definition 8 (*Prefix*). Let $S = \langle (t_1, I_1) \dots (t_n, I_n) \rangle$ and $S' = \langle ([m,M], I) \rangle$. The subsequence $\langle (t_1, I_1), (t_2, I_2) \dots (t_j, I_j) \rangle$ is a Prefix of S w.r.t S' iff $I_j \supseteq I$ and $t_j \subseteq [m,M]$. We denote by Prefix(S,S') the set of prefixes of S w.r.t S'.

We define the $wsufix_{\triangleleft}$ that represents the possible continuations (T-extensions and S-extensions) of a sequence S' on a sequence S by taking into account the window size backward. We use property 1, and the $\triangleleft$ order to avoid the extraction of patterns already discovered. In this way, the $\triangleleft$ order selection is applied on T-extensions which extend on an area span equal to $2ws$ and centered on S. Formally,

Definition 9 ($wsufix_{\triangleleft}$). Let $\omega = \{e_1, e_2 \dots e_m\}$, $S = \langle (t_1, I_1) \dots (t_n, I_n) \rangle$ and $S' = \langle ([m,M], I) \rangle$.

- For $(1 \leq j \leq n)$ such that $I \in I_j$ and $t_j \in [m,M]$ $\langle (t'_k, I'_k) \dots (t'_j, I'_j \setminus \{I\}) \dots (t'_n, I'_n) \rangle$ is a suffix of S w.r.t S' iff:
  1. $\forall i, k \leq i \leq n, t'_i = t_i - t_j$
  2. $t'_k \leqslant (t'_j - ws)$ and $t'_{k-1} > (t_j - ws)$
  3. $\forall i, k \leq i \leq n$ and $t'_i \leq ws$ then $I'_i = I_i \setminus \{e_u | e_u \triangleleft I\}$
- Otherwise, the empty sequence $\langle \emptyset \rangle$ is the suffix of S w.r.t S'.

Fig. 2 illustrates the Prefix and Suffix concepts.

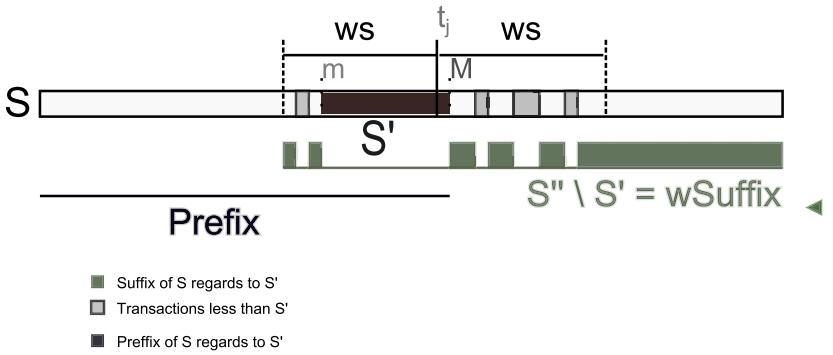


Fig. 2 Illustration of Prefix and Suffix of S w.r.t S'

We denote by $wsuffix_{\triangleleft}(S_1, S')$ the set of suffix of S regards to S'.

We define now the projection considering the new definition of suffix. It resumes a sequence database w.r.t an ITS by calculating the possible continuations of this ITS in the data sequences.

Definition 10 ($wprojection_{\triangleleft}$). Let BDS and $S' = ([m,M]I)$. We define the projection $wprojection_{\triangleleft}$ of BDS by SI as follows:

$$wprojection_{\triangleleft}(BDS|S') = \{S''|S'' = wsuffixe_{\triangleleft}(S,S'), S \in BDS\}$$

Example 11. Let us reconsider Example 3 by applying the $wprojection_{\lhd}$. Let $\lhd$ be the lexicographic order. In SDB_A the following 1-ITS are frequent: $\langle([1,3]B)\rangle$ and $\langle([2,2]D)\rangle$. Considering $\lhd$, $\langle([0,0]A)\rangle$ is first extended by $\langle([1,3]B)\rangle$ and the pattern $\langle([0,0]A)\rangle \oplus \langle([1,3]B)\rangle = \langle([0,0]A)\rangle \oplus_S \langle([1,3]B)\rangle = \langle([0,0]A)([1,3]B)\rangle$ is identified. The projection of SDB_A by $\langle([1,3]B)\rangle$ is then calculated. The result is denoted by $SDB_{A,B}$ and is represented in table (5) of Fig. 3. In $SDB_{A,B}$ the 1-ITS $\langle([-1,1]D)\rangle$ is frequent and allows to identify the extended pattern $\langle([0,0]A)([1,3]BD)\rangle$. Then, $SDB_{A,B}$ is projected by $\langle([-1,1]D)\rangle$ and the result is represented in table (8) of Fig. 3. It does not contain any frequent event. The extraction process extends then the pattern $\langle([0,0]A)\rangle$ by $\langle([2,2]D)\rangle$.

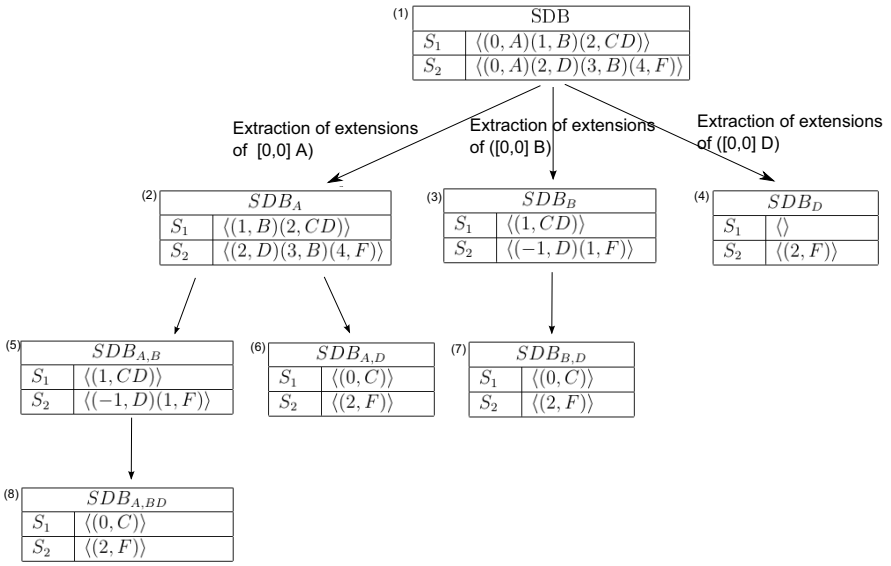


Fig. 3 Extraction steps during the processing of ITS-PS over *SDB*

$\langle([0,0]A)\rangle \oplus \langle([2,2]D)\rangle = \langle([0,0]A)\rangle \oplus_T \langle([2,2]D)\rangle = \langle([0,0]A)([2,2]D)\rangle$ is identified and the projection of SDB_A by $\langle([-1,1]D)\rangle$ is calculated. The result of this last projection is illustrated on Table (6) of Fig. 3. In this sequences database, B doesn't appear because it precedes D wrt the lexicographic order and all its occurrences are close to D in the sequences of the projected database. So in $SDB_{A,D}$ there is no frequent ITS. At this stage of the extraction process, all patterns extending $\langle([0,0]A)\rangle$ are identified. Now the extraction process extends the pattern $([0,0]B)$. First, the initial database (table (1) of Fig. 3) is projected by the pattern, the resulting search space denoted by SDB_B is illustrated in table (3). In this search space the item A does not appear because its close to B w.r.t *ws* and $A \lhd B$. In SDB_B, the 1-ITS $([-1,1]D)$ is frequent and is used to extend $([0,0]B)$. It identifies the pattern $([0,2]BD)$ by adjusting the time reference of both patterns to the smallest timestamp. The extraction projects SDB_B by $([-1,1]D)$, the resulting search space $SDB_{B,D}$ is illustrated in

the table(7) of Fig. 3. $SDB_{B,D}$ does not contain any frequent 1-ITS. The extraction process extends the $[0,0]D)$ pattern. *SDB* is projected by $([0,0]D)$ to obtain SDB_D illustrated in table(4) Fig. 3. It does not contain any frequent 1-ITS to extend the pattern. Here the extraction process is done.

This section presented our algorithm *ITS-PS*. It extracts $(i+1)$-sequences from an i-sequence by progressive reductions of the search database following a pattern growth procedure. The temporal intervals are built by using the sliding window on two levels of the extraction process: the identification of frequent events and the search space projection.

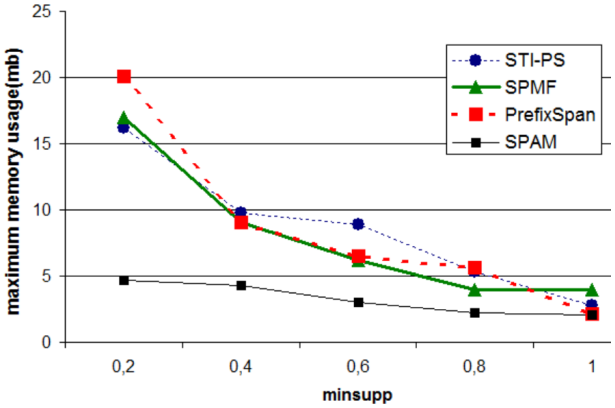
5 Experiments

In this section we evaluate the performances of the *ITS-PS* algorithm. In a first paragraph we analyze its computation time and memory consumption. They are compared with those of other FP-growth algorithms: *PrefixSpan*, *SPMF* and *SPAM*. *PrefixSpan* [Pei et al., 2001] is the pathfinder algorithm of the FP-growth extraction approach which perform a divide and conquer extraction. The *SPMF* algorithm [Fournier-Viger et al., 2008] is an Fp-growth algorithm that applies a time grouping constraint and *SPAM* algorithm [Ayres et al., 2002] enforce bitmap representation for sequences database. In a second paragraph we study the relevance of the *ITS* extracted patterns and compare them with patterns extracted by the *GSPM* algorithm.

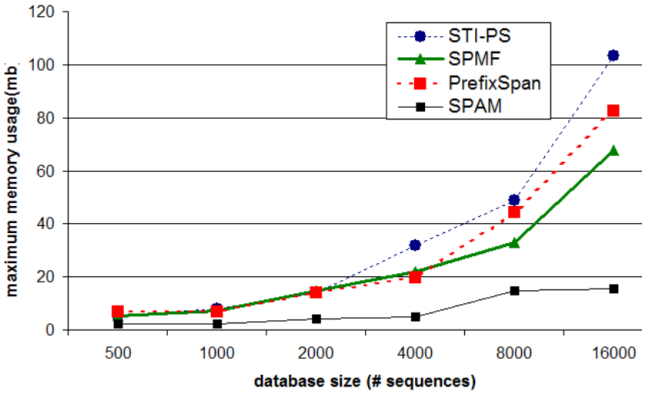
From a theoretical point of view, there is no hope to come up with an extraction algorithm having a worst case complexity less than exponential w.r.t. the number of events appearing in the mined data since the number of returned sequences may itself be in exponential size. Therefore, our algorithm has an exponential worst case complexity.

In order to work over the performance evaluation of the *ITS-PS* algorithm, we analyze in the following its computation time and memory consumption. For this proposal, we compare both criterion to those of other *FP-growth* algorithms (*PrefixSpan*, *SPMF* and *SPAM*) by varying the support threshold and the sequences database size. We use randomly generated data.

Fig. 4 shows the maximum memory consumption of *ITS-PS* compared to those of the algorithms SPMF [Fournier-Viger et al., 2008], PrefixSpan [Pei et al., 2001] and SPAM [Ayres et al., 2002]. Considering the support threshold variation (Fig. 4a), the memory consumed by ITS-PS is similar to that of SPMF and PrefixSpan unless for a 0.6 support value since data events are especially close and frequent. Considering the database size variation (Fig. 4b), the memory consumption behavior of *ITS-PS* algorithm is similar to memory consumed by the other studied algorithms, except that for larger database the memory consumption of SPAM is especially important because of the high cost of database bitmap transformation. So, Fig. 4a and Fig. 4b show that despite the extended projection of *ITS-PS*, the memory consumption of our algorithm has the same trend that the other "classical" *FP-growth* algorithms (except *SPAM* algorithm that use bitmap representation).



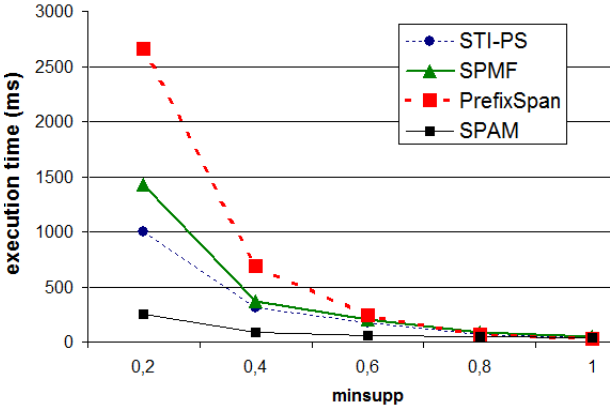
(a) Maximum memory consumption vs support variation



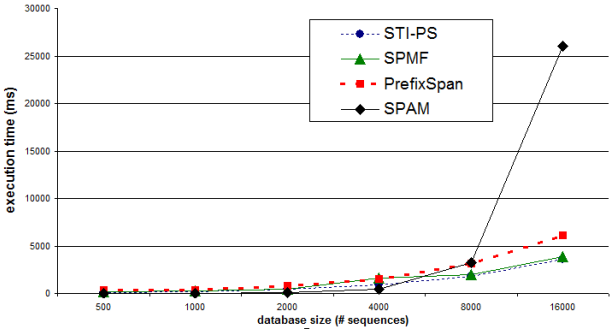
(b) Maximum memory consumption vs database size

Fig. 4 Maximum memory consumption

Fig. 5 displays computation time behaviors of *ITS-PS* compared to those of the algorithms *SPMF*, *PrefixSpan* and SPAM with regards to support threshold variation (Fig. 5a) and database size variation (Fig. 5b). Considering the support variation, Fig. 5a shows that time consumption of *ITS-PS* outperforms those of "classical" *FP-growth* algorithms. However, it is higher than the time consumption of SPAM algorithm, since bitmap representation reduces considerably time consumption. Considering the database variation, Fig. 5b shows that for larger databases ITS-PS needs big execution time since larger projection computation is costly. We can say that the time consumption of the *ITS-PS* algorithm is regular regards to *SPMF* and PrefixSpan ones. We can conclude that execution performances of *ITS-PS* algorithms are regular with respect to the performance of classical *FP-growth* algorithms. Since the extension of projections space results does not affect the cost of the algorithm.



(a) computation time vs support variation



(b) computation time vs database size

Fig. 5 Computation time

In the rest of this section, we evaluate the relevance of the extracted patterns. In the following, we compare patterns extracted by our method with those extracted by *GSPM* presented in [Hirate and Yamana, 2006]. Both algorithms are based on the *PrefixSpan* method. They are different because of the application of distinctive grouping methods: *GSPM* is based on the application of an increment function unlike *ITS-PS* which uses a sliding window. For a meaningful comparison, when the sliding window is fixed to a *ws* value, the *GSPM* step function is set to $f(t) = \lfloor 1/ws \rfloor$. The following example explains the *GSPM* process. More details can be found in the original paper.

Both *GSPM* and *ITS-PS* look for frequent 1-pattern associated with time intervals on the projected databases. However, while time intervals identified by *GSPM* are defined by a step wise function, those looked by *ITS-PS* are defined by merging close occurrences of a frequent item. Thus, for *GSPM* an occurrence of an event e at a timestamp t denoted by (t, e) can be associated with a single 1-pattern.

This timestamp is equal to $f(t)$. However, for *ITS-PS*, a such event may be associated with as many intervals as possible in the margin $[t-ws, t+ws]$ as item e is fairly frequent in this period. A 1-patterns selection such that of *ITS-PS* allows us to broaden the number of continuation possibilities of a pattern by associating a frequent item to a full pallet of intervals.

Example 12. Consider the database $\{S_1 = \langle(0,A)(1,B)\ (2,C)(3,F)(4,B)\ (6,G)\rangle$, $S_2 = \langle(0,A)(1,C)(2,B)(3,D)(4,F)(5,G)\rangle\}$, a threshold support $minsupp = 2$, a sliding window $ws = 2$ and a step function $f(t) = \lfloor t/2 \rfloor$. Timestamps interval provided by *GSPM* are in the form $[2 \times f(t), 2 \times (f(t)+1)[$. The extraction algorithm first identifies the frequent 1-sequences A, B, C, F and G (They are timestamped with null intervals). If we consider the frequent B, the projection provides: $\{S'_1 = \langle(1,C)(2,F)(3,B)(5,G)\rangle$, $S''_1 = \langle(2,G)\rangle$, $S'_2 = \langle(2,F)(3,G)\rangle\}$. In this search space, the pattern $([2,4[,F)$ is identified as frequent since (1) F appears twice: in S'_1 and in S'_2 and (2) for the both occurrences, $f(t) = \lfloor t/2 \rfloor = 1$. Then, in order to identify the interval timestamps to be associated with F, we apply $[2.f(t), 2.(f(t)+1)[$ which provides $[2,4[$. In the same projection, G appears in 3 sequences. In S''_1 and S'_2 with $f(t) = 1$, while in S'_1 its function step value corresponds to $f(t) = 2$. So, only the 1-sequence $([2,4[,G)$ is extracted and $([4,6[,G)$ is not considered so.

Both algorithms are implemented in JAVA using a *PrefixSpan* version[1] proposed in [Fournier-Viger et al., 2008]. The implementation is done on a Windows 7(64) machine, Intel(R) Core(TM) 3 CPU 2.40 GHz with 3 GB RAM.

We compare both extraction results using synthetic data. Data sequences have 7 different events, the average deviation between strictly successive transactions is equal to 3 time units and a sequence average length is equal to 15 transactions. During extraction executions the time constraints *mingap* (respectively *maxgap*, *min_whole_interval* and *max_whole_interval*) are fixed to 0 (resp. 1, 0 and 15). Synthetic sequences database contains 12 sequences since we focus our experimentation on the nature and the number of results that is why we choose a small sequences database. In the following, our goal is focused on validating our algorithm by checking the relevance of its extracted frequent patterns regards to our interested application domain. Considering the time constraints relaxation employed by our approach, we expect that the *ITS-PS* algorithms provide more information because of two point: (1) first, we use the extended projection which is larger than the projection used on *GSPM* and than offer a larger range of continuities by considering the backward projection. (2)Second, the *ws* relaxation allows to group a larger palette of timestamps to identify the *1-ITS*. A such frequent selection provides more options of time intervals and more patterns and the extraction process is deeper.

Fig. 6, Fig. 7, Fig. 8 and Fig. 9 show the number of returned patterns by both algorithms regards to support variation for different values of merging parameters (*ws* and $f(t)$). for each parameter configuration, we measure the number of returned patterns by each algorithm and compute the maximal ones. Fig. 6 (respectively Fig. 7, Fig. 8 and Fig. 9) show that the amount of $ITS-PS$ results is greater than the

[1] `http://www.philippe-fournier-viger.com/spmf/index.php`

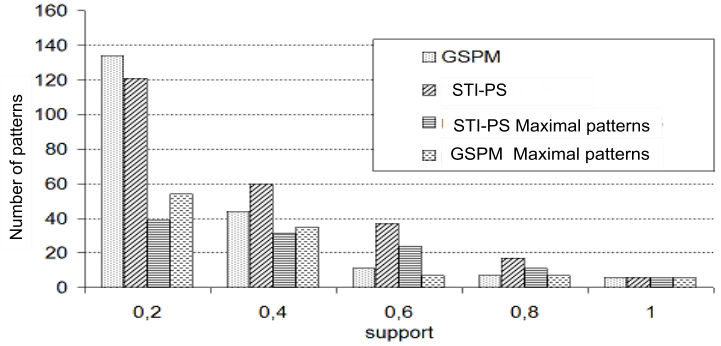


Fig. 6 Number of extracted sequences by varying *minsupp*, *ws* and the step function (WS=1, $f(t) = \lfloor t/1 \rfloor$)

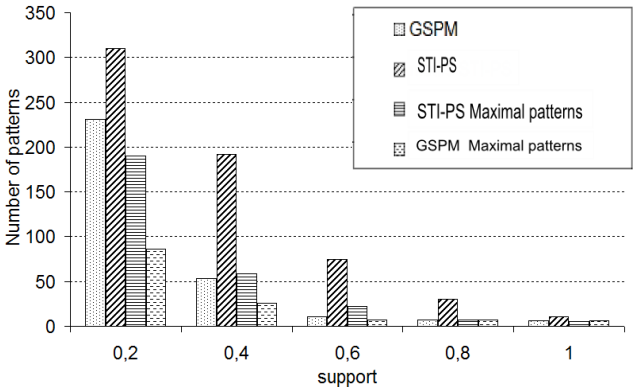


Fig. 7 Number of extracted sequences by varying *minsupp*, *ws* and the step function (WS=3, $f(t) = \lfloor t/3 \rfloor$)

amount of *GSPM* result. Actually, the application of the sliding window gradually groups successive transactions and considers all possible merging combinations. It also allows longer sequences extraction since more events combination are considered frequent from the data sequences and the extraction process stop 'later'. On the other hand, the backward projection employed by *ITS-PS* takes into account more continuation possibilities and so some events see their support growing up.

Considering an aeronautics's historical sequences where each transaction relates flight parameters by indicating (1) the hauled distance done which can be *high.haul*, *med.haul* and *low.haul*.(2) the filling degree of the plane: $Pfull.fill$, $Pmed.fill$, and $Plow.fill$ indicate how full is the plane (3) the third flight parameter is the crossed environment which can be: salt, sand. If this last parameter is normal no information are mentioned. Table 5 presents patterns extracted from such sequences database by both algorithms *GSPM* and *ITS-PS*. The outstanding difference between patterns

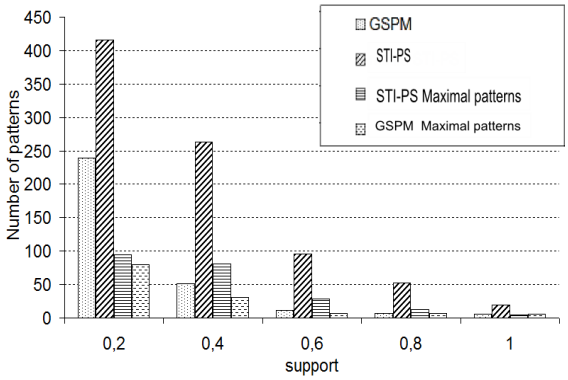


Fig. 8 Number of extracted sequences by varying *minsupp*, *ws* and the step function (WS=5, f(t)= $\lfloor t/5 \rfloor$)

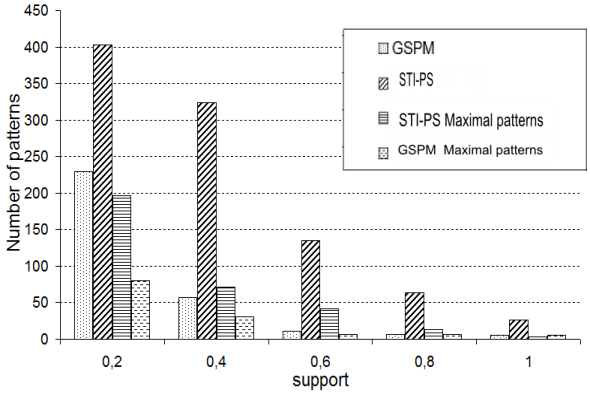


Fig. 9 Number of extracted sequences by varying *minsupp*, *ws* and the step function (WS=7, f(t)= $\lfloor t/7 \rfloor$)

extracted by both algorithms is the timestamp representation. Actually, patterns extracted by *GSPM* are timestamped with discrete values that represent predefined intervals (by the step wise function). However, Patterns extracted by *ITS-PS* convey more flexible temporal interval information where intervals may be narrower than *ws* and cover sliding span. This last point helps us to perform the accuracy of events prediction and provides more precise time laps occurrence.

In Table 5, transactions of the *ITS-Ps* patterns have overlapped intervals, such a representation is made since the *mingap* constraint is aborted. Also, we choose to not apply merging for $\oplus_T$ when intervals are not equal in order to preserve time precision and reduce uncertainty.

The *ITS* patterns convey more realistic time information as regards to data behavior. This information can be handled in order to regulate time precision and uncertainty.

Table 5 Example of patterns extracted from an aeronautical data sequence

GSPM patterns	$\langle(0, sand)(1, verification.mot)\rangle$ $\langle(0, Plow.fill\ med.haut)(1, sand)\rangle$
ITS-PS patterns	$\langle([0,0], Plow.fill)([1,4], long.haul)([4,4], sand)([4,7], verification.mot)\rangle$ $\langle([0,3], Plow.fill)([2,2], med.haut)([5,5], verification.mot)\rangle$ $\langle([0,1], Plow.fill)([3,6], med.haut)([5,7], verification.mot)\rangle$

Table 6 Number of extracted i-sequences (L_i) by varying the window size, the step function depth and fixing *minsupp* to 0.4

	maximal GSPM patterns			ITS-PS maximal						ITS-PS patterns				
ws	L_1	L_2	L_3	L_1	L_2	L_3	L_4	L_5	L_6	L_2	L_3	L_4	L_5	L_6
1	13	21	1	17	39	14	0	0	0	21	0	14	0	0
2	11	16	5	7	47	44	3	0	0	3	19	3	0	0
3	9	12	5	7	53	96	26	3	0	0	30	26	3	0
4	8	12	6	7	53	108	62	9	1	0	23	34	8	1
5	9	13	2	7	55	133	75	9	1	0	26	42	13	1
6	9	14	4	7	52	98	88	38	7	0	26	20	27	7
7	9	19	3	7	51	115	88	21	4	0	24	29	12	4

Table 6 details the number of k-patterns extracted by *ITS-PS* and *GSPM* for a fixed *minsupp* value (equal to 0.4) and different grouping values. We notice that when both methods provide the same patterns length results (correspondence between Fig.6 and Table 6), maximal sequences extracted by our approach are fewer than maximal patterns obtained by *GSPM*. Such situation is illustrated in Example 13. However, when the sequences returned by *ITS-PS* are longer than those provided by *GSPM*, *ITS-PS* maximal sequences are more than *GSPM*'s ones and majority represent longer patterns then those from maximal *GSPM* result. Finally, notice that the number of maximal sequences extracted by our approach is still similar to those extracted by *GSPM*.

Example 13. If we consider Example 12 then the longest maximal sequences extracted by *GSPM* are: $\langle([0,0[, B)([2,4[, F)\rangle$, $\langle([0,0[, G)\rangle$, $\langle([0,0[, A)\rangle$, $\langle([0,0[, C)\rangle$. The only one extracted by *ITS-PS* is $\langle([0,2], ABC)([3,4], F)([5,6], G)\rangle$. The sequence $\langle([0,0[, B)([2,4[, F)\rangle$ extracted by *GSPM* means that "*F appears randomly in* $[2,4[$ *after B*". However, the sequence $\langle([0,2], ABC)([3,4], F)([5,6], G)\rangle$ provided by *ITS_PS* means, among others, that *F appears in the interval* $[3-2=1, 4-0=4]$ *after B*. Given that $[1,4]$ contains $[2,4[$, we can say that the maximal sequence

provided by our approach includes all maximal sequences extracted by *GSPM* by tolerating more uncertainty.

6 Conclusion

This paper presents *ITS-PS*, a sequences extraction algorithm based on the sliding window principle allowing time constraints relaxation. The sliding window gradually merges close transactions (co-occurring events) by considering several merging combinations. The algorithm extracts interval temporal sequences from a collection of discrete temporal ones. The interval timestamps express an uncertainty of the exact moment when transaction events occur. The uncertainty magnitude is managed by the size of the sliding window fixed by the user. The implementation of our algorithm is inspired by that of [Pei et al., 2001]. We compared qualitatively the results of our method to those provided by the *GSPM* algorithm proposed in [Hirate and Yamana, 2006]. It turns that our algorithm provides more and longer sequences than *GSPM* since result patterns convey more information from input data. Actually, when "local" events appear (in the data sequences) with an alternate order are met in the data sequences, *GSPM* (and other extraction algorithms) stops extension of the pattern. However, the *ITS-PS* algorithm extract the same kind of information as frequent and continues its extension. This makes this latter comparing to *GSPM* providing such additional patterns.

Future works will concern the optimization of maximal patterns extraction process. Indeed, due to our relaxation of the chronological sequence of event occurrences, we extract more sequences than other approaches. However, when we restrict our result to the maximal sequences, not only the amount of result is lower than that of the other approaches but it encompasses it. From a practical viewpoint, it is not relevant to first extract all patterns and then select the maximal ones. In order to cope with the huge data manipulated by our targeted industrial application (aeronautic maintenance tasks prediction), we are currently optimizing the present proof-of-concept implementation.

References

[Agrawal and Srikant, 1995] Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proceeding of ICDE Conference, Taipei, Taiwan, pp. 3–15. IEEE Computer Society Press (1995)

[Agrawal and Srikant, 1996] Agrawal, R., Srikant, R.: Mining sequential patterns: Generalizations and performance improvements. In: Apers, P.M.G., Bouzeghoub, M., Gardarin, G. (eds.) EDBT 1996. LNCS, vol. 1057, pp. 1–17. Springer, Heidelberg (1996)

[Agrawal et al., 1994] Agrawal, R., Srikant, R., et al.: Fast algorithms for mining association rules. In: Proc. 20th Int. Conf. Very Large Data Bases, VLDB, vol. 1215, pp. 487–499 (1994)

[Allen, 1983] Allen, J.F.: Maintaining knowledge about temporal intervals. Communications of ACM 26 (1983)

[Ayres et al., 2002] Ayres, J., Flannick, J., Gehrke, J., Yiu, T.: Sequential pattern mining using a bitmap representation. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada, pp. 429–435. ACM (2002)

[Ceci et al., 2011] Ceci, M., Loglisci, C., Salvemini, E., D'Elia, D., Malerba, D.: Mining spatial association rules for composite motif discovery. In: Mathematical Approaches to Polymer Sequence Analysis and Related Problems, pp. 87–109 (2011)

[Fournier-Viger et al., 2008] Fournier-Viger, P., Nkambou, R., Nguifo, E.M.: A knowledge discovery framework for learning task models from user interactions in intelligent tutoring systems. In: Gelbukh, A., Morales, E.F. (eds.) MICAI 2008. LNCS (LNAI), vol. 5317, pp. 765–778. Springer, Heidelberg (2008)

[Giannotti et al., 2006] Giannotti, F., Nanni, M., Pedreschi, D., Pinelli, F.: Mining sequences with temporal annotations. In: Proceedings of the 2006 ACM Symposium on Applied Computing (SAC), pp. 593–597. ACM (2006)

[Guyet and Quiniou, 2008] Guyet, T., Quiniou, R.: Mining temporal patterns with quantitative intervals. In: Workshops Proceedings of the 8th IEEE International Conference on Data Mining (ICDM), Pisa, Italy, pp. 218–227. IEEE Computer Society (2008)

[Guyet and Quiniou, 2011] Guyet, T., Quiniou, R.: Extracting temporal patterns from interval-based sequences. In: IJCAI, Barcelona, Catalonia, Spain, pp. 1306–1311 (2011)

[Hirate and Yamana, 2006] Hirate, Y., Yamana, H.: Generalized sequential pattern mining with item intervals. JCP 1(3), 51–60 (2006)

[Li et al., 2012] Li, C., Yang, Q., Wang, J., Li, M.: Efficient mining of gap-constrained subsequences and its various applications. ACM Trans. Knowl. Discov. Data 6(1), 2:1–2:39 (2012)

[Pei et al., 2001] Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.: Prefixspan: Mining sequential patterns by prefix-projected growth. In: Proceedings of the 17th International Conference on Data Engineering ICDE, pp. 215–224 (2001)

[Pham et al., 2009] Pham, Q., Raschia, G., Mouaddib, N., Saint-Paul, R., Benatallah, B.: Time sequence summarization to scale up chronology-dependent applications. In: EDBT 2008, 11th International Conference on Extending Database Technology, Hong Kong, China, pp. 1137–1146 (2009)

[Plantevit et al., 2007] Plantevit, M., Laurent, A., Teisseire, M., et al.: Extraction de motifs séquentiels multidimensionnels clos sans gestion d'ensemble de candidats. In: EGC 2007: Extraction et Gestion des Connaissances, p. 6 (2007)

[Rabatel et al., 2009] Rabatel, J., Bringay, S., Poncelet, P.: So_mad: Sensor mining for anomaly detection in railway data. In: Perner, P. (ed.) ICDM 2009. LNCS (LNAI), vol. 5633, pp. 191–205. Springer, Heidelberg (2009)

[Srinivasulu et al., 2010] Srinivasulu, P., Rao, J.R., Babu, I.R.: Network intrusion detection using fp tree rules. CoRR, abs/1006.2689 (2010)

[Wu and Chen, 2007] Wu, S., Chen, Y.: Mining nonambiguous temporal patterns for interval-based events. IEEE Trans. on Knowl. and Data Eng. 19, 742–758 (2007)

[Yi-Cheng et al., 2010] Yi-Cheng, C., Ji-Chiang, J., Wen-Chih, P., Suh-Yin, L.: An efficient algorithm for mining time interval-based patterns in large database. In: ACM, Proceedings of CIKM Conference, Hong Kong, China, pp. 49–58 (2010)

Semantic Event Extraction from Biological Texts Using a Kernel-Based Method

Rim Faiz, Maha Amami, and Aymen Elkhlifi

Abstract. As research into protein and gene interactions continues to produce vast amount of data, concerning to biological event, there is an increasing need to capture these results in structured formats allowing for computational analysis. Although many efforts have been focused to create databases that store this information in computer readable form, populating these sources largely requires a manual process of interpreting and extracting biological event templates from the biological research literature. Being able to efficiently and systematically automate the extraction of biological events from unstructured text, would improve the content of these databases, and provide methods to collect, maintain, interpret, curate, and discover knowledge needed for research or education. Hence, it is important to have an automated extraction system to extract events from biological texts. In this paper, we present an automated information extraction approach, to identify biological events in text. Our approach is based on, identifying event triggers and extracting event participants by using a kernel learner that operates on dependency and semantic information to calculate similarity between feature vectors.

Rim Faiz
LARODEC University of Carthage, IHEC 2016 Carthage Presidence, Tunisia
e-mail: Rim.Faiz@ihec.rnu.tn

Maha Amami
LARODEC University of Tunis, ISG 2000 Bardo, Tunisia
e-mail: Amami.Maha@ymail.com

Aymen Elkhlifi
LaLIC, Université Paris-Sorbonne, 28, rue Serpente, 75006 Paris, France
e-mail: Aymen.Elkhlifi@paris4.sorbonne.fr

F. Guillet et al. (eds.), *Advances in Knowledge Discovery and Management,*
Studies in Computational Intelligence 527,
DOI: 10.1007/978-3-319-02999-3_5,

1 Introduction

The biological information is growing explosively. The experimental and computational results are appearing daily in scientific publications. The MEDLINE[1] database contained in 2013 over 22 million articles, and this database is currently growing; about 500 000 new papers are added each year [Mitchell et al., 2003].

However, extraction of useful information from these online sources is difficult due to the lack of formal structure in the natural language in biological texts [Mukherjea et al., 2004].

For instance, using keyword queries that retrieve a large set of relevant papers, scientists can navigate through hyperlinks between genome database and referenced papers. To extract the requisite knowledge from the retrieved papers, they must identify the relevant information. Such manual processing is time consuming and repetitive, because of the bibliography size, and the database continuous updating. From the MEDLINE database, the query *"Bacillus subtilis and transcription"* which returned 2209 abstracts in 2002, retrieves 3942 of them today.

As a result, there has been an increased interest in the application of information extraction techniques to support database building and to intelligently find knowledge in documents.

The biological information extraction is a set of techniques that extract the essential biological information through analysis of scientific texts and represent them as a template whose slots are filled on the basis of what is found from the text.

In the past decade, most of the efforts on biological information extraction have been focused on the task of recognizing entity names in text such as genes or proteins names and on the extraction of relations of these entities. Thus, several information extraction systems have been developed for detecting interaction information from texts. Most previous efforts have focused on protein-protein interactions. For instance, the GENIES system [Friedman et al., 2001] extracts bio-molecular interactions relevant to signal transduction and biochemical pathways. The extracted relationships between proteins are encapsulated in interactions with common type-value frames. The RelEx system [Fundel et al., 2006] looks for relations using parse trees and a set of rules from one million MEDLINE abstracts.

Recently, the focus of research has been moving to higher levels of information extraction such as co-reference resolution and event extraction [Faiz, 2006]. We are interested in event extraction task particularly biological event extraction task which involves the filling of event templates from biological texts (e.g. *phosphorylation of TRAF2*; *Type:Phosphorylation, Theme:TRAF2*).

The paper is organized as follows: Section 2 presents methods to extract events from texts. Section 3 describes our event extraction approach. Section 4 reports the implementation of our event extraction system and experiments.

[1] `http://www.ncbi.nlm.nih.gov/pubmed/`

2 Methods for Extracting Events from Biological Texts

The task of extracting biological events refers to the task of detection of event templates using basic tools from biological texts.

A biological event template is specified by a trigger and arguments. The semantic roles are assigned to these arguments [Kim et al., 2009].

For instance, in the sentence *"5-LOX is expressed in leukocytes."*, the event is characterized by a verb *"expressed"*, and the argument is *"5-LOX"* which is tagged by the semantic role *"Theme"*.

Table 1 shows examples of event types. For each primary (obligatory) event argument, the role of the argument *(Theme, Cause)* is shown, with the possible argument filler type shown in parentheses (P, protein; Ev, event). Binding events can take an arbitrary number (+) of proteins as primary arguments, which form protein complexes.

Table 1 Examples of event types

Event type	Examples	Arguments
Gene expression	5-LOX is **expressed** in leukocytes	T(P)
Transcription	promoter associated with IL-4 gene transcription	T(P)
Localization	nuclear **translocation** of STAT6	T(P)
Protein catabolism	I kappa B-alpha **proteolysis** by phosphorylation	T(P)
Phosphorylation	BCL-2 was **phosphorylated** at the G(2)/M phase	T(P)
Binding	Theme (Protein)	
Regulation	c-Met expression is **regulated** by Mitf	T(P/Ev), C(P/Ev)
Positive regulation	IL-12 **induced** STAT4 binding	T(P/Ev), C(P/Ev)
Negative regulation	DN-Rac **suppressed** NFAT activation	T(P/Ev), C(P/Ev)

Several techniques to extract events from biological texts are currently being used extensively: full parsing, pattern marching, machine learning and ontology-driven approaches.

Yakushiji et al. [Yakushiji et al., 2001] introduce an information extraction system based on full parsing and a set of rules. Given a list of target verbs, an argument structure extractor applies full parsing to input texts and extracts argument structures. The argument structures are then transformed to frame representations using domain-specific mapping rules.

The work of Vlachos et al. [Vlachos et al., 2009] belongs to this family, using a domain independent parser to get sets of head-dependent grammatical relations that connect an event trigger with an appropriate argument like *VERB-TRIGGER-subject-ARG, NOUN-TRIGGER-iobj-PREP-dobj-ARG*.

Full parsing implies reference to a theory of syntax. For instance, [Hakenberg et al., 2009] apply Link Grammar theory [Sleator and Temperley, 1993] to the extraction of biological event. The Link Grammar (LG) parser is a deep syntactic

parser based on the link grammar theory, which consists of a set of words and linking requirements between words. The work [Hakenberg et al., 2009] is based on the the BioLG parser described in [Pyysalo et al., 2004], which modifies the original parser (LG parser) by extending its dictionary and by adding more rules for guessing structures when facing unknown words. A parse tree database stores the output of the parser on arbitrary texts. Parse trees are accessed by a query language, called PTQL. The PTQL query describes the hierarchical structure of a parse tree and the linkage of the dependencies between words.

The pattern-based approaches try to use context information for finding biological events. They usually look for certain words occurring near entity names or use part-of-speech (POS) and/or syntax information and semantic information. Hence, patterns can be written using dictionaries, preposition based parsing and so forth.

An example of dictionary-based approach is provided by [Buyko et al., 2009] who build a dictionary of event trigger verbs and their associated event classes ranked according to their frequency in the training data. The final set of candidate event triggers is selected based on the importance degree of an event trigger for an event class. Another technique of pattern matching is the preposition parsing which is presented by Leroy et al. [Leroy and Chen, 2002]. The event extraction templates are filled with parsed material surrounding prepositions such as "by" and "of" which are often cue strings of *Theme* or *Cause* roles.

For example, the sentence *"apoptosis induced by the p53 tumor suppressor"* contains preposition *"by"* which mentions using parse trees and hand-coded templates, *"p53 tumor suppressor"* as *Cause* argument, *"apoptosis"* as *Theme* argument and *"induced"* as an event trigger verb.

The work by Björne et al. [Björne et al., 2009] applies SVM to detect biological events using a set of features and semantic networks derived from full dependency analysis. Thus, they represent each sentence in term of graph where the nodes correspond to protein and event triggers and edges correspond to event arguments.

The event nodes are formed by the prediction of individual tokens, and event edges are identified by predicting for each trigger-trigger or trigger-named entity pair whether it corresponds to an appropriate event argument. The features used in the SVM model include the morphological properties of the token to be classified, such as character bigrams and trigrams, and tokens that depend on it, the number of named entities and the bag of word of token counts in the sentence. For a given class, the SVM model calculates the confidence score of a token belonging to the class. After event trigger detection, all potential edges, which connect an event node to another or to a named entity node, are classified using SVM classifier as a *Theme*, *Cause* or negative class. The set of edge features is built by combining the attributes of tokens, the n-grams which define the variation of dependency directions, the node features which combine the token features of the two terminal event or entity node of the potential edge, individual component features which combine a token or an edge attribute with the token or edge position at either the interior or the end of the path.

Neves et al. [Neves et al., 2009] use the case-based reasoning (CBR) approach to extract events. For extracting triggers, first, documents of the training set are

tokenized and the resulting tokens are saved in a base of cases. In the training step, tokens are mapped to cases of features: the token itself, the stem of the token, the part-of-speech tag, the chunk tag, the biomedical entity tag, the term type and the event type. During the testing step, the unknown feature (the event type) is inferred. For each token from the test set (a case problem), the method proceeds by retrieving the most similar cases with the higher number of features that have exactly the same values of the case problem respective features. The best case solution will be the one with the higher frequency. The argument detection consists of post-processing rules for each type of argument to map case solutions (event trigger) to its respective arguments. The method starts from the event trigger to search in both directions for extracted arguments. For example, the *Theme* detection starts from the event trigger in both directions until a *Theme* argument is found in the sentence or reaching 20 tokens in each direction.

The ontology-driven approaches include those that attempt active use of the ontology in processing to strongly guide and constrain analysis. A good example of an ontology-driven system, which primarily targets events is presented by [Cohen et al., 2009]. They apply the OpenDMAP semantic parser with manually written-patterns. OpenDMAP is an ontology-driven integrated concept analysis system that supports information extraction through the use of patterns presented in a form of "semantic grammar". The patterns characterize the linguistic expression of that event and identify the arguments of the events according to occurrence in relevant linguistic context and satisfaction of appropriate semantic constraints as defined by ontology. The reference ontology combines elements of several ontologies available in the biomedical domain - Gene ontology (GO), Cell Type Ontology (CTO), BRENDA Tissue Ontology (BTO), etc. - and additional concepts to formally define entities, events, and constraints on slots. Concepts are represented as frames, with slots for the event trigger words and the various slot fillers which are constrained in the ontology to be of type protein from the Sequence Ontology while the type of the other event arguments varied. For instance, the binding argument of a binding event is constrained to be one of binding site, DNA, domain or chromosome. The method obtains a recall rate of about 13.45% and a precision rate of about 71.81% for the BioNLP'09 Shared Task on Event Extraction, which is the best among all reported results.

Many of the above mentioned approaches do not exploit any forms of semantic understanding, treating texts as dependency trees. For instance, the work of Björne et al. [Kim et al., 2009] represents the sentence structure as a syntactic graph.

While a type of an event is a property of an individual token, we should note that the trigger detection requires looking at the context in which token is used. The context indicators surrounding the candidate trigger hold semantic information in a sentence. However, the choice of a particular indicator is a dependent task.

In event extraction task the most commonly used features are string based features. One approach of using such string features in SVM classifier is to use the binary feature vectors [Kim et al., 2009], where a particular feature is converted into several binary values. For example, the feature "POS" is converted into N binary features where N is the total number of unique POS in the lexicon. However,

the kernel function applicable on binary features is not able to capture the specific similarity between the features.

3 Event Extraction Approach

We aim at developing a new and effective method for detecting biological events from the literature. Our proposal is to generate automatically a wide number of features, and use SVM classifier with a suitable kernel function able to capture the specific similarity between these features.

The whole process is described in Figure 1.

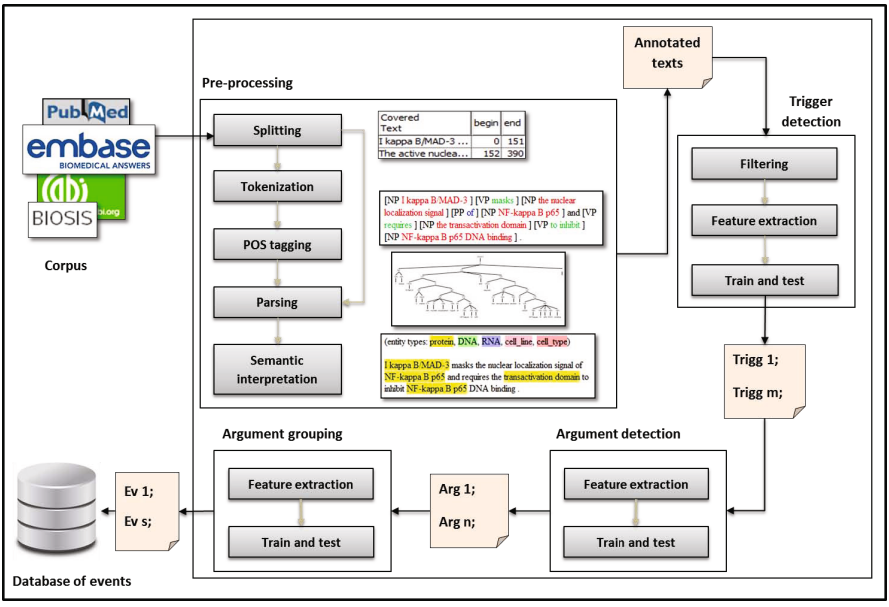


Fig. 1 The whole process

3.1 Pre-processing

For extracting events from text, we employ many Natural Language Processing (NLP) techniques. We apply state-of-the-art systems trained on biological corpora for splitting, tokenization and part-of-speech tagging. Then, we use parsers to analyze the syntactic relations among the entities in the sentence. Finally, the syntactic analysis is complemented by a semantic processing, a step which assigns semantic classes (e.g., gene, protein, cell type, etc.) using semantic resources.

3.1.1 Splitting

First, we proceed by splitting documents of the training and development data set. The most common way to identify sentence boundaries is with manually defined patterns that rely on the fact that, i.e., a period is followed by an uppercase letter. In contrast, there are more sophisticated approaches based on the structure of the sentence and the punctuation presented in [Elkhlifi and Faiz, 2010].

In biological domain, compared to the targets of traditional information extraction such as newspaper articles, the structure of a sentence tends to be more complicated: sentence boundaries are much more ambiguous because they often appear within biological entities, as well as within formulae, bibliographic references, etc. Hence, we use the state-of-the-art splitter optimized for the biological corpora, namely the GENIA sentence splitter GENIASS[2].

GENIASS [Rune et al., 2007] is based on a supervised leaning method using maximum entropy modeling including a set of features: delimiters of the candidate boundary, previous/next word, presence of capitals in the previous word, presence of comma, brackets, quotations, numbers in previous or next word. First, it detects candidate positions for splitting using selected delimiters: periods, commas, single or double quotation marks, right parentheses, etc. Then, it classifies whether each candidate is a sentence boundary or not.

3.1.2 Tokenization

It is the process of breaking the sentence up into linguistic units, known as tokens (e.g., words, acronyms, abbreviations, numbers, punctuation symbols, and so forth). There exist many algorithms to identify tokens. They can vary from simple algorithms, separating tokens by white spaces and punctuation, to slightly more sophisticated techniques, such as using finite-state regular expression matching and lexicon-based approaches, to address the problem of abbreviations, apostrophes and hyphenation.

In the biological domain, there are some remaining challenges in tokenization, due to domain-specific terminology, nonstandard punctuation and orthographic patterns (e.g., an alpha-galactosyl-1,4-beta-galactosyl-specific adhesin or the free cortisol fractions were 4.53 / 0.15% and 8.16 / 0.23%).

3.1.3 POS Tagging

We proceed with the assignment of a part-of-speech class (e.g., noun, verb, adjective, preposition, number, and proper noun) to terms in a document. Several approaches exist to POS tagging. For example, given an annotated POS corpus and a small set of lexical and contextual patterns, Brill's tagger [Brill, 1995] proceeds by iteratively proposing patterns, comparing the results of this pattern application to the annotated corpus, updating patterns to avoid mistakes.

[2] `http://www-tsujii.is.s.u-tokyo.ac.jp/y-matsu/geniass/`

3.1.4 Parsing

It is the process of determining the syntactic structure of a sentence. The full parsing establishes relations between the organizing verb and its dependent arguments [Ananiadou et al., 2010]. This is why it is commonly used to parse biological texts [McDonald et al., 1995; Yakushiji et al., 2001]. In our work, we use the McClosky Charniak domain adapted parser [McClosky and Charniak, 2008a] which is among the best performing parsers trained on the GENIA Treebank corpus. The output of the parser is transformed to the "collapsed" form of the Stanford dependency scheme [Marneffe and Manning, 2008] using the Stanford parser tools.

3.1.5 Named Entity Recognition

We identify biological terms in the scientific literature and annotate terms with their semantic classes (or concepts) by doing a lookup on various publicly available biomedical dictionaries, such as Unified Medical Language System (UMLS) Metathesaurus. As an example from semantic interpretation, we may annotate "to inhibit [something] and transcribe [something]" with "to inhibit [Biological process] and transcribe [Nucleic acid]".

Table 2 shows our example sentence with POS tags to each token. The named entity recognizers classify a given token with a semantic type of real-word entities (e.g., gene, protein, etc.).

Table 2 Pre-processing output

Token	POS tag	Type
apoptosis	NN	programmed cell death
induced	VBN	natural process
by	IN	-
the	DT	-
p53	NN	protein
tumor	NN	protein
suppressor	NN	protein

3.2 Trigger Detection

The event trigger detection is the task of identifying individual words in the sentence that acts as an event trigger word and assigning the correct event class to each of the determined triggers. We proceed with extracting a set of features for each candidate trigger based on both the context in sentence and the dependency parse. Then, we attempt to classify candidate triggers into event classes, including a negative event class.

3.2.1 Filtering Out Candidate Triggers

Because the set of tokens includes too many negative classes, we perform filtering step before trigger detection. We filter out tokens, that are: a named entity and whose POS tag is not a singular noun (NN), a plural noun (NNS), a verb in base form (VB), a verb in past tense (VBD), a verb in gerund or present participle (VBG), a verb in past participle (VBN), an adjective (JJ) and a comparative adjective (JJR) and sentences that do not have any proteins.

3.2.2 Trigger-Based Features Extraction

We use the output of the preprocessing step to construct feature vectors for the machine learning algorithm. They are considered as distinguishing token attributes. Our features are similar to those used in [Vlachos et al., 2009] and we create new ones (cf. Table 3).

Table 3 Features for trigger detection

Type	Feature
Token features	Token word
	Stem from the Porter stemmer [Humphreys et al., 2000]
	Lemma from the Natural Language Toolkit (`http://www.nltk.org/Home`)
	Token POS
	Capitalization
	Presence of symbol
	N-gram (n = 2, 3) characters
	Indicator whether the token is a stop word
	Presence of an adjacent verb or noun
	Presence in the trigger gazetteer
	Semantic type
Frequency features	Number of named entities in the sentence
	Number of stop words in the sentence
	Bag of word counts of token words in the sentence
	TF-IDF score of token word in the training set
Dependency features	Set of dependency chain features up to depth of three
Shortest path features	Dependency label path to the nearest protein
	N-grams of dependencies (n = 2, 3, 4)
	N-grams of words (n = 2, 3, 4)
	Length of the shortest path
	Presence of some token along the shortest path in the trigger gazetteer

Note that we prepare a gazetteer of trigger stems derived from the training set. Then, we extended it with corresponding WordNet synsets and using the pattern induction method described below. Thus, we add as features the presence of the token in the trigger gazetteer and the presence of some token along the shortest path.

The largest number of features comes from the dependency parse. For this reason, we add a number of features that aim to capture the full semantic context of candidate trigger. We construct entity context patterns and we extend the gazetteer of trigger stems as presented in [Talukdar et al., 2006]. The process is detailed in the following:

1. Extracting context. We find occurrences of seed list entities in the corpus. For each such occurrence, we extract a number W (context window size) of immediate neighbors on both the left and the right hand sides. Then we replace all entity tokens by the single token "-ENT-". Examples of extracted contexts are shown below:

   increased -ENT- of CAD in vad mice
   the -ENT- of insulin-like growth factor 2 mrna was greater
   -ENT- of the he nitric oxide synthase gene in mouse

2. Trigger word selection. The trigger words mark the beginning of a context pattern. The selected trigger word should be frequent in the extracted context and specific to event trigger of interest. Hence, we use the TF-IDF weight to rank candidate trigger words. The TF-IDF weight f_w for a word w occurring in a corpus is defined by,

$$f_w = \log\left(\frac{N}{n_w}\right) \tag{1}$$

   where,

   - N: number of documents in the corpus,
   - n_w: number of documents containing w.

   Then for each context segment, we select the dominating word d_c, which have a high TF-IDF.
3. Automata induction. We summarize contexts sharing the same trigger word into a pattern automaton with transitions that match the trigger word and also the wildcard -ENT-.

3.2.3 Training and Testing Trigger Detection

We perform the event trigger detection using a machine learning classifier. Each token vector is assigned to an event class or a negative class if it does not belong to an event trigger. However, traditional machine learning classification techniques perform poorly when working directly because of the high dimensionality of the

data. Thus, we use the kernel-based method SVM which scales relatively well to high dimensional data.

However, one of the major challenges in kernel-based method is the choosing of a suitable kernel function for the given classification problem [Kim et al., 2009]. In fact, there are standard choices such as a gaussian or polynomial kernel that are the default options, but they prove ineffective to train the classifier with large data sets. In the work presented by Björne et al. [Vlachos et al., 2009], the linear kernel function is used in trigger detection with large training sets. It computes the dot product between instances defined as follows:

$$\begin{aligned} K(X,Y) &= \phi(X) \cdot \phi(Y) \\ &= X^T \cdot Y \end{aligned} \tag{2}$$

where $\langle x_i, y_i \rangle = 1$ if x_i and y_i are the same and 0 otherwise.

However, when we are dealing with string features, such dot product based similarity computation is not able to capture the specific similarity between features.

In this study, we use different kernel functions to compute the similarity between each group of features. Specifically, we adopt popular kernel functions, i.e., n-gram kernel, dependency kernel, edit distance similarity and linear kernel.

We now discuss *how the sub-kernels are computed* for the feature groups, i.e., n-gram, contextual path and dependency path.

First, we give the definition of a similarity matrix in the input X. S is a $n \times n$ similarity matrix with two entries for every pair of vectors in X, $S(i_k, j_k) = s_{ij}$ for i, $j \in \{1, 2 \ldots, n\}$ the indices of instances in X.

$$S = \begin{pmatrix} s_{11} & s_{12} & \cdots & \cdots & s_{1n} \\ \vdots & \vdots & \ddots & s_{ij} & \vdots \\ s_{n1} & s_{n2} & \cdots & \cdots & s_{nn} \end{pmatrix}$$

Note that the large values for the entries indicate a high degree of similarity between the corresponding data points and small values indicate a low degree of similarity between the corresponding data points.

Then, we define the global similarity matrix based inner product $\langle .|. \rangle_{MAT} : X^l \times X^l \mapsto \mathbb{R}$ as,

$$\langle x_i | x_j \rangle_{MAT} = \sum S(i_k, j_k) \tag{3}$$

where,

- $(x_i, x_j)_{MAT}$ is the kernel matrix
- i and j are instances in the input X
- k is a group of features,
  $k = \{n-gram, dependencypath, contextualpath, string, binary\}$.

For example, for the calculation of the similarity measure between n-grams characters, we use the k-spectrum (n-gram) kernel function. Given a string x, an alphabet $A(|A| = l)$, we define a feature map from X to R by,

$$\phi_k(x) = (\phi_a(x))_{a \in A^k} \tag{4}$$

where $\phi_a(x)$ is the number of occurrences of a in x. Then, the k-spectrum kernel function is defined as,

$$K_k(x,y) = \langle \phi_k(x), \phi_k(y) \rangle \tag{5}$$

For the dependency path features, we use the dependency kernel of Kim et al. [Lodhi et al., 2002] walk kernel, which is tested with a SVM classifier on the LLL 05 challenge task to extract genic interactions and achieved a promising result. We define our dependency graph kernel to capture the isomorphism between two graph structures. For this purpose, we sum up the number of common walks features between two dependency graphs $G(V,E)$ and $G'(V',E')$. Note that the graph means the directed dependency chain paths at depth of n $(n = 1,2,3)$.

In our work, we consider the walk of length 1 called a v-walk. In addition, we present an e-walk that begins and ends with an edge e. We generate lexical walk features, which consist of lexical words L_w; and syntactic walk features S_w, which consist of POS and dependency relations. The set of lexical and syntactic walk features is noted by F_w of an edge e. Hence, our dependency graph kernel is expressed by,

$$K(G,G') = \sum_{e \in E} \sum_{e' \in E'} K_{walk}(e,e') \tag{6}$$

where,

$$K_{walk}(e,e') = \begin{cases} 1 & \text{if} \quad f_w = f'_w \\ 0 & \text{Otherwise.} \end{cases}$$

We calculate the edit distance kernel [Erkan et al., 2007] for the semantic label paths. Suppose p_i and p_j are the semantic paths extracted from a pattern in sentence s_i and sentence s_j. The edit distance between p_i and p_j is the minimum of number of operations (deletion, insertion, substitution at dependency path level) that have to transform the first path to the second.

The distance measure is converted into a similarity measure as follows:

$$edit\_sim(p_i,p_j) = e^{(} - \lambda\, edit\_dist(p_i,p_j) \tag{7}$$

where $\lambda > 0$.

3.3 Argument Detection

First, we generate features for all shortest dependency paths between predicted trigger and named entity. Then, we define a kernel based similarity computation which

is able to capture the argument detection task specific similarity between shortest path features. Each shortest path example is classified as belonging to one of the argument type classes (*Theme* or *Cause*) or as negative.

3.3.1 Argument-Based Features Extraction

Many of the features used are inspired by those used in [Vlachos et al., 2009]. Given the shortest dependency path between event trigger and named entity, we extract rich features to represent candidate arguments. The feature set is showed in Table 4.

Table 4 Features for argument detection

Type	Feature
Frequency features	Length of the shortest path between two entities Number of named entities and event triggers per type in the sentence
N-grams features	N-grams of dependencies (n = 2, 3, 4) N-grams of consecutive words
Element features	Trigger / Argument word Trigger / Argument stem Trigger / Argument type Trigger / Argument POS Confidences of trigger tokens obtained by trigger detection
Dependency path features	Directions of dependency edges relative to the shortest path Types of dependency edges relative to the shortest path
Semantic features	Annotation label of the shortest path Combination of the specific type of the terminus token of the shortest path and their categories

3.3.2 Training and Testing Argument Detection

Like the trigger detection, we define a kernel function that present the shortest path features. The kernel matrix of the argument detection is presented as the following:

$$\langle x_i | x_j \rangle_{MAT} = \sum_k S(i_k, j_k) \tag{8}$$

where $k = \{\text{dependency n-gram, dependency path, semantic path, frequency, string}\}$.

- Dependency n-gram: we use the k-spectrum kernel between the n-grams of the two shortest paths.

- Dependency path: we use the dependency kernel described above in the trigger detection to calculate the similarity between the dependency edges relative to the two shortest paths.
- Frequency: we calculate the similarity between two frequency feature vectors using the cosine similarity measure.
- Semantic path: we use the Edit distance kernel [Lodhi et al., 2002] which calculates the similarity between the annotation labels of the two shortest paths.

3.4 Argument Grouping

The target output of the argument detection is in the form of a primary frame consisting of an event class, semantic roles and participants (protein or event). For argument grouping, we need to find the best combinations of event frames that are detected by the argument detector to represent complex events (i.e., binding and regulation).

We construct classification models for the complex event detection. First, we create negative and positive examples from the combinations of detected arguments (complex event candidate). We design features of a complex event candidate for complex event detection that constrain the event argument types and combinations defined in the event ontology.

The features contain three relations, relations between arguments, relations between triggers and outer proteins, and relations between arguments and outer triggers. Hence, we apply the feature-based argument extractor as mentioned in the above section for three types of substructures: each shortest path in the complex event, all pairs among arguments, all shortest paths including event trigger outside of events, all pairs between argument proteins and their closest proteins in binding event. The first relations are used to remove candidates that contain non-related arguments, and the second and third relations are used to remove candidates by finding the shortest paths that should be included in the candidates and more appropriate combinations of event arguments.

4 Implementation and Experimental Results

we present the implementation of our system BioEv to solve the event extraction task with the BioNLP'09 Shared Task resources provided by [Kim et al., 2009].

We summarize the different corpora and tools used in this implementation and we present our experimental results.

4.1 Experimental Data

The experimental data set are prepared based on the GENIA corpus in the context of the BioNLP'09 Shared Task. They consist of PubMed documents (title and abstract only). The Table 5 shows the GENIA corpus statistics.

Table 5 Experimental data sets [Kim et al., 2009]

	Train	Development	Test
Abstract	800	150	260
Sentence	7499	1450	2447
Token	176146	33937	57367
Event	8597	1809	3182

4.2 Tools

The event extraction pipeline consists of four major parts, a pre-processor, a trigger detector, an argument detector and a complex event detector. Data pre-processing is performed using a pipeline of NLP tools. First, we use the GENIA sentence splitter and the GENIA tagger provided by U-Compare [Elkhlifi and Faiz, 2010] for splitting and tokenizing the documents of the training and test data sets. Then, we use the McClosky-Charniak domain-adapted parser [McClosky and Charniak, 2008b]. The output of the parser is provided in the standard Penn Treebank (PTB) format. The output of the parser is converted with the Stanford tools to the collapsed form of the Stanford dependency scheme.

Finally, we annotate the semantic class for each term using WordNet and the UMLS Metathesaurus. Gene, protein, RNA, cell line and cell type names are identified by ABNER.

After document of the training set has passed through a pre-processor, we filter out the tokens in the texts by part-of-speech. For trigger-based features extraction, we implement methods to extract feature vectors, which are restricted to dependency values and semantic values. We use a number of tools to extract features. We create the shortest path using the NetworkX, which is in Python. For stemmatization and lemmatization, we use the Porter Stemmer [Brill, 1995] and WordNet natural language toolkit[3], respectively. We implement JAVA methods to extract features for event detection sub tasks, i.e., trigger detection, argument detection and argument grouping. For SVM classification, we use the LIBSVM software[4].

4.3 Results

We present three state of the art systems. These are the UTurku system scored on first rank in the BioNLP'09 Shared Task of Björne et al. [Vlachos et al., 2009] and the CCP-BTMG system achieved the highest precision of [Buyko et al., 2009].

The overall performance scores of our system are 50.57% recall, 64.88% precision and 56.83% f-score in Table 6.

[3] `http://www.nltk.org/`

[4] `http://www.csie.ntu.edu.tw/cjlin/libsvm/`

Table 6 Experimental results for event extraction (Recall / Precision / F-score)

	Precision	Recall	F-score
UTURKU system	46.73	58.48	51.95
CCP-BTMG system	13.45	**71.81**	22.66
Our system	**50.57**	64.88	**56.83**

5 Conclusion

In the previous decade of work on automatic information extraction from biological texts, efforts have focused in particular on the basic task of recognizing entity names and on the extraction of simple relations of these entities and, more recently, on the biological event extraction.

In our work, we propose an event extraction approach using support vector machines and composite kernel function. We start processing texts by analyzing natural language documents using lexical and syntactic resources to obtain sentences, tokens and POS tags. Then, tokens are organized into groups after a syntactic and semantic analysis has assigned meaning to these tokens or groups of tokens. In the trigger and argument detection phase, we extract feature vectors for training and testing using a SVM modeling. We combine multiple layers of syntactic and semantic information by applying distinct kernels on features. The combination of distinct kernels is achieved through summing the values of each kernel for each type of feature.

In order to evaluate our approach, we implement our event extraction system. We obtain a recall around 50.57%, a precision around 64.88% and an f1-score around 56.83%, for a set of GENIA abstracts.

Our first future work consists of evaluating the performance of our approach on the GENIA full text articles, different corpora such as BioInfer corpus and comparing our composite kernels to benchmarks of various kernels.

Another line of research will be to exploit the event extraction output in text mining tasks such as event network analysis, hypothesis generation, pathway extraction and others.

References

[Ananiadou et al., 2010] Ananiadou, S., Pyysalo, S., Tsujii, J., Kell, D.B.: Event extraction for systems biology by text mining the literature. Trends in Biotechnology 28, 381–390 (2010)

[Björne et al., 2009] Björne, J., Heimonen, J., Ginter, F., Airola, A., Pahikkala, T., Salakoski, T.: Extracting complex biological events with rich graph-based feature sets. In: Proceedings of the Workshop on BioNLP: Shared Task, pp. 10–18 (2009)

[Brill, 1995] Brill, E.: Transformation-based error-driven learning and Natural Language Processing: A case study in part-of-speech tagging. Computational Linguistics 21, 543–565 (1995)

[Buyko et al., 2009] Buyko, E., Faessler, E., Wermter, J., Hahn, U.: Event extraction from trimmed dependency graphs. In: Proceedings of the Workshop on BioNLP: Shared Task, pp. 19–27 (2009)

[Cohen et al., 2009] Cohen, K.B., Verspoor, K., Johnson, H.L., Roeder, C., Ogren, P.V., Baumgartner Jr., W.A., White, E., Tipney, H., Hunter, L.: High-precision biological event extraction with a concept recognizer. In: Proceedings of the Workshop on BioNLP: Shared Task, pp. 50–58 (2009)

[Elkhlifi and Faiz, 2010] Elkhlifi, A., Faiz, R.: French-written event extraction based on contextual exploration. In: Proceedings of the 23th International FLAIRS 2010, pp. 180–185. AAAI Press (2010)

[Erkan et al., 2007] Erkan, G., Ozgur, A., Radev, D.R.: Semi-supervised classification for extracting protein interaction sentences using dependency parsing. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pp. 228–237 (2007)

[Faiz, 2006] Faiz, R.: Identifying relevant sentences in news articles for event information extraction. International Journal of Computer Processing of Oriental Languages, 1–19 (2006)

[Friedman et al., 2001] Friedman, C., Kra, P., Yu, H., Krauthammer, M., Rzhetsky, A.: GENIES: a natural-language processing system for the extraction of molecular pathways from journal articles. Bioinformatics 17, 74–82 (2001)

[Fundel et al., 2006] Fundel, K., Kuffner, R., Zimmer, R.: RelEx: relation extraction using dependency parse trees. Bioinformatics, 365–371 (2006)

[Hakenberg et al., 2009] Hakenberg, J., Solt, I., Tikk, D., Tari, L., Rheinlander, A., Ngyuen, Q.L., Gonzalez, G., Leser, U.: Molecular event extraction from Link Grammar parse trees. In: Proceedings of the Workshop on BioNLP: Shared Task, pp. 86–94 (2009)

[Humphreys et al., 2000] Humphreys, K., Demetriou, G., Gaizauskas, R.: Two applications of information extraction to biological science journal articles: Enzyme interactions and protein structures. In: Proceedings of the Pacific Symposium on Biocomputing, pp. 505–516 (2000)

[Kim et al., 2009] Kim, J.D., Ohta, T., Pyysalo, S., Kano, Y., Tsujii, J.: Overview of BioNLP09 Shared Task on event extraction. In: Proceedings of the Workshop on BioNLP: Shared Task, pp. 1–9 (2009)

[Leroy and Chen, 2002] Leroy, G., Chen, H.: Filling preposition-based templates to capture information from medical abstracts. In: Proc. Pacific Symp. on Biocomputing 7 (PSB), pp. 362–373 (2002)

[Lodhi et al., 2002] Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.: Text classification using string kernels. Journal of Machine Learning Research 2, 419–444 (2002)

[Marneffe and Manning, 2008] Marneffe, M.C.D., Manning, C.: Stanford typed hierarchies representation. In: Proceedings of the COLING 2008 Workshop on Cross-Framework and Cross-Domain Parser Evaluation, pp. 1–8 (2008)

[McClosky and Charniak, 2008a] McClosky, D., Charniak, E.: Selftraining for biomedical parsing. In: Proceedings of ACL 2008: HLT, pp. 101–104 (2008a)

[McClosky and Charniak, 2008b] McClosky, D., Charniak, E.: Selftraining for biomedical parsing. In: Proceedings of ACL 2008: HLT, pp. 101–104 (2008b)

[McDonald et al., 1995] McDonald, D.M., Chen, H., Su, H., Marshall, B.B.: Extracting gene pathway relations using a hybrid grammar: the Arizona relation parser. Bioinformatics 9, 3370–3378 (1995)

[Mitchell et al., 2003] Mitchell, J., Aronson, A., Mork, J.: Gene indexing: Characterization and analysis of nlm's generifs. In: Proceedings of the AMIA Symposium, pp. 460–464 (2003)

[Mukherjea et al., 2004] Mukherjea, S., Subramaniam, L.V., Chanda, G., Kothari, R., Batra, V., Bhardwaj, D., Srivastava, B.: Enhancing a biomedical information extraction system with dictionary mining and context disambiguation. IBM Journal of Research and Development 48, 693–701 (2004)

[Neves et al., 2009] Neves, M.L., Carazo, J.M., Montano, A.P.: Extraction of biomedical events using case-based reasoning. In: Proceedings of the Workshop on BioNLP: Shared Task, pp. 68–76 (2009)

[Pyysalo et al., 2004] Pyysalo, S., Ginter, F., Pahikkala, T., Boberg, J., Jarvinen, J., Salakoski, T., Koivula, J.: Analysis of Link Grammar on biomedical dependency corpus targeted at protein-protein interactions. In: NLPBA/BioNLP at COLING, pp. 15–21 (2004)

[Rune et al., 2007] Rune, S., Yoshida, K., Yakushiji, A., Miyao, Y., Matsubayashi, Y., Ohta, T.: AKANE System: Protein-protein interaction pairs in Biocreative2 challenge. In: Proceedings of the Second BioCreative Challenge Evaluation Workshop, pp. 209–212 (2007)

[Sleator and Temperley, 1993] Sleator, D., Temperley, D.: Parsing english with a link grammar. In: 3rd Int. Workshop on Parsing Technologies, pp. 277–291 (1993)

[Talukdar et al., 2006] Talukdar, P.P., Brants, T., Pereira, M.: A context pattern induction method for named entity extraction. In: Proceedings of the 10th Conference on Computational Natural Language Learning, pp. 141–148 (2006)

[Vlachos et al., 2009] Vlachos, A., Buttery, P., Seaghdha, D.O., Briscoe, T.: Biomedical event extraction without training data. In: Proceedings of the Workshop on BioNLP: Shared Task, pp. 37–40 (2009)

[Yakushiji et al., 2001] Yakushiji, A., Tateisi, Y., Miyao, Y.: Event extraction from biomedical papers using a full parser. In: Pacific Symposium on Biocomputing, vol. 6, pp. 408–419 (2001)

Supervised Pre-processing of Numerical Variables for Multi-Relational Data Mining

Dhafer Lahbib, Marc Boullé, and Dominique Laurent

Abstract. In Multi-Relational Data Mining (MRDM), data are represented in a relational form where the individuals of the target table are potentially related to several records in secondary tables in one-to-many relationship. Variable pre-processing (including discretization and feature selection) within this multiple table setting differs from the attribute-value case. Besides the target variable information, one should take into account the relational structure of the database. In this paper, we focus on numerical variables located in a non target table. We propose a criterion that evaluates a given discretization of such variables. The idea is to summarize for each individual the information contained in the secondary variable by a feature tuple (one feature per interval of the considered discretization). Each feature represents the number of values of the secondary variable ranging in the corresponding interval. These count features are jointly partitioned by means of data grid models in order to obtain the best separation of the class values. We describe a simple optimization algorithm to find the best equal frequency discretization with respect to the proposed criterion. Experiments on a real and artificial data sets reveal that the discretization approach helps one to discover relevant secondary variables.

1 Introduction

Most of existing data mining algorithms are based on an attribute-value representation. In this flat format, each record represents an individual and the columns represent variables describing these individuals. In real life applications, data usually

Dhafer Lahbib · Marc Boullé
France Telecom R&D, 2 Avenue Pierre Marzin, 23300 Lannion, France
e-mail: {dhafer.lahbib,marc.boulle}@orange.com

Dhafer Lahbib · Dominique Laurent
ETIS-CNRS-Universite de Cergy Pontoise-ENSEA, 95000 Cergy Pontoise, France
e-mail: dominique.laurent@u-cergy.fr

F. Guillet et al. (eds.), *Advances in Knowledge Discovery and Management,*
Studies in Computational Intelligence 527,
DOI: 10.1007/978-3-319-02999-3_6,

present an intrinsic structure which is hard to express in a tabular format. This structure may be naturally described using the relational formalism where each object (target table record) refers to one or more records in other tables (secondary tables) through a foreign key.

Example 1. In the context of the Customer Relationship Management (CRM) problem, Figure 1 shows an extract of a virtual CRM relational database schema. In this schema, the table *Customer* is the target table, whereas *Order* and *Service* are secondary tables related to *Customer* through the foreign key *CID*. In this context, the problem may be, for instance, to identify the customers likely to be interested in a certain product or service. This problem turns into a classification problem for which the target variable is the *Status* attribute, which denotes whether the customer has already ordered a particular product.

Learning from relational data has recently received increasing attention in the literature. The term Multi-Relational Data Mining (MRDM) was initially introduced by [Knobbe et al., 1999] to address novel knowledge discovery techniques from multiple relational tables. The common point between these techniques is that they need to transform the relational representation. In Inductive Logic Programming ILP [Džeroski, 1996], data is recoded as logic formulas. This causes scalability problems especially with large-scale data. Other methods called by Propositionalisation [Kramer et al., 2001] try to flatten the relational data by creating new variables. These variables aggregate the information contained in non target tables in order to obtain a classical attribute-value format. Consequently, not only the naturally compact initial representation is lost but there is a risk of introducing statistical bias because of potential dependencies between the newly added variables.

Although variable pre-processing is at the core of the majority of propositional (single table) Data Mining systems, it has received much less attention in MRDM. Pre-processing, including variable selection and discretization of numerical values, is of great importance particularly in Multi-Relational context. This step is justified not only to improve the accuracy but also to reduce the very large hypothesis spaces in MRDM. The difficulty when dealing with multiple table data arises from the presence of one-to-many associations. In the attribute-value mono table case, each individual has a single value per variable. While in multiple table setting, for a non target table variable, an individual may have a value list (eventually empty) of varying size.

Example 2. Referring back to Example 1, predicting whether the customer would be interested in a given product does not only depend on the information of that customer. Indeed, the other products ordered by this customer might be relavant, because, for instance, variables such as the product *Weight* or *Price* may present correlations with the target variable. Assessing the relevance of these variables is not straightforward, since each customer may have made many orders. The same difficulty arises when trying to discretize accurately these numerical variables, especially when taking the class label into account.

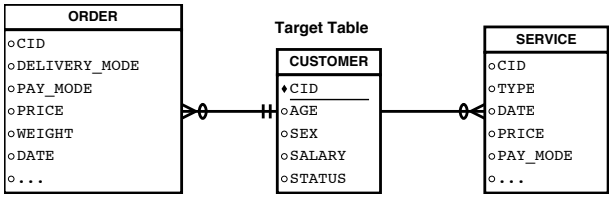


Fig. 1 Relational schema of a CRM database

To the best of our knowledge, only few studies in the literature have treated the numeric variable discretization in multi-relational data. Discretizing numerical attributes in multiple tables is different from handling attributes from a single table, due the presence of one-to-many associations. Under the multi-relational setting, the state of the art discretization approaches of a secondary numerical attribute differ along 2 axes: (i) whether they make use of the class label and (ii) whether they consider one-to-many relationships when computing cut points. The simplest methods that can be applied are the *equal-width* and *equal-frequency* interval binning. Both, are unsupervised and they compute boundaries regardless of any multi-relational structure. Whereas, the former divides the range of observed values into k equal sized bins, the latter discretizes the variable in such a way that each bin will have approximately the same number of values. To take into account the one-to-many association problem, [Knobbe and Ho, 2005] proposed an *Equal-weight* discretization method which involves an idea proposed by [an LaerVAN LAER et al., 1997]: individuals with large number of related records in the non target table have a bigger influence on the choice of boundaries since they have more contributing numeric values. In order to compensate this impact, numeric values are weighted with the inverse of the size of the bags of records they belong to. Instead of producing ranges of equal size like in the equal frequency method, cut points are computed so that bins of equal weight can be obtained. All the above methods are class-blind since they do not use class labels. In order to take into account both the target variable information and the one-to-many association between records stored in the target and non-target tables, [Alfred, 2009] proposes a modification of the entropy-based multi-interval discretization method introduced by Fayyad and Irani [Fayyad and Irani, 1993]. Besides the class information entropy, another measure that uses individual information entropy is added to select multi-interval boundaries for the numerical secondary variable. The drawback of this approach is that it is relatively expensive and may lead to statistical skews since the entropy measures are computed by propagating the class labels to the non target tables. When performing such transformations, variables in the secondary table are not independent and identically distributed (i.i.d.). In fact, individuals with a large number of related records in a secondary table will be overestimated thereby causing overfitting.

In this paper, we are interested in pre-processing a variable located in a secondary table having a one-to-many relation with the target one[1]. We propose to discretize the set of related values of a variable A and use an optimization criterion to find the best partitioning of the set such that the class Y is maximally differentiated. The idea is to use multi-variate data grids to estimate the conditional probability $P(Y \mid A)$. This univariate pre-processing extended to the relational context is of a great interest for filter feature selection [Guyon and Elisseeff, 2003] or as pre-processing step for classifiers such as Naive Bayes or Decision Tree.

The remainder of this paper is organized as follows. Section 2 describes our approach in the case of a secondary numerical variable. In Section 3 we evaluate the approach on artificial and real data sets. Finally, Section 4 gives a summary and discusses future work.

2 Secondary Variables Pre-processing

In this section, we describe how a numerical variable belonging to a non-target table can be discretized in a class-dependent way.

2.1 Illustration of the Approach

Let us take the simplest case: a binary variable with two values v_1 and v_2. In this case, each individual is described by a bag of values among v_1 and v_2[2]. Given an individual, all that we need to know about the secondary variable are the number of v_1 and the number of v_2 in the bag of records related to that individual (we denote them respectively n_1 and n_2). Thus, the whole information about the initial variable can be captured by considering jointly the pair (n_1, n_2). With such a representation, the conditional probability $P(Y \mid A)$ is then equivalent to $P(Y \mid n_1, n_2)$.

This approach can be generalized to a numerical secondary variable. In that case, the variable needs to be discretized into K intervals. The idea is to create in the target table K new variables n_k ($1 \leq k \leq K$). For each individual, n_k stands for the number of related records in the secondary table which have a value of A located in the k^{th} interval. As in the bivariate case, $P(Y \mid A)$ is approximated by evaluating $P(Y \mid (n_1, n_2, \cdots, n_K))$.

Multivariate data grid models have been shown to be good estimators for the probability of a class, given a set of input variables [Boullé, 2011]. The idea is to jointly discretize in an optimal way the numeric variables n_k into intervals. This joint partitioning defines a distribution of the instances in a K-dimensional input data grid whose cells are defined by interval tuples. Therefore, our goal is to find the optimal

[1] The one-to-one relationship is equivalent to the single table case. For simplification reasons, we limit the relationship to the first level: tables directly related to the target one.

[2] This is different from the attribute-value setting, where for a given variable, an individual can only have a single value.

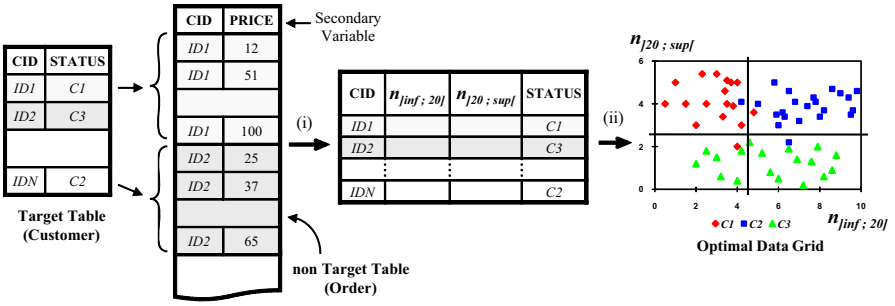


Fig. 2 Illustration of the Approach

multivariate discretization which maximizes the class distribution. In other words, we look for the optimal grid with homogeneous cells according to the class values.

Example 3. In the context of Example 1, consider, for instance, the secondary variable "PRICE" in the database of figure 1. Assume that we discretize this variable into two intervals: $]inf;20]$ and $]20;sup[$. Then PRICE is equivalent to the pair of variables $\left(n_{]inf;20]}, n_{]20;sup[}\right)$ where $n_{]inf;20]}$ (respectively $n_{]20;sup[}$) stands for the number of orders whose prices are less than 20 (respectively greater than 20). If we assume that the price is correlated with the target variable and that the discretization in two intervals is relevant, the target classes can be separated easily, using a grid similar to that of Figure 2.

The correlation between the cells of the data grid and the target values allows to quantify the joint classificatory information. The conditional probability distribution $P(Y \mid A)$ is evaluated locally in each cell. Consequently, classifiers like Naive Bayes or Decision Trees can easily be used. Moreover, it is important to note that the data grid provides an interpretable representation, since it shows the distribution of the individuals while jointly varying the count variables n_k. Each cell can be interpreted as a classification rule in the multi-relational context.

For example, the top-left cell of the data grid of Figure 2 is interpreted by: **if** "the number of orders with a price less than 20 is less than 5" **and** "the number of orders with a price greater than 20 is more than 2" **then** the class is $C1$.

Given that we use an equivalent representation, with the suitable discretization, we expect that the optimal related data grid will be able to detect the pattern contained in the secondary variable. Thus the problem is twofold: how to find the best discretization and how to optimize the related data grid. We address these two problems simultaneously by applying a model selection approach. To do so, we follow the MODL (Minimum Optimized Description Length) approach [Boullé, 2006]. The best model is chosen according to a Maximum A Posteriori (MAP) approach by maximizing the probability $p(\text{Model}|\text{Data})$ of the model given the data. By applying the Bayes rule, this is equivalent to maximizing $P(\text{Model})p(\text{Data}|\text{Model})$ since the probability $P(\text{Data})$ is constant under varying the model. The considered models include the discretization of the secondary variable A and the joint partitioning of

the generated count variables n_k. In the remainder of this section, we describe the criterion used to evaluate these models and we propose optimization algorithms.

2.2 Evaluation Criterion

A model is completely defined by the discretization of the secondary variable (number and bounds of the intervals), the partitioning of the count variables n_k and the target distribution in each cell of the resulting data grid. To describe such a model, we use the following notation.

Notation 1. • *N: number of individuals (number of target table records)*
- *J: number of target values*
- N_s*: number of records in the non target table*
- *K: number of discretization intervals for the secondary variable A*
- n_k*: number of non target table records having a value of the secondary variable A in the* k^{th} *interval* $(1 \leq k \leq K)$
- I_k*: number of discretization intervals for the count variable* n_k $(1 \leq k \leq K)$
- N_{i_k}*: number of individuals in the interval* i_k *for variable* n_k $(1 \leq k \leq K)$
- $N_{i_1 i_2 \ldots i_K}$*: number of individuals in the cell* $(i_1, i_2, \ldots, i_K)$
- $N_{i_1 i_2 \ldots i_K j}$*: number of individuals in the cell* $(i_1, i_2, \ldots, i_K)$ *for the target value* j

Using the notation above, a model is completely defined by the parameters $\{K, \{n_k\}, \{I_k\}, \{N_{i_k}\}, \{N_{i_1 i_2 \ldots i_K j}\}\}$. In order to compute the criterion, we introduce in Definition 1 a prior distribution $p(\text{Model})$ on this model space. This prior makes explicitly the independence assumptions and exploits the hierarchy of the parameters. The number of discretization intervals of the secondary variable A is first chosen, then their bounds. After computing the count variables n_k, a K-dimensional data grid is built by choosing for each n_k the number of intervals, their bounds and finally the frequencies of the target values in each cell. At each stage of this hierarchy the choice is assumed to be uniform.

Definition 1. *The hierarchical prior of the parameters of discretization models is defined as follows:*

- *the numbers of intervals for the secondary variable discretization are independent from each other, and uniformly distributed between* 1 *and* N_s*,*
- *for a given number of intervals, every discretization of the secondary variable into intervals is equiprobable,*
- *for the discretization of the count variable* n_k*, the numbers of intervals are independent from each other, and uniformly distributed between* 1 *and N,*
- *for each count variable* n_k *and for a given number of intervals, every partition into intervals is equiprobable,*
- *for each cell of the data grid, all the parameters of the multinomial distribution of the target classes are equiprobable,*
- *the parameters of the multinomial distributions of the target classes in each cell are independent from each other.*

The first hypothesis of the above prior is that, for the secondary variable being discretized, the number of intervals is uniformly distributed between 1 and N_s. Thus we get

$$p(K) = \frac{1}{N_s} \tag{1}$$

The second hypothesis is that all discretizations of the secondary variable into K intervals are equiprobable for a given K. If N_s is the number of the secondary table records, there is $\binom{N_s + K - 1}{K - 1}$ ways to discretize N_s values into K intervals. Thus we obtain

$$p(\{n_k\} \mid K) = \frac{1}{\binom{N_s + K - 1}{K - 1}} \tag{2}$$

For each count variable n_k, the number of discretization intervals is uniformly distributed between 1 and N. Thus, we get

$$p(I_k \mid n_k, K) = \frac{1}{N} \tag{3}$$

For each count variable n_k, all the divisions of N instances into I_k intervals are equiprobable.

$$p\left(\left\{N_{i_k}\right\} \mid I_k, n_k, K\right) = \frac{1}{\binom{N + I_k - 1}{I_k - 1}} \tag{4}$$

Given K univariate discretizations of the count variables n_k, the frequency $N_{i_1 i_2 \ldots i_K j}$ of each cell $(i_1, i_2, \ldots, i_K)$ of the data grid can be derived from the input data sample. According to the fifth hypothesis of the prior distribution, in each cell $(i_1, i_2, \ldots, i_K)$, all the parameters of the multinomial distributions of the $N_{i_1 i_2 \ldots i_K}$. instances of the cell on the J target classes are equiprobable. Calculating the probability of a such set of multinomial parameters is a combinatorial problem, which turns into computing the number of ways of decomposing a natural number $N_{i_1 i_2 \ldots i_K}$ as a sum of J terms. Since each set of multinomial parameters is equiprobable, we obtain

$$p\left(\left\{N_{i_1 i_2 \ldots i_K j}\right\} \mid \left\{N_{i_k}\right\}, \{I_k\}, n_k, K\right) = \frac{1}{\binom{N_{i_1 i_2 \ldots i_K} + J - 1}{J - 1}} \tag{5}$$

For the likelihood term $p(\text{Data}|\text{Model})$, we assume further that the multinomial distributions of the target values in each cell are independent from each other. This term is evaluated locally in each cell by considering the probability of observing the target values (classes) of the cell given the parameters of the multinomial distribution in this cell. The number of ways of observing $N_{i_1 i_2 \ldots i_K}$ instances distributed according to a multinomial distribution is given by the multinomial coefficient:

$$\frac{N_{i_1 i_2 \ldots i_K}!}{\prod_{j=1}^{J} N_{i_1 i_2 \ldots i_K j}!}.$$

The conditional likelihood per cell is thus

$$\frac{1}{\frac{N_{i_1 i_2 \ldots i_K}!}{\prod_{j=1}^{J} N_{i_1 i_2 \ldots i_K j}!}} \tag{6}$$

Taking the negative log of $P(\text{Model})p(\text{Data}|\text{Model})$, the generalized optimization criterion is given below.

$$\begin{aligned}
&\log N_s + \log \binom{N_s + K - 1}{K - 1} \\
&+ \sum_{k=1}^{K} \log N + \sum_{k=1}^{K} \log \binom{N + I_k - 1}{I_k - 1} \\
&+ \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_K=1}^{I_K} \log \binom{N_{i_1 i_2 \ldots i_K} + J - 1}{J - 1} \\
&+ \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_K=1}^{I_K} \left(\log N_{i_1 i_2 \ldots i_K}! - \sum_{j=1}^{J} \log N_{i_1 i_2 \ldots i_K j}! \right)
\end{aligned} \tag{7}$$

In formula 7, the first line stands for the choice of the discretization of the secondary variable: The first and second terms represent respectively the choices of the number of intervals, and the bounds of the intervals. The second and the third lines stand for the choice of the discretization of each variable n_k and the multinomial distribution parameters for the target values in each grid cell. The last term represents the conditional likelihood of the data given the model.

The criterion given in the above formula is related to the probability that the final data grid (obtained after the discretization of the secondary variable, as described before) explains the target variable given the secondary one. It can also be interpreted as the ability of a data grid to encode the target classes given the secondary variable, since negative log of probabilities is none other than a description length [Shannon, 1948].

Based on this cost, we can define a normalized compression gain $g(M)$ by considering the null model, denoted by $M_\emptyset$, where the secondary variable is discretized into one interval.

$$g(M) = 1 - \frac{cost(M)}{cost(M_\emptyset)} \tag{8}$$

This relevance level can be used as a filter criterion for ranking secondary variables [Guyon and Elisseeff, 2003].

Algorithm 2: Optimization Algorithm

input : $\mathscr{K}$: initial number of quantiles,
: K_{max}: max number of evaluated ranges
output : D^*: best secondary variable discretization,
: G^* best Data Grid
require: $K_{max} \ll \mathscr{K}$

Compute secondary variable quantiles bounds ($\mathscr{K}$-way equal frequency discretization) ;
Compute initial count variables $(v_k)_{1 \leq k \leq \mathscr{K}}$;
`/* Init solution (c*: best cost) */`
$c* \leftarrow \infty$, $D^* \leftarrow$ One interval, $G^* \leftarrow$ One cell;
for $K \leftarrow 2$ **to** K_{max} **do**
  $D \leftarrow$ discretize into K intervals;
  Estimate count variables $(n_k)_{1 \leq k \leq K}$ $n_k = \sum_{i=1+\left[\mathscr{K}\frac{k-1}{K}\right]}^{\left[\mathscr{K}\frac{k}{K}\right]} v_i$;
  Initialize G_K (data grid with n_k as input variables);
  `/* Optimize the data grid G_K */`
  $G_K^{'} \leftarrow OptimizeDataGrid(G_K)$;
  if $cost(G_K^{'}) < c^*$ **then** `// if improved cost`
    `/* save improved solution */`
    $c^* \leftarrow cost(G_K^{'})$, $G^* \leftarrow G_K^{'}$, $D^* \leftarrow D$;
  end if
end for

2.3 Optimization Algorithm

The choice of the secondary variable discretization is determined by the minimization of the criterion seen in Section 2.2, which is a combinatorial problem with 2^{N_s} possible discretizations for the secondary variable. Then, for each discretization into K intervals, there are $\left(2^N\right)^K$ possible data grids, which represent the number of the multivariate partitioning of the count variables $n_1, \ldots, n_K$. An exhaustive search through the whole space of models is unrealistic.

Algorithm 2 provides a simple procedure to optimize the discretization of the secondary variable. The method starts by making a fine $\mathscr{K}$-way equal frequency discretization of the secondary variable, which produces $\mathscr{K}$ initial count variables $v_1, \ldots, v_{\mathscr{K}}$. Then we iterate merging these initial ranges in order to simulate different equal frequency binnings. Each candidate discretization D_k is evaluated by optimizing the corresponding data grid G_K. This is done using the multivariate data grid optimization heuristics detailed in [Boullé, 2011], which have practical scaling properties, with $O(N)$ space complexity and $O(N\sqrt{N}\log N)$ time complexity. At the end of Algorithm 2, we select the secondary discretization with the minimum evaluation cost (see criterion 7).

Table 1 Description of the used data sets

	# tables	# Numerical Sec. var.	# non target records	# Individuals	# target values
Mutagenesis-atoms[3]	2	2	1618	188	2
Mutagenesis-bonds[3]	2	4	3995	188	2
Mutagenesis-chains[3]	2	6	5349	188	2
Diterpenses[4]	2	1	30060	1503	23
Miml[5]	2	15	18000	2000	2
Stulong[6]	2	29	10572	1417	2
Xor 2D	2	1	987762	10000	2
Xor 3D	2	1	1843282	10000	2

Although this simple algorithm clearly partially exploits the richness of the considered models, it is a good validation of the overall approach. As a priority for future work, we plan to extend this optimization procedure in order to better explore the search space and discover more complex discretization patterns.

3 Experiments

Our approach has been evaluated through its impact as a pre-processing step to a Naive Bayes (NB) classifier. In this multi-relational NB, for a given one-to-many numerical variable X_i, the optimal data grid gives an estimation of the corresponding univariate conditional density $P(X_i \mid Y)$, which is computed by considering the class frequencies in each cell. To show the contribution of our pre-processing approach over aggregation based methods, for each secondary attribute, the average value has been computed and a usual NB has been applied on the resulting flat table. Other aggregates were tested, namely Max, Min and the Number of records in secondary table. Results similar to those described below were obtained, and are omitted due to lack of space.

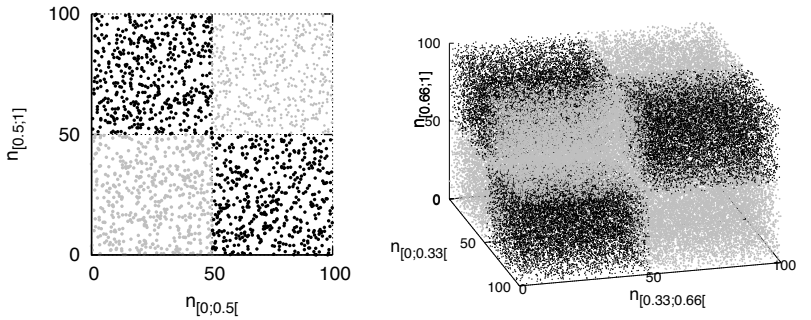
In our experiments, we have considered different classification tasks based on synthetic and real world data sets, whose characteristics are shown in Table 1.

[3] `http://sourceforge.net/projects/proper/files/datasets/0.1.0/`

[4] `http://cui.unige.ch/~woznica/rel_weka/`

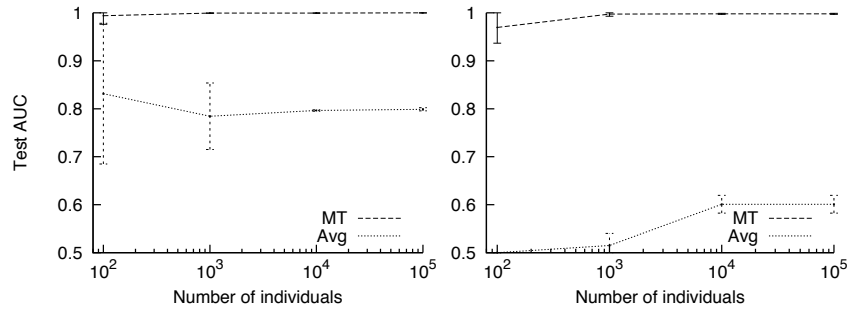
[5] `http://lamda.nju.edu.cn/data_MIMLimage.ashx`

[6] The study (STULONG) was realized at the 2nd Department of Medicine, 1st Faculty of Medicine of Charles University and Charles University Hospital, U nemocnice 2, Prague 2 (head. Prof. M. Aschermann, MD, SDr, FESC), under the supervision of Prof. F. Boudík, MD, ScD, with collaboration of M. Tomečková, MD, PhD and Ass. Prof. J. Bultas, MD, PhD. The data were transferred to the electronic form by the European Centre of Medical Informatics, Statistics and Epidemiology of Charles University and Academy of Sciences (head. Prof. RNDr. J. Zvárová, DrSc). The data resource is on the web pages `http://euromise.vse.cz/challenge2004`. At present time the data analysis is supported by the grant of the Ministry of Education CR Nr LN 00B 107.



(a) 2D Xor pattern (b) 3D Xor pattern

Fig. 3 Scatter plots of synthetic data sets. Colors (black and gray) refer to the class labels.



(a) 2D Xor pattern (b) 3D Xor pattern

Fig. 4 Experimental results obtained on synthetic data sets

Regarding synthetic data sets, the ideal binning pattern is known in advance, and the target label is generated according to an Xor function between the count variables n_i. Figure 3 depicts the scatter plots of the 2D and 3D Xor datasets. For instance, in the 3D Xor pattern (Figure 3b), the secondary variable is supposed to be discretized into three intervals: $[0;0.33[$, $[0.33;0.66[$ and $[0.66;1[$. In this rather complex pattern, data points located, for example, at the corner near the origin (in gray) refer to individuals which have less than 50 values in the non target table, respectively, in the intervals $[0;0.63[$, $[0.33;0.66[$ and $[0.66;1[$.

To compare results, we recorded the Area Under the ROC Curve (AUC) using ten-fold cross-validation. The AUC criterion (see [Fawcett, 2003]) evaluates the ranking of the class conditional probabilities. In a two-class problem, the AUC is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. In our experiments, we use the approach of [Provost and Domingos, 2001] to calculate the multi-class AUC,

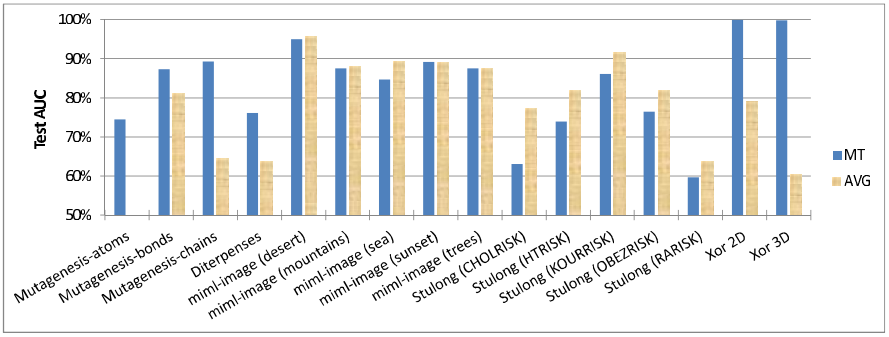


Fig. 5 Results of empirical data experiments obtained on artificial and real world data sets

by computing each one-against-the-others two-classes AUC and weighting them by the class prior probabilities $P(Y_j)$.

In all experiment, only secondary numerical variables have been considered in the data sets and we have chosen $\mathscr{K} = 100$ and $K_{max} = 10$ as parameter for the optimization algorithm (cf. Algorithm 2). Obviously, it is not enough that only 10 equal frequency discretization are evaluated among $o\left(2^{N_s}\right)$ candidate discretization of the secondary variable. The objective of these experimentations is mainly to evaluate the potential of the approach, and investigate whether working on more sophisticated optimization algorithms is worth it.

Figure 5 shows the generalization performance (test AUC) obtained with a NB using our discretization approach (denoted MT) compared to the same classifier based on aggregated variables (denoted Avg). A two-tailed Student test at the 5% confidence level is performed in order to evaluate the significant wins or losses of our method versus the AVG method.

On synthetic data sets (Xor 2D and Xor 3D) our method widely outperforms the NB approach using the average value. Not surprisingly, this is explained by the fact that aggregation implies loss of information. On the other hand, our approach is able to recognize the pattern in the secondary variable and thus to discretize it correctly. This is confirmed by Figure 4, which summarizes the classification results obtained by varying the number of individuals in the artificial data sets. It can be seen that, with enough individuals, our approach reaches the theoretical performance. On the other hand, other experiments on a totally random pattern show that our method is robust, in the sense that it can detect the absence of predictive information in the secondary variable (which is materialized by a single interval discretization and an AUC near 50%).

On real world data sets, neither of the two methods dominates the other. Indeed, Figure 5 shows that: (i) our approach might perform better than the aggregation approach (Mutagenesis (atoms, bonds, chains) and Diterpenses), (ii) the two approaches might perform equivalently (Miml), and (iii) on Stulong data set, the aggregation approach might perform better than ours. This can be explained by the fact that our criterion needs a large number of individuals to recognize existing patterns

(this has been shown in [Boullé, 2011], for a similar criterion in the case of a single table), whereas, as shown in Table 1, the used real world data sets are relatively small. Furthermore, we recall that Algorithm 2 is fairly simple and does not exploit the whole potential of the discretization criterion. Indeed, Algorithm 2 simulates an equal frequency discretization, meaning that many improvements can be brought to it. These results reported above are confirmed by the student's test in terms of significant wins, draws and loses of our method compared to the AVG method. The test showed 4 significant wins of our method on the Mutagenesis data set (atoms, bonds and chains) and Diterpenses, 4 draws on the Miml dataset (desert, mountains, sunset and trees) and six loses on Stulong (CHOLRISK, HTRISK, KOURRISK, OBEZRISK and RARISK) as well as Miml (sea).

We would like to emphasize that, although our approach does not always perform better than the average approach, this could be explained by insufficient exploration of the model space. Moreover, the approach is able to detect complex patterns (cf. Figure 3) that any aggregate approach can *not* discover. The obtained discretization yields rules that can be of interest to the user. On the other hand, it should be clear that aggregate methods can *not* produce such rules.

To see an example of how we can interpret the resulting discretization of a secondary variable, let us consider the Stulong data set (consisting of a target table Patient in a one-to-many relationship with a table Exam), along with the secondary numerical variable CHLSTMG that describes for each exam the cholesterol level (mg). It turns out that this variable is relevant to predict the value of the target variable CHOLRISK, which indicates whether the patient has high cholesterol risk according to the two target values: Normal and Risky. Applying Algorithm 2 in this case leads to a discretization of CHLSTMG into two intervals, namely $]inf, 228.5[$

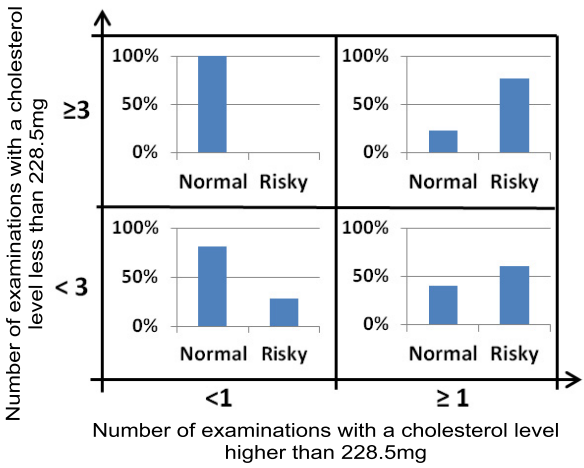


Fig. 6 Contingency table corresponding to the discretization of variable CHLSTMG

and $[228.5, sup[$, and Figure 6 depicts the optimal data grid corresponding to this binning (histograms show the distribution of the target values in each cell). This table can be interpreted as a set of four classification rules, one for each cell.

For example the top-left cell is equivalent to the rule: **If** there are at least 3 examinations with a cholesterol level less than 228.5 mg **and** there is no examination with a cholesterol level higher than 228.5 mg **then** the class is Normal (meaning no cholesterol risk).

4 Conclusion

In this paper, we have presented a novel approach to discretize numerical variables in a multi-relational setting. Specifically, we propose to project numerical data in secondary tables on the target one by means of binning, and then for each individual, to count records in each interval. Additionally, we have seen how candidate discretizations can be evaluated in a class-dependent way. A criterion has been proposed to evaluate to what extent a given discretization of a secondary numerical variable preserves the correlation with the target variable. Finally, an optimization algorithm has been provided for computing the optimal discretization. We have shown that the criterion is robust and is able to evaluate a given discretization in a reliable way.

An algorithm has been given for computing an estimation of equal-frequency interval binning. This procedure, however, does not take full advantage from the potential of the criterion. We are currently investigating how to extend our algorithm in order to better explore the search space, so as to discover more accurate discretization patterns.

This study has shown, through experiments on artificial data sets, that the criterion and the discretization procedure may help in discovering relevant secondary variables and achieving high accuracy. However, in the case of real world data sets, we need to look for larger data sets, in order to better assess our approach and to compare it to other multi-relational data mining techniques.

References

[Alfred, 2009] Alfred, R.: Discretization Numerical Data for Relational Data with One-to-Many Relations. Journal of Computer Science 5(7), 519–528 (2009)

[Boullé, 2006] Boullé, M.: MODL: A Bayes optimal discretization method for continuous attributes. Machine Learning 65(1), 131–165 (2006)

[Boullé, 2011] Boullé, M.: Data Grid Models for Preparation and Modeling in Supervised Learning. In: Guyon, I., Cawley, G., Dror, G., Saffari, A. (eds.) Hand on Pattern Recognition: Challenges in Machine Learning, pp. 99–130. Microtome Publishing (2011)

[Džeroski, 1996] Džeroski, S.: Inductive logic programming and knowledge discovery in databases. In: Advances in Knowledge Discovery and Data Mining, pp. 117–152. American Association for Artificial Intelligence, Menlo Park (1996)

[Fawcett, 2003] Fawcett, T.: ROC graphs: Notes and practical considerations for researchers. Technical report, Technical Report HPL-2003-4. Hewlett Packard Laboratories (2003)

[Fayyad and Irani, 1993] Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In: Thirteenth International Joint Conference on Artificial Intelligence, pp. 1022–1027 (1993)

[Guyon and Elisseeff, 2003] Guyon, I., Elisseeff, A.: An Introduction to Variable and Feature Selection. Journal of Machine Learning Research 3, 1157–1182 (2003)

[Knobbe et al., 1999] Knobbe, A.J., Blockeel, H., Siebes, A., Van Der Wallen, D.: Multi-Relational Data Mining. In: Proceedings of Benelearn 1999 (1999)

[Knobbe and Ho, 2005] Knobbe, A.J., Ho, E.K.Y.: Numbers in multi-relational data mining. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 544–551. Springer, Heidelberg (2005)

[Kramer et al., 2001] Kramer, S., Flach, P.A., Lavrač, N.: Propositionalization approaches to relational data mining. In: Džeroski, S., Lavrač, N. (eds.) Relational Data Mining, ch. 11, pp. 262–286. Springer, New York (2001)

[Provost and Domingos, 2001] Provost, F., Domingos, P.: Well-trained pets: Improving probability estimation trees. Technical Report CeDER #IS-00-04, New York University (2001)

[Shannon, 1948] Shannon, C.: A mathematical theory of communication. Technical report. Bell Systems Technical Journal (1948)

[Van Laer et al., 1997] Van Laer, W., De Raedt, L., Džeroski, S.: On multi-class problems and discretization in inductive logic programming. In: Raś, Z.W., Skowron, A. (eds.) ISMIS 1997. LNCS, vol. 1325, pp. 277–286. Springer, Heidelberg (1997)

Part II
Classification and Feature Extraction or Selection

Combination of Single Feature Classifiers for Fast Feature Selection

Hassan Chouaib, Florence Cloppet, and Nicole Vincent

Abstract. Feature selection happens to be an important step in many classification tasks. Its aim is to reduce the number of features and at the same time to try to maintain or even improve the performance of the used classifier. The selection methods described in the literature present some limitations at different levels. For instance, some are too complex to be operated in reasonable time or too dependent on the classifier used for evaluation. Others overlook interactions between features. In this paper, in order to limit these drawbacks, we propose a fast feature selection method. Each feature is closely associated with a single feature classifier. The weak classifiers we considered have several degrees of freedom and are optimized on the training dataset. Within the genetic algorithm, the individuals who are classifier subsets are evaluated by a fitness function based on a combination of single feature classifiers. Several combination operators are compared. The whole method is implemented and extensive trials are performed on four databases built from the MNIST handwritten digits database using four different descriptors. Results show how robust is our approach and how efficient is the method. On average, the number of selected features is about 70% smaller than the initial set while keeping the level of recognition rate.

1 Introduction

In many domains such as computer vision or pattern recognition, solving a problem is based on processing data extracted from a set of real world data acquired by means of sensors or resulting from some data processing. Data are structured as vectors. The quality of processing systems highly depends on the choice of the

Hassan Chouaib · Florence Cloppet · Nicole Vincent
Laboratoire LIPADE, Université Paris Descartes, France
e-mail: {hassan.chouaib,florence.cloppet,
nicole.vincent}@mi.parisdescartes.fr

F. Guillet et al. (eds.), *Advances in Knowledge Discovery and Management,*
Studies in Computational Intelligence 527,
DOI: 10.1007/978-3-319-02999-3_7,

vector contents. However, in many cases the vectors' high dimensionality makes it almost impossible to use them to solve the problem because of the data nature and of the learning set size. The high dimension of the representation space makes any learning set too sparse for using the common methods [Bins and Draper, 2001]. Hence it is usually recommended, and sometimes required, for example in bioinformatic studies or text analysis, to reduce the vector size in order to make data more usable. There is benefit even if the reduction might lead to loss of information. Sometimes, solving complex problems with large descriptors can also be accomplished using a small set of features selected from the initial data set. This can be done if the selected features are relevant enough with respect to the problem being considered [Zhou and Dillion, 1991]. According to the feature nature, feature selection can either improve the quality of the system if the eliminated features are the too noisy ones, or improve computation time when redundant or irrelevant features are present in the feature set.

Reducing vector dimensionality is often considered as a pre-processing step dedicated to noise and redundant information elimination. Among dimensionality reduction methods, feature extraction (the most representative is Principal Component Analysis) and feature selection can be considered. Here, we focus on feature selection. It consists of selecting the most relevant features from an initial set. Among the applications needing feature selection methods we can distinguish between clustering [Bouguila and Ziou, 2012] and classification. In this paper, only the classification problem is considered.

Existing feature selection methods reveal limitations on many levels such as complexity, interaction between the features, dependency on the evaluation classifier, and so on. In order to overcome these limitations, we introduce a new method for feature selection. It is based on selecting the best classifier combination from a set of simple classifiers. Each of these classifiers is built using a single feature and the selection is accomplished using a genetic algorithm. Moreover, the intermediate use of classifiers enable to handle a set of mixed numerical and symbolical features.

The paper is organized as follows. In Section 2, we motivate our choice for feature selection method by showing the limitations of existing methods. In Section 3, we introduce our **F**ast **F**eature **S**election **M**ethod (FFSM). In Section 4, different elements of the method are discussed and in Section 5, an extensive experimental study is carried out. Finally, conclusions are drawn and perspectives are given in Section 6.

2 Feature Selection

Feature selection is generally defined as a search process that finds a *relevant* feature subset from an initial feature set. The relevance of a feature subset always depends on the objectives and criteria of the problem to be solved.

A selection method [Liu and Yu, 2005] generally incorporates several phases (see Figure 1).

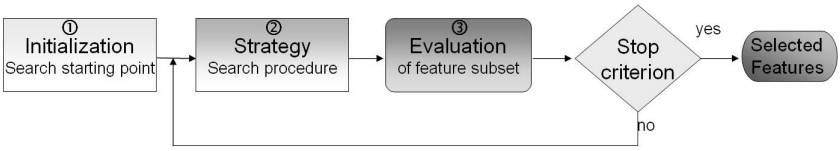


Fig. 1 Overview of a feature selection method

The two first steps initialize a search starting point and apply a search procedure. Once the subsets are generated, an evaluation is computed in the third step. Steps 2 and 3 are repeated until a stop criterion is satisfied. A search procedure consists of generating feature subsets which will be evaluated to select the best subset. In general, search strategies can be classified into three categories: exhaustive, heuristic and random. The evaluation function computes the suitability of the selected subset and compares it with the previous best candidate, replacing it if the current subset is estimated as being better.

Besides, feature selection algorithms may be classified into two categories depending on their evaluation procedure *filter or wrapper*. Feature statistical properties are taken into account in *filter* approach while *wrapper* methods are based on a classifier the efficiency of which is optimized by learning on a training data set while selecting the features. Pros and cons of both approaches are considered in the following sub-sections.

2.1 Filter Approach

The *filter* model (see Figure 2) was the first one used in feature selection. The used criterion for feature relevance evaluation is based on measures that rely on training data properties. Different measures [Guyon and Elisseeff, 2003] may be used such as correlation criterion [Hall, 2000], Fischer criterion [Furey et al., 2000], mutual information [Ben-Bassat, 1983], consistency [Dash and Liu, 2003] and signal to noise ratio. This type of method is usually considered as a pre-processing step (filtering) done before the training phase. In other words, evaluation is generally done independently of any classifier [John et al., 1994]. Methods that are based on this feature evaluation model often use a heuristic approach as search strategy [Chapelle and Vapnik, 2000].

The main advantage of filtering methods is their computational efficiency and robustness against over-fitting. Unfortunately, these methods do not take into account interactions between features and tend to select features that are redundant rather than complementary. Furthermore, these methods do not absolutely take into account the performance of classification methods subsequent to selection [Kohavi and John, 1997].

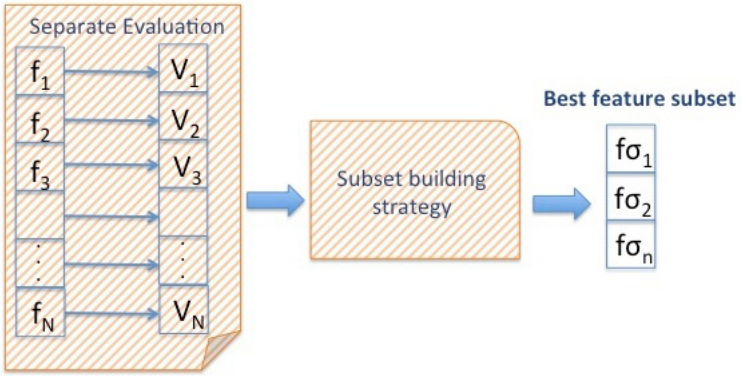


Fig. 2 General overview of a filter selection method

2.2 *Wrapper Approach*

As seen in the previous section, the main drawback of *filter* approaches is that they ignore the potential influence of the selected features on the performance of the classifiers to be used later. To solve this problem Kohavi and John introduced the concept of *wrapper* for feature selection [Kohavi and John, 1997]. The *wrapper* methods (see Figure 3) evaluate feature subsets on the basis of their classification performances using a learning algorithm.

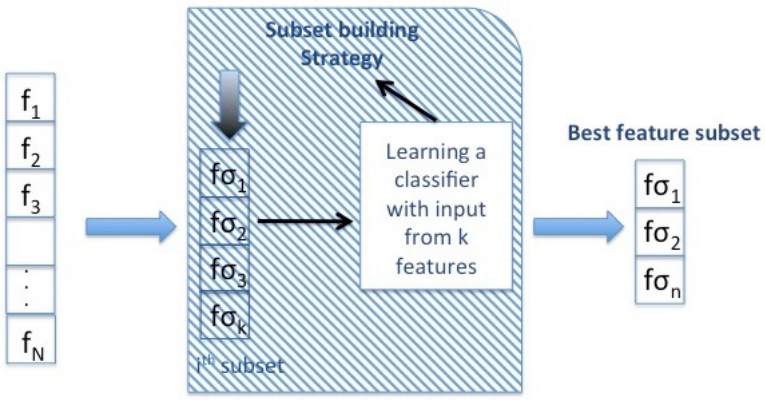


Fig. 3 General overview of a wrapper selection method

This evaluation is done using a classifier that enables to estimate the relevance of a given feature subset. The feature subset selected is always well adapted to the used classification algorithm but it is not necessarily valid if the classifier is changed. The

complexity of the learning algorithm makes the wrapper methods rather expensive regarding time complexity. It was shown that these methods generally give better results than filter methods, as they take into account both feature interactions and interactions between the classifier and the data set [Kohavi and John, 1997; Li and Guo, 2008; Huang et al., 2008].

However, they are time consuming because the learning step is performed with each feature subset. This main drawback makes it impossible to use an exhaustive search strategy (NP-complete problem), heuristics or random search strategies are then often preferred. Though, even in this case, the search becomes more and more unconceivable as the initial feature set size increases. An other drawback is the dependence of the relevant selected features on the used classifier. The feature evaluation is done using a chosen classifier during the selection phase. Each classifier has its own specificities and assumptions. But, if a change of classifier is required in order to better fit evolving data, the all selection process has to be restarted.

Genetic Algorithms seem to be a well suited search strategy used to take into account the dependency between features and to come as near as possible to the optimum. They are more global than forward or backward strategies.

2.3 Genetic Algorithms and Feature Selection

Genetic algorithms (GA) are one of the latest techniques in the field of feature selection [Kitoogo and Baryamureeba, 2007; Kim et al., 2000b; Oliveira et al., 2002; Yang and Honavar, 1998; Leardi, 1994]. Unlike classical feature selection strategies where one solution is optimized, a population of solutions can be modified at the same time. This can result in several optimal feature subsets as output. To apply a GA to solve a given problem, one should encode its potential solutions by finite strings of bits forming chromosomes. The main opened questions are the definition of an evaluation function, the *fitness* function, that allows good chromosome discrimination as well as the definition of genetic operators that will be used. The *fitness* function can be used either in *filter* or *wrapper* models.

The fitness evaluation of all chromosomes (coding each feature subset) in all generations can be very costly. This is particularly a problem for *wrapper* approaches where each chromosome is associated with a classifier that has to be trained and evaluated. To limit this problem we propose and describe in next section a new fast feature selection method (FFSM) that takes advantage of both *filter and wrapper* approaches

- As in *filter* methods, quality is associated with each feature.
- As in *wrapper* methods, the efficiency of a classifier is optimized. It is built on a feature subset, and do not need a new learning phase for each feature subset.

3 Proposed FFSM Method

On one side, filtering methods for feature selection have limitations with regard to the consideration of potential interactions between features. On the other side, *wrapper* methods present a very high time complexity and dependence on the classifier evaluation. Filtering methods derive their rapidity from taking into account features in an individual manner. We retain this idea by building a set of classifiers, each associated with one feature. They will be defined in Section 3.2. The overall vision of the *wrapper* method is preserved while considering a selection criterion that takes into account all the used features. This is implemented in a GA whose *fitness* function will be detailed in Section 3.3. Thus, we consider interactions between features. Finally, the features associated with the subset of classifiers selected at the last iteration represent the final feature subset. But first, in Section 3.1, an overall vision of FFSM method citeChouaib12 is given.

3.1 Selection Process

Let set $F = \{f_1, f_2, \ldots, f_N\}$ be composed of N features and $B_{app} = \{X_1, X_2, \ldots, X_M\}$ be a training dataset consisting of M samples where each $X_i = (f_{i1}, f_{i2}, \ldots, f_{iN})$ represents the i^{th} sample. A sample of dimension N is represented by a vector whose components are the values of features (f_i), where N is the total number of features. Let $Y = \{y_1, y_2, \ldots, y_M\}$ be the sample labels. For a bi-class classification problem we have $y_i \in \{-1, 1\}$. In order to minimize overfitting possibilities, the training set was divided into two parts: a training dataset A which contains M_A samples used to build the classifier set, a validation dataset V which contains other M_V samples used by the GA algorithm. Figure 4 represents the two-step process of our feature selection method:

- The construction of N simple classifiers H_i through a learning based on the dataset A. For each H_i, only the i^{th} feature f_i is taken into account.
- A selection among classifiers (H_i) by mean of a GA using the V dataset.

The first selection step consists in building a set of classifiers that represent the initial features which will be given as inputs to the GA. Each classifier is a simple model trained on a single feature. Once this classifier set is built, in the second step, we apply a genetic algorithm to select, after several generations, a *good* subset of classifiers. The features associated with the final selected models represent the final feature subset.

3.2 Classifier Set

In this step, a set of classifiers is built so that each classifier input is based on only one feature that can be either ordinal, with one or several dimensions, or symbolic. A classifier learnt on a single feature is a simple model defined using a learning

algorithm based on the content of the A training dataset. Such a classifier must be simple and as efficient as possible on a single feature.

It is among these classifiers that a subset, optimizing the defined criterion, will be extracted. This optimization will be carried out by a GA. It is then necessary to encode the subsets. The most classical way consists in encoding each possible solution by a binary string of size equal to the total number of classifiers, N. A gene of index i has the value 1 if the initial set i^{th} classifier is present in the current subset and 0 otherwise. We denote by $C = (c_1, c_2, \ldots, c_N)$ a chromosome where each c_i belongs to $\{0, 1\}$ and S_c is the set $\{i/c_i = 1\}$.

This type of coding prevents selecting more than once the same classifier for an individual. The control of the selected classifier number is left to the GA, which can be a major drawback in some applications.

The selection criterion, expressed in the GA's fitness function, is specified in the following section.

3.3 Selection Criterion

The problem is to find a subset having a reduced number of highly efficient classifiers. In *wrapper* methods, the *fitness* function is related to the building of a new classifier based on features that are involved in the individual (feature subset). To overcome the heaviness of this approach, we made a compromise. We build a new classifier that does not need a training phase but involves all the features present in the individual. Thus each selected classifier participates in the decision making. Therefore, we introduce, without any new learning phase, a classifier built as a combination of classifiers:

$$H^c = \boldsymbol{Comb}_{i \in S_c}(H_i) \tag{1}$$

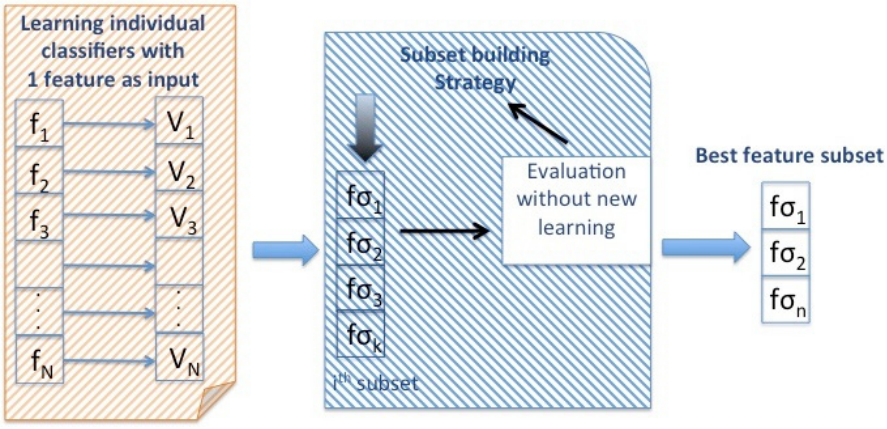


Fig. 4 General flow chart of the FFSM method

where $\boldsymbol{Comb}_{i \in S_c}(H_i)$ is the combination of the classifiers present in an individual and S_c is the set $\{i/c_i = 1\}$. Thus, for the GA fitness function, we compute the error given by this new classifier, related to the V sample set. It can be written as:

$$fitness = error(H^c) \tag{2}$$

Let see below an example showing how to compute the $fitness$ of an individual: Given a set of classifiers $H = \{H_1, H_2, \ldots, H_{12}\}$ which contain twelve classifiers. Suppose we want to combine classifiers whose answers are between -1 and +1 (-1 being associated with an element of the first class labeled -1 and +1 characterizing an element of the second class labeled +1). Let $I =$ "100110010110" be an individual for which six out of twelve classifiers are present. If we use the mean as a method for combining the classifiers then $\boldsymbol{Comb}_{i \in S_c}(H_i) = \frac{1}{6}(H_1 + H_4 + H_5 + H_8 + H_{10} + H_{11})$ the *fitness* function on I will be calculated as follows:

$$fitness(I) = error(sign(\boldsymbol{Comb}_{i \in S_c}(H_i)))$$

The ***Comb*** operator can take many forms, such as voting methods or mean approaches. Some of them will be discussed in Section 4.5.

4 Experiments

In this section we present the experiments carried out to illustrate the FFSM method. We first describe the used databases. In different studies the experiments are performed on different databases that have different properties. Then the comparisons are very difficult to draw. So we have decided to use a single application and several kinds of descriptors are applied to this single problem. Thus, this approach enables to define several databases with different dimensions as input of the system. The difficulty of the problem is exactly the same in the different cases but the descriptors have different properties (projection on base with loss of information, or raw data, ...). Before presenting the results and comparisons with other selection methods, we describe the problem and the experimental protocol. The different descriptors that are used are presented, they enable to build four different databases on which our trials are based. Then, we present the choice of implementation of our method at different levels, the classifiers, the combination method and the GA.

4.1 Problem and Material

For our experiments we used the *MNIST* database. It is a database of isolated handwritten digits (from 0 to 9) built in 1998 [Lecun et al., 1998]. Each digit is associated with an image of size 28×28 in 256 grey levels (example in Figure 5). The *MNIST* database is divided into two subsets, a training set of 60 000 examples and a test set of 10 000 examples.

We process *a priori* two-class problems, in the more general case of n classes it is necessary to build subsets within the labelled sample set to allow the use of a *one*

Fig. 5 Samples of images extracted from the *MNIST* database

versus all approach. Each subset is associated with a class. Let $A = \{A_i\}$, $V = \{V_i\}$ and $T = \{T_i\}$, which represents respectively training, validation and test datasets. A_i, V_i and T_i are constructed for the use of a *one versus all* method. On the one hand, each of the A_i datasets and T_i contain $2 * N_p$ samples: N_p samples of class i and N_p samples of all the other classes. On the other hand V_i only contains N_p elements: $\frac{N_p}{2}$ samples of class i and $\frac{N_p}{2}$ samples of all the other classes. For the *MNIST* dataset we have $i \in \{0,1,2,..,9\}$ and $N = 1000$.

4.2 Descriptors – Databases

We have used four descriptors for the representation of these data:

- Generic Fourier descriptor (GFD)[Zhang and Lu, 2002] is a descriptor based on the Fourier transform. The radial (***R***) and angular resolutions (***T***) represent two of its parameters.
- *R*-signature [Tabbone and Wendling, 2003] uses a Radon transform to represent an image.
- Zernike descriptor [Kim et al., 2000a] is a descriptor based on Zernike moments.
- Luminance of the 28×28 pixels.

Table 1 resumes the size of the feature vectors of each of these four data representations, constituting four experimental databases.

Table 1 Vector dimension (number of features) for each database

Name of database		***Dimension***
GFD	R=8,T=12	96
	R=10,T=15	150
R-signature		180
Zernike		66
Pixels		784

4.3 Classifier Sets

As we deal with numerical features, we propose to take advantage of the two following approaches to build a classifier set. One is to compute a single classification

threshold, the other, in order to improve the classifier efficiency, is to combine different weak classifiers in a strong classifier in order to obtain classifiers with multiple classification thresholds. This makes our approach original.

Case of Single Classification Threshold

In [Alamdari, 2006], a simple binary classifier is proposed in order to evaluate features individually. It was used as a selection criterion in a feature filtering method. The threshold that was used is the midpoint of a segment whose endpoints are the barycentre of data feature values in each class. In the following, we use the name *Classif_Alamdari* for this classifier. This classifier, for the i^{th} feature is defined by: Let $X^i = \{f_{1i}, f_{2i}, .., f_{Mi}\}$ be a feature value set where each element is the value of the i^{th} feature of one of the training samples. Let $X^{i,1} = \{f_{ki} | y_k = 1\}$ and $X^{i,-1} = \{f_{ki} | y_k = -1\}$.

$$y = sign((f_i - \frac{\mu_i^1 + \mu_i^{-1}}{2})(\mu_i^1 - \mu_i^{-1})) \tag{3}$$

Where μ_i^1 and μ_i^{-1} represent means of data for the i^{th} feature of class "1" and class "-1" respectively.

Another classifier of this type is the *decision stump*. It is a decision tree with only one internal node (the root) which is immediately connected to the terminal nodes [Iba and Langley, 1992]. It defines the best threshold that minimizes the classification error on a single feature.

Case of Multiple Classification Thresholds

To improve the efficiency of so simple classifiers presented in the previous paragraph, we propose to introduce multiple classification thresholds in the classifier building. To do that, we associate a threshold with the nodes of a decision tree [Breiman et al., 1984], or we use an AdaBoost [Freund and Schapire, 1995] algorithm from weak classifiers of type *decision stump* computed on different sample sets. Then, an H classifier is associated with feature f. This is illustrated in figure 6.

4.4 Genetic Algorithm

A genetic algorithm is composed of several parts. First a population has to be initialised, then the population evolves along iterations through genetic operators. The individuals are evaluated by the fitness function value and are introduced in the next generation by means of an elitist process. The evolution is stopped when some criterion is reached. The fitness function has been described in Section 3.3. As the Genetic Algorithm is not the purpose of the paper we just present here the choices that have been made and the parameter values that have been experimentaly fixed.

The initial population is composed of 200 chromosomes, a higher number of individual does not improve the results while increasing processing time. The

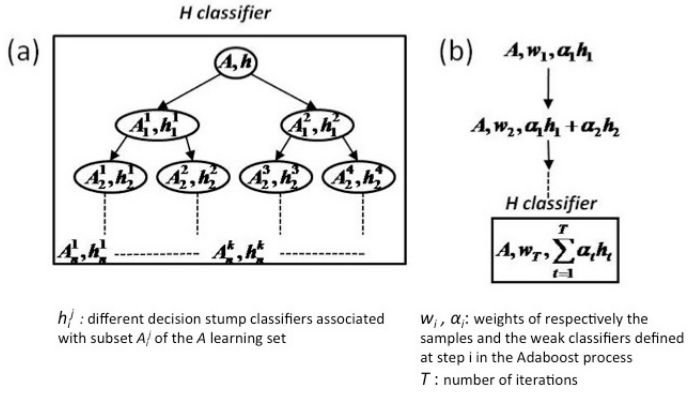


Fig. 6 Modeling of a Classifier H using multiple thresholds (a)Decision Tree (b)Adaboost

individuals have been randomly initialised with genes value being 0 or 1 according to a Bernouilli probability law with parameter p. This p parameter enables to handle the decrease of the number of selected features. The p value is here experimentaly fixed to 0.5. The genetic operators used are quite common. The crossover operator is a one-point crossover. And the mutation operator concernes the switching of one gene form 1 to 0 or from 0 to 1 according to the initial value with a probability fixed to 0.005. The members of a generation are becoming parents of next generation individuals using a tornament process where the best individual among 3 randomized individuals is selected. The stopping criterion is the maximal number of generations, it is set to 50 as we have experimented the evolution of the best individual quality is not more significant when this value is increased.

4.5 Classifier Combination

For combining classifiers, we used several conventional combining methods such as majority voting, weighted majority voting, mean, weighted mean and median. Another method that we used, is called *AWFO* (**A**ggregation **W**eight-**F**unctional **O**perators) [Dujet and Vincent, 1998]. In *AWFO* method, the assigned weights to each of the values given by the classifiers are adaptive. They do not only depend on each value but also on the general distribution of data. For the AWFO method we propose a modification to make it better adapted to our case. In the original version, it is assumed that the set of values to aggregate belongs to an interval on which the quality of values with respect to a goal is monotone (see Figure 7a). In our case of a two-class classification, the two classes both have an equivalent status, making it impossible to define a *distinguished value* with a significant value with respect to the problem (see Figure 7b).

In our case we want to combine classifiers whose answers are between -1 and +1. We can say the more a positive value is near +1, the more the element has a chance to belong to class labeled +1, and the more a negative value is near -1, the more the

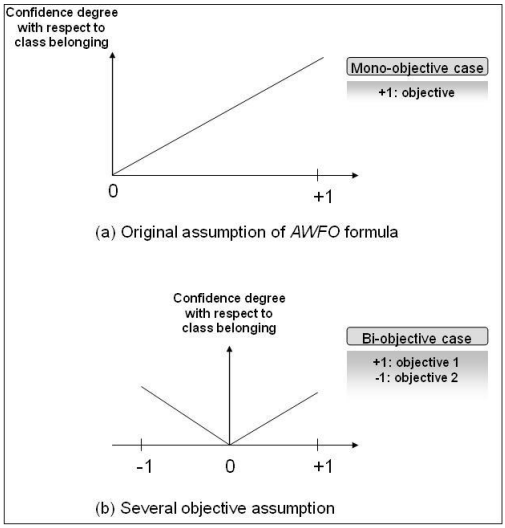


Fig. 7 Mono(a) and bi-objective (b) contexts of *AWFO* aggregation method

element has a chance to belong to class labeled -1. Therefore, we have chosen to aggregate separately the positive and the negative values. Refering to the original method, we have two *optimal* values. Thus, we have two *distinguished values*, -1 for negative values and +1 for positive values. The *AWFO* method does not only consider the classifier's answer but also the distribution of all answers to achieve aggregation.

Let us give an example to better understand the principle of the method. If we have the answers of ten classifiers (x_i for $i \in \{1,2,..,10\}$) with five positive answers and five negative answers, we propose to compute a weight for each positive answer while taking into account the other positive answers (respectively a weight for each negative answer while taking into account only the other negative answers). The weight of each answer ($W(x_i)$) is calculated using equation 4:

$$W(x_i) = \frac{d_{cum}(x_i)}{\sum\limits_{sign(x_j)=sign(x_i)} d(x_j)} \tag{4}$$

where

$$\begin{cases} d_{cum}(x_i) = \sum\limits_{j \in E_i} d(x_j) \text{ with } E_i = \{j/(d(x_j) \geq d(x_i)) \ \& \ (sign(x_j) = sign(x_i))\} \\ \text{and} \\ d(x) = 1 - |x| \end{cases}$$

In this formula, $d(x)$ is the distance between x and the associated distinguished value +1 or -1 according to the sign of x. Table 2 shows the details of this example.

Table 2 Combination example with the AWFO method

Initial answers	-0.2	0.4	-0.5	-0.7	0.8	0.1	-0.9	0.6	0.5	-0.3
Sorted answers (x_i)	-0.9	-0.7	-0.5	-0.3	-0.2	0.1	0.4	0.5	0.6	0.8
Distance (d)	0.1	0.3	0.5	0.7	0.8	0.9	0.6	0.5	0.4	0.2
Cumulative distance (d_{cum})	2.4	2.3	2	1.5	0.8	0.9	1.5	2	2.4	2.6
Final weight ($W(x_i)$)	1	0.95	0.83	0.62	0.33	0.34	0.57	0.77	0.92	1

Finally, in the case of negative answer very close to -1 and if we have a lot of negative answers, the cumulative sum of distances increases and its weight will be high. Similarly for an answer very close to +1.

5 Results

In this section, we show the contribution of combining methods used in our approach and we compare the results with those obtained by other feature selection methods. Indeed, the aim of the method is to select the lowest number of features, while keeping the efficiency of the recognizer system or even improving it. The quality of a recognition method is linked to the quality of the features, our purpose is not to solve the problem of figure recognition but to prove the efficiency of the FFSM method according to the feature nature.

5.1 Evaluation

In this section, we show the results obtained on the databases defined in Section 4.2 using different types of classifiers. The construction of the initial set of simple classifiers is made using one of the classifiers described in Section 4.3. The first considered classifier is an AdaBoost classifier. On the one hand, it finds several thresholds adapted to the learning examples, this is an advantage compared to classifiers based on a single threshold. On the other hand, the answer of this classifier is numeric: the sign indicates the class and the module gives a kind of confidence degree between 0 and 1. This output format allows the implementation of different combining methods.

After simple classifiers set building using the AdaBoost algorithm, and after best classifier subset selection for the different databases, an experimental study was carried out to evaluate the combining method's influence on the selected subset quality. Table 3 shows the average number of selected features for each descriptor on the ten classes of our experimental databases. We notice that the final subsets are on average 69.9% smaller than the initial set. We can also notice the regular aspect of the dimension reduction ratio as the normalized standard deviation values are low and similar.

To evaluate the quality of the subsets found by the FFSM method, we did not used the classifier involved in the GA selection process but we chose a classifier the

Table 3 Number of selected features in each experimental databse used for digit recognition

	Zernike	*GFD_8×12*	*GFD_10×15*	*R-signature*	*Pixels*	*Mean*
Initial	66	96	150	180	784	-
Mean	25	30	46	42	245	-
Standard Deviation (SD)	3.72	4.57	5.36	10.91	17.43	-
Normalized SD	0.15	0.15	0.12	0.26	0.07	0.15
% reduction	66.12	68.75	69.33	76.66	68.75	69.92

efficiency of which is generaly admitted, a SVM classifier learnt on training datasets A_i and tested on datasets T_i. Table 4 shows the classification average rate for each experimental database before and after selection.

Table 4 Results of a SVM classifier with and without selection for each experimental database used for digit recognition

	Without selection	*With selection*	*% variation*
Zernike	92.47 ± 3.99	92.42 ± 4.19	-0.04
GFD_8×12	92.38 ± 3.48	92.55 ± 3.35	+0.17
GFD_10×15	91.97 ± 4.10	92.10 ± 3.66	+0.13
R-signature	75.95 ± 7.67	79.55 ± 6.97	+3.60
Pixels	97.73 ± 1.03	97.60 ± 1.05	-0.13

We can notice that the rates before and after selection are relatively close regardless of the descriptor. In all the cases we managed to select feature subsets 69.9% smaller than the originals sets but with similar recognition rate. We notice the improvment of recognition rate occurs when the initial recognition rate is the lowest. The nature of the features can explain this fact. Besides, the rates depend on the digit recognized.

Finally we compared the selection results using different combining methods. Table 5 shows the comparison results averaged on the ten classes. In this table, we note that the results may be gathered in two groups. Within the two groups, the results are not significantly different. The three combining methods *AWFO, mean* and *weighted mean* are close for different databases and are more efficient than the *majority voting*, *weighted majority voting* and *median combining methods*.

We have also tested different types of classifiers using the previous combining methods. As the results are not significantly different we present in table 6 the best ones using the three different classifiers *Decision stump*, *Classif_Alamdari* and *Decision trees*.

We can notice from such results that AdaBoost classifiers give the best selection result. The obtained results from decision tree classifiers are close to those of AdaBoost.

Then the FFMS method as a whole, combining multiple views of the database, is not too sensible to the different choices that may be made in the various steps. We

Table 5 Comparison of the different combining methods

	Zernike	*GFD_8×12*	*GFD_10×15*	*R-signature*	*Pixels*	*Mean*
AWFO	92.25	92.55	92.08	79.19	97.60	90.74
Mean	92.42	91.90	91.95	79.04	97.40	90.54
Weighted mean	92.28	92.34	92.10	79.55	97.55	90.76
Majority voting	91.71	90.55	91.65	77.38	96.80	89.61
Weighted voting	91.95	91.95	90.98	79.05	97.20	90.22
Median	92.14	91.06	92.05	78.25	97.5	90.20
Without selection	92.47	92.38	91.97	75.95	97.73	90.10

Table 6 Comparison of results drawn from several classifiers

	Zernike	*GFD_8×12*	*GFD_10×15*	*R-signature*	*Pixels*
AdaBoost	92.42	92.55	92.1	79.55	97.6
Classif_Alamdari	91.50	90.15	90.85	74.15	93.85
Decison_stump	92.05	92.05	91.95	79.25	97.45
Decision trees	91.95	92.17	91.85	79.35	97.50

have here made the classifiers and the combining vary. In any application the user of the FFMS method may incorporate the elements he is the most familiar with or adapt some to its specific problem.

Using genetic algorithm the results may depend on the various runs of the process. Then we have run the process several times and we present in table 7 the best, the average and the standard deviation while applying 5 times the process using the same environment and choices for the different elements. The classifiers are Adaboost classifiers, the combination is an AWFO operator. We can notice the results are stable as the standard deviation is low.

Table 7 Stability of selection on the recognition rates

	Zernike	*GFD_8×12*	*GFD_10×15*	*R-signature*	*Pixels*
best	92.25	92.34	92.08	79.19	97.6
mean	92.08	92.21	91.98	78.89	97.45
standard-deviation	0.14	0.11	0.11	0.19	0.09
without selection	92.47	92.38	91.97	75.95	97.73

5.2 Comparison with Other Selection Methods

We compared our selection method with three other existing methods. These methods are based on *filter* and *wrapper* evaluation approaches. We considered three methods: *Relief* [Kira and Rendell, 1992], *SAC* [Kachouri et al., 2010] and the third one is a classic *wrapper* method based on random search and using the same GA as our method, with the same parameters but with a different *fitness* function

defined by the classification error of a *SVM* classifier. Table 8 shows the comparison of results between our method and other selection methods. Results are computed on the mean of ten digit classes using the same databases as in Section 4.2 taken from the *MNIST* dataset.

Table 8 Comparison with other methods for each experimental database

	Relief	***Wrapper_SVM***	***SAC***	***FFSM method***
Zernike	89.85	92.61	91.11	92.42
GFD_8×12	90.05	92.55	91.15	92.55
GFD_10×15	90.15	92.01	91.45	92.10
R-signature	73.55	80.05	75.88	79.55
Pixels	95.85	97.68	96.35	97.60

We can notice that our method is significantly better than *Relief* and *SAC* methods. These results are very close to the *Wrapper_SVM* method for all experimental databases (Table 8), but the computation time of our method is significantly lower than the one of the *Wrapper_SVM* method. Table 9 shows on the one hand, that our method, in worst case is 125 times faster and 250 times faster in the best case (for the database *Pixels*) and on the other hand, that the size of feature subsets selected by our method is 6% smaller in worst case and 15% smaller in the best case. The

Table 9 Comparison of computation relative time for feature selection and number of selected features with FFSM method and the *Wrapper_SVM* method

		Zernike	***GFD_8×12***	***GFD_10×15***	***R-signature***	***Pixels***
FFSM method	Nb of features	25	30	46	42	245
	Time	0.001	0.0015	0.0022	0.0026	0.004
Wrapper_SVM	Time	0.13	0.22	0.28	0.36	1
	Nb of features	36	52	65	79	299

reference time concerns the case of 784 features on a bi-class problem, processed by a matlab (c) software on a 2GHz processor computer. With the classic wrapper method, the duration is equal to 489 minutes, where as with our FFSM method duration is less than 2 minutes.

6 Conclusion

In this paper, a combination of single feature classifiers and a genetic algorithm are used to define a new fast feature selection method. The used fitness function is based on a combination of single feature classifier. Many classifiers and combining methods are possible and we have illustrated some of them, showing their efficiency. It

is obvious the user of the FFMS process may introduce its own choices either for the classifiers or the combination process. Our experiments on the four databases issued from the digit recognition problem using the MNIST dataset show that similar results can be obtained using about 69.9% less features for different descriptors whatever their properties are. Moreover, the proposed method is faster, in the worst case, 125 times than a classical wrapper method. The method can be adapted in any context as the simple classifier construction is free, the only constrain is to have a numerical output comprised between -1 and +1. The features may mix numerical and symbolical data. The selection is here presented as a selection method but may be applied at another level for descriptor selection. The method can be applied several times and then enables to define some hierarchical subset among the features. The experiment we have conducted on these four databases as well as on other databases, more precisely on databases studied in bioinformatics showed the stochastic aspect of our method was not leading to results with a too heterogeneous quality.

In most applications, the real problem is to find the best representation space, in which the problem is solved in the easiest way, that is to say where the error rate or evaluation indexes are optimum. To do so, a feature selection process enables to consider only relevant and robust features. These common features are mathematical functions with generic properties. The nature of the data is not taken into account in the learning phase of a classifier for example.

In our work, we have changed the features' definitions to replace them by new features that are given by the classifier functions. When the feature values are very intricate, the classifier function is a modification of the feature according to the data. This makes our method robust and not too much dependant on the general classifier used in the chosen representation space.

Indeed some improvements can be added. We here indicate some hints. Whereas only one criterion is used in the optimization phase, an error rate, some other properties of the features might be considered such as the classifier's diversity that could minimize redundancy between the selected features. Thus, a multi-objective approach can be used to integrate this new objective.

References

[Alamdari, 2006] Alamdari, A.: Variable selection using correlation and single variable classifier methods: Applications. In: Guyon, I., Nikravesh, M., Gunn, S., Zadeh, L. (eds.) Feature Extraction. STUDFUZZ, vol. 207, pp. 343–358. Springer, Heidelberg (2006)

[Ben-Bassat, 1983] Ben-Bassat, M.: Use of distance measures, information measures and error bounds in feature evaluation. In: Krishnaiah, P., Kanal, L. (eds.) Classification, Pattern Recognition and Reduction of Dimensionality. HandBook of Statistics II, vol. 2, pp. 773–791. North Holland (1983)

[Bins and Draper, 2001] Bins, J., Draper, B.: Feature selection from huge feature sets. In: Proceedings of the International Conference on Computer Vision (ICCV), pp. 159–165. IEEE (2001)

[Bouguila and Ziou, 2012] Bouguila, N., Ziou, D.: A countably infinite mixture model for clustering and feature selection. Knowledge and Information Systems 33, 351–370 (2012)
[Breiman et al., 1984] Breiman, L., et al.: Classification and Regression Trees. Chapman and Hall, New York (1984)
[Chapelle and Vapnik, 2000] Chapelle, O., Vapnik, V.: Model selection for support vector machines. In: Proceedings of the Neural Information Processing Systems, ANIPS 2000, Denver, Colorado, USA, pp. 230–236. MIT Press (2000)
[Dash and Liu, 2003] Dash, M., Liu, H.: Consistency-based search in feature selection. Artif. Intell. 151(1-2), 155–176 (2003)
[Dujet and Vincent, 1998] Dujet, C., Vincent, N.: Data fusion modeling human behavior. International Journal of Intelligent System 13, 27–39 (1998)
[Freund and Schapire, 1995] Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. In: Vitányi, P.M.B. (ed.) EuroCOLT 1995. LNCS, vol. 904, pp. 23–37. Springer, Heidelberg (1995)
[Furey et al., 2000] Furey, T.S., Cristianini, N., Duffy, N., Bednarski, D.W., Schummer, M., Haussler, D.: Support vector machine classification and validation of cancer tissue samples using microarray expression data. Bioinformatics 16(10), 906–914 (2000)
[Guyon and Elisseeff, 2003] Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. 3, 1157–1182 (2003)
[Hall, 2000] Hall, M.: Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning. In: 17th International Conference on Machine Learning, ICML 2000. LNCS, pp. 359–366. Morgan Kaufmann Publishers, San Fransico (2000)
[Huang et al., 2008] Huang, C.-J., Yang, D.-X., Chuang, Y.-T.: Application of wrapper approach and composite classifier to the stock trend prediction. Expert Syst. Appl. 34, 2870–2878 (2008)
[Iba and Langley, 1992] Iba, W., Langley, P.: Induction of one-level decision trees. In: Proceedings of the ninth International Workshop on Machine Learning, ML 1992, pp. 233–240. Morgan Kaufmann Publishers Inc., San Francisco (1992)
[John et al., 1994] John, G.H., Kohavi, R., Pfleger, K.: Irrelevant features and the subset selection problem. In: Machine Learning: Proceedings of the Eleventh International Conference, pp. 121–129. Morgan Kaufmann (1994)
[Kachouri et al., 2010] Kachouri, R., Djemal, K., Maaref, H.: Adaptive feature selection for heterogeneous image databases. In: Djemal, K., Deriche, M. (eds.) Second IEEE International Conference on Image Processing Theory, Tools 38; Applications, 10, Paris, France (2010)
[Kim et al., 2000a] Kim, H., Kim, J., Sim, D., Oh, D.: A modified zernike moment shape descriptor invariant to translation rotation and scale for similarity-based image retrieval. In: ICME 2000, p. MP5 (2000a)
[Kim et al., 2000b] Kim, Y., Street, W., Menczer, F.: Feature selection in unsupervised learning via evolutionary search. In: 6th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), pp. 365–369 (2000b)
[Kira and Rendell, 1992] Kira, K., Rendell, L.A.: The feature selection problem: Traditional methods and a new algorithm. In: AAAI, pp. 129–134. AAAI Press and MIT Press, Cambridge, MA, USA (1992)
[Kitoogo and Baryamureeba, 2007] Kitoogo, F.E., Baryamureeba, V.: A methodology for feature selection in named entity recognition. International Journal of Computing and ICT, 18–26 (2007)
[Kohavi and John, 1997] Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artif. Intell. 97, 273–324 (1997)

[Leardi, 1994] Leardi, R.: Application of a genetic algorithm to feature selection under full validation conditions and to outlier detection. Journal of Chemometrics 8(1), 65–79 (1994)

[Lecun et al., 1998] Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE, 2278–2324 (1998)

[Li and Guo, 2008] Li, Y., Guo, L.: Tcm-knn scheme for network anomaly detection using feature-based optimizations. In: Proceedings of the 2008 ACM Symposium on Applied Computing, SAC 2008, pp. 2103–2109. ACM, New York (2008)

[Liu and Yu, 2005] Liu, H., Yu, L.: Toward integrating feature selection algorithms for classification and clustering. IEEE Transations on Knowledge and Data Engineering 17, 491–502 (2005)

[Oliveira et al., 2002] Oliveira, L.S., Sabourin, R., Bortolozzi, F., Suen, C.Y.: Feature selection using multi-objective genetic algorithms for handwritten digit recognition. In: Proceedings of the 16th International Conference on Pattern Recognition, ICPR 2002, vol. 1. IEEE Computer Society, Washington, DC (2002)

[Tabbone and Wendling, 2003] Tabbone, S., Wendling, L.: Binary shape normalization using the Radon transform. In: Nyström, I., Sanniti di Baja, G., Svensson, S. (eds.) DGCI 2003. LNCS, vol. 2886, pp. 184–193. Springer, Heidelberg (2003)

[Yang and Honavar, 1998] Yang, J., Honavar, V.: Feature subset selection using a genetic algorithm. IEEE Intelligent Systems and their Applications 13(2), 44–49 (1998)

[Zhang and Lu, 2002] Zhang, D., Lu, G.: Shape based image retrieval using generic fourier descriptors. Signal Processing: Image Communication 17, 825–848 (2002)

[Zhou and Dillion, 1991] Zhou, X., Dillion, T.: A statistical-heuristic feature selection criterion for decision tree induction. IEEE Trans. Pattern Anal. Mach. Intell. 13, 834–841 (1991)

Classification of EEG Signals by an Evolutionary Algorithm

Laurent Vézard, Pierrick Legrand, Marie Chavent, Frédérique Faïta-Aïnseba, Julien Clauzel, and Leonardo Trujillo

Abstract. The goal of this work is to predict the state of alertness of an individual by analyzing the brain activity through electroencephalographic data (EEG) captured with 58 electrodes. Alertness is characterized here as a binary variable that can be in a “normal" or “relaxed" state. We collected data from 44 subjects before and after a relaxation practice, giving a total of 88 records. After a pre-processing step and data validation, we analyzed each record and discriminate the alertness states using our proposed “slope criterion". Afterwards, several common methods for supervised classification (*k* nearest neighbors, decision trees (CART), random forests, PLS and discriminant sparse PLS) were applied as predictors for the state of alertness of each subject. The proposed “slope criterion" was further refined using a genetic algorithm to select the most important EEG electrodes in terms of classification accuracy. Results show that the proposed strategy derives accurate predictive models of alertness.

1 Introduction

The electrical activity of the brain is divided into different oscillatory rhythms characterized by their frequency bands. The main rhythms in ascending order of

Laurent Vézard · Pierrick Legrand · Marie Chavent
IMB, UMR CNRS 5251, INRIA Bordeaux Sud-Ouest and Bordeaux Segalen University, France
e-mail: {laurent.vezard,marie.chavent}@inria.fr, pierrick.legrand@u-bordeaux2.fr

Frédérique Faïta-Aïnseba · Julien Clauzel
Bordeaux Segalen University, France
e-mail: frederique.faita@u-bordeaux2.fr,julien.clauzel@gadz.org

Leonardo Trujillo
Instituto Tecnológico de Tijuana, Mexico
e-mail: leonardo.trujillo@tectijuana.edu.mx

F. Guillet et al. (eds.), *Advances in Knowledge Discovery and Management,*
Studies in Computational Intelligence 527,
DOI: 10.1007/978-3-319-02999-3_8,

frequency are delta (1-3.5 Hz), theta (4-8 Hz), alpha (8-12 Hz) and beta (12-30 Hz). Alpha waves are characteristic of a diffuse awake state for healthy subjects and can be used to discern the normal awake and relaxed states, which is the topic of this experimental study. The oscillatory alpha rhythm appears as visually observable puffs on the electroencephalogram (EEG), especially over the occipital brain areas at the back of the skull, but also under certain conditions in more frontal recordings sites. The distribution of cortical electrical activity is taken into account in the characterization of an oscillatory rhythm. This distribution can be compared between studies reported in the literature through the use of a conventional electrode placement, the international system defined in [Jasper, 1958] and shown in Figure 1.

In this paper, a number is given to each electrode to simplify the interpretation of the figures. The number is incremented horizontally from the occipital electrodes to the frontal electrodes, left to right and top to bottom, as shown in Figure 2.

The brain electrical activity is non-linear and non-stationary, as specified in [Subasi et al., 2005]; i.e., EEG signals are time varying. EEG signals are almost always pre-treated before any further analysis is performed. Some authors [Ben Khalifa et al., 2005; Cecotti and Graeser, 2008] use the Fourier transform, others [Subasi et al., 2005; Hazarika et al., 1997] prefer to use a discrete wavelet decomposition. [Shaker, 2005] suggests to first use a wavelet decomposition and then to apply a Fourier transform to the result.

To predict the state of alertness, the most common method is neural networks (see for example [Subasi et al., 2005] or [Vuckovic et al., 2002]). However, the disadvantage of this approach is that it requires having a large set of test subjects relative to the number of predictive variables. To avoid this problem, [Subasi et al.,

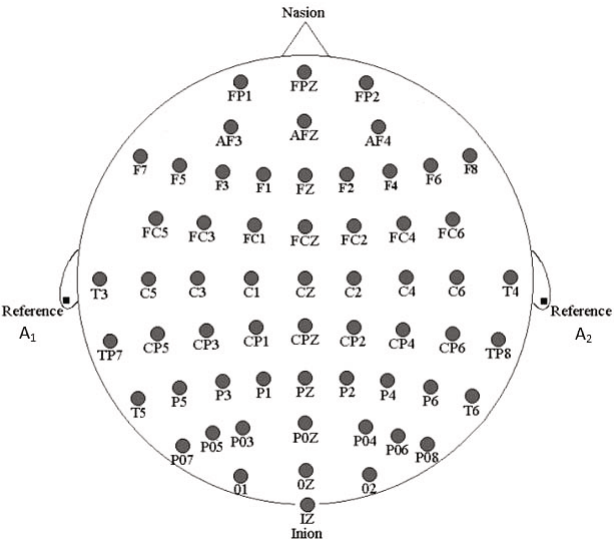


Fig. 1 Representation of the distribution of electrodes in the international system 10/10

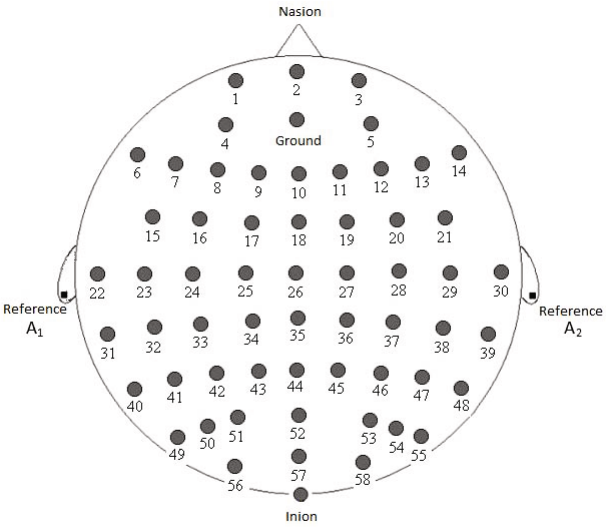


Fig. 2 Representation of the distribution of electrodes in the international system

2005] and [Vuckovic et al., 2002] split their signal into several segments of a few seconds, called "epochs". Other approaches use different statistical methods.

For example, [Yeo et al., 2009] use Support Vector Machine, [Anderson and Sijercic, 1996] use autoregressive models (AR) and [Obermaier et al., 2001] use hidden Markov chains.

In the present paper, wavelet decomposition is used as a pre-processing step and a new criterion for state discrimination is proposed. Then, several standard methods for supervised classification (binary decision tree, random forests and others) are used to predict the state of alertness of the participants. The criterion is then refined using a genetic algorithm to improve the quality of the prediction.

2 Data Acquisition

An experiment was conducted to obtain data for our study. This section will describe the participants, the experiment and will explain the data validation step.

2.1 Participants

This work uses 44 participants, of whom 26 are women, with ages between 18 and 35 and all are right-handed, to avoid variations in the characteristics of the EEG due to age or handedness linked to a functional interhemispheric asymmetry.

2.2 Procedure

The experiment was conducted individually in a soundproof room, where the participant was comfortably seated in front of the computer screen. It takes approximately two hours and a half to place the EEG cap and to perform a final explanatory interview with the participant. Data collection was controlled by the acquisition system Coherence 3NT (Deltamed, *http://www.natus.com/*). The data acquisition procedure is composed by five steps:

1. First EEG: the participant has to look at a cross (fixation point) at the center of the screen to reduce eye movements. This first recording corresponds to the reference state, considered as the normal vigilance state of the participant.
2. Attentional task devoted to collect contingent negative variation (CNV) (see section 2.3): The participant was instructed to press as quickly as possible on the spacebar of the keyboard in front of him at each appearance of a square which replaces the cross on the screen. For each appearance of this square, a warning sound (beep) presented 2,5 seconds before allowed the participant to prepare his response. The experimental session included 50 pairs of stimuli (S1: beep, S2: square), with a random amount of time elapsing between each pairs. The purpose of this task is specified in the next paragraph.
3. Relaxation session: The participant was fully guided by a soundtrack broadcast through loudspeakers placed in the room. The soundtrack suggested the participant to perform three successive exercises of self-relaxation, based on muscular relaxation and mental visualization. The purpose of this session is to try to bring the participant to a lower level of vigilance, qualified as the "relaxed" state.
4. Second EEG recording: 3 minutes of EEG were recorded with the same protocol as in the step 1. This second recording should reflect the relaxed state of the participant's brain if it was reached in the prior step.
5. Second CNV task: CNV is collected using exactly the same protocol as in step 2.

2.3 Contingent Negative Variation Extraction

CNV extraction has been performed by applying the Event-Related Potentials (ERPs) method [Rosenblith, 1959]. It consists, in the present experimental design, on averaging the electrical activity recorded in synchrony with all warning signals (S1: beep) until the response stimulus (S2: square). Such average allows event-related brain activity components, reflecting stimulus processing, to emerge from the overall cortical electrical activity, unrelated to the task performed. Thus in our paradigm, a negative deflection of the averaged waveform, called CNV, is obtained [Walter et al., 1964]. This attentional component has the property of decreasing in amplitude when the participant is less alert, either because he is distracted [Tecce, 1979], is deprived of sleep [Naitoh et al., 1971] or is falling asleep [Timsit-Berthier et al., 1981]. This fundamental result is shown in Figure 3. In this figure, the CNV is plotted as a dotted line for an alert participant and as a solid

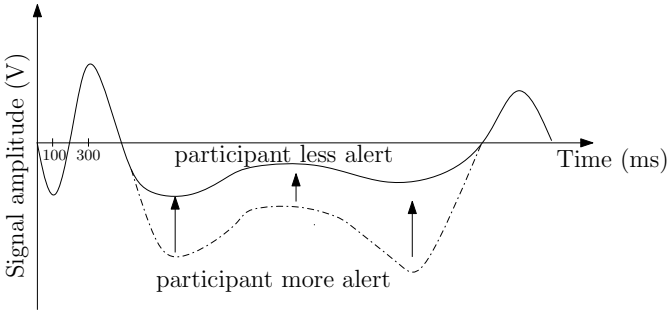


Fig. 3 Representation of the amplitude variation of the CNV with respect to the alertness of a participant

line for a participant which is less alert. The amplitude of the CNV is proportional to the alertness of the subject.

That is why, although the instruction given to the participant during CNV acquisition was to press the space bar as quickly as possible after the square appearance, the reaction time is not investigated in this study. However, the way the participant prepares to perform the task is observed.

The comparison of the amplitude of the CNV between tasks performed in steps 2 and 5 is used to determine if the alertness of a participant has changed. It allows us to know if he is actually relaxed. Only the positive cases, for which the amplitude of the CNV has significantly declined, were selected for comparative analysis of their raw EEG's (stages 1 and 4). Their EEG were then tagged respectively as "normal" or "relaxed" state. An example of a participant kept after studying his CNV is shown in Figure 4 and an example of a rejected participant is given in Figure 5.

In these figures, the solid curve represents the CNV recorded during step 2 and the dotted curve represents the CNV recorded in step 5. The solid vertical lines correspond to warning signals (S1: beep, S2: square). The area between the curve and the x-axis is calculated between T1 and T2 (section framed by the dotted vertical lines). A participant is kept if the area calculated with the CNV recorded in step 5 is lower than the area calculated with the CNV recorded in step 2. The study of CNV was performed on the 44 participants of the study and 13 participants were kept for further analysis.

Thus, an important number of participants are rejected. The stress due to the experiment and the duration of the installation of the cap may be factors that deteriorate the efficiency of the relaxation session. Moreover, to limit the duration of the cap wearing, the relaxation session is relatively short. Thus, it is possible that the duration of the relaxation session (20 minutes) is too short to achieve fully relax these subjects. The participants selected are those that succeed to relax in a relatively short period of time and in conditions that can be stressful. Those points can explain the high proportion of rejected participants in our study.

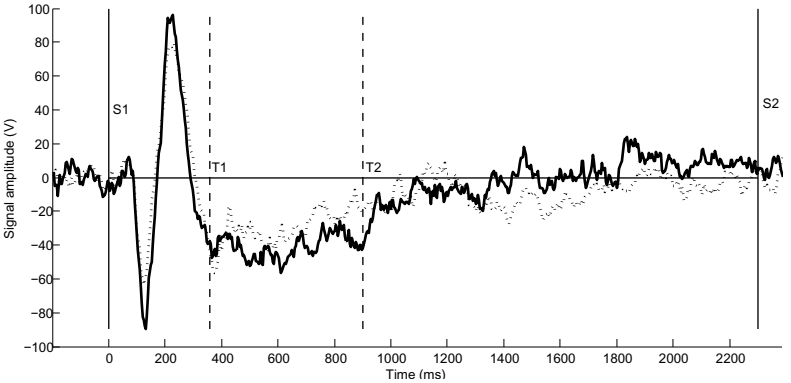


Fig. 4 Representation of CNV recorded on participant 4 during steps 2 (solid curve) and 5 (dotted curve). The solid vertical lines correspond to warning signals (S1: beep, S2: square). This participant is kept because the solid curve is mainly below the dotted curve between T1 and T2 (framed by the dotted vertical lines).

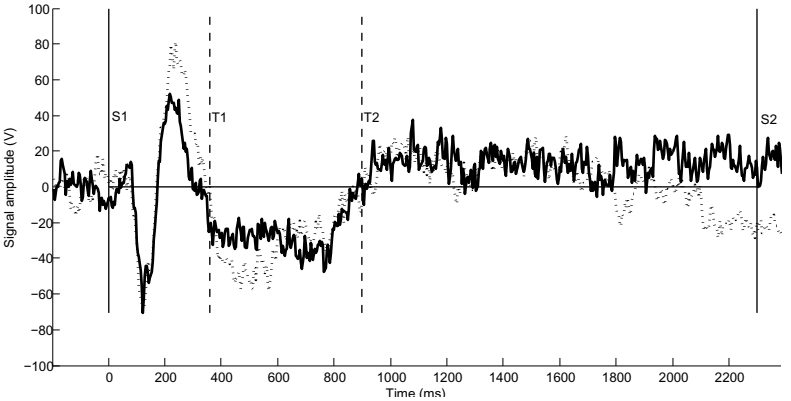


Fig. 5 Representation of CNV recorded on participant 9 during steps 2 (solid curve) and 5 (dotted curve). The solid vertical lines correspond to warning signals (S1: beep, S2: square). This participant is rejected because the solid curve is mainly above the dotted curve between T1 and T2 (framed by the dotted vertical lines).

2.4 Data

Finally, the data consist of 26 records of 3 minutes of raw EEG signals from 13 selected participants (one “normal” EEG and one “relaxed” EEG for each participant). Each record contains variations of electric potential obtained with a sampling frequency of 256 Hz (Deltamed acquisition system) with 58 active electrodes placed on a cap (ElectroCap). Using this sampling frequency, each signal recorded by an electrode for a given subject in a given alertness state contains 46000 data points. A representation of the data matrix is given in Figure 6.

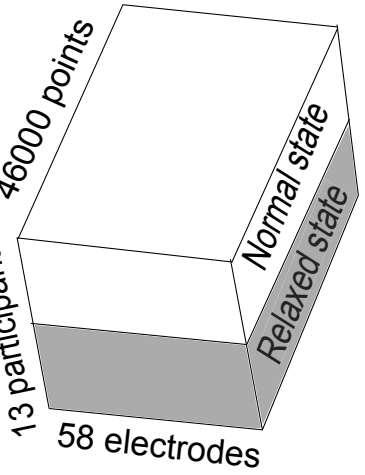


Fig. 6 Representation of the data matrix. There are three dimensions: one for the participants, one for the time (46000 points corresponding to the number of points in each 3 minutes EEG signals recorded using a sampling frequency of 256 Hz) and one for the electrodes.

3 Data Pre-processing

The data is specified in 3 dimensions (time, electrodes and participants). The proposed approach is to extract a feature in 2 dimensions to implement common classification tools. To do this, the signal energy, obtained by the wavelet decomposition, is considered.

3.1 Wavelet Decomposition

Wavelet decomposition [Daubechies, 1992; Mallat, 2008] is a method widely used in signal processing. Its main advantage is that it can be used to analyze the evolution of the frequency content of a signal in time. It is therefore more suitable than the Fourier transform for analyzing non-stationary signals.

A wavelet is a function $\psi \in L^2(\mathbb{R})$ such that $\int_{\mathbb{R}} \psi(t)dt = 0$. The continuous wavelet transform of a signal X can be written as

$$X(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} X(t)\psi\left(\frac{t-b}{a}\right) dt,$$

where a is called the scale factor that represents the inverse of the signal frequency, b is a time-translation term and function ψ is called the mother wavelet. The mother wavelet is usually a continuous and differentiable function with compact support. Several families of wavelet mother exist such as Daubechies wavelets or Coiflets.

It is also possible to define the discrete wavelet transform, starting from the previous formula and discretizing parameters a and b. Then, let $a = a_0^j$, where a_0 is the

resolution parameter such as $a_0 > 1$ and $j \in \mathbb{N}$ and let $b = kb_0a_0^j$, where $k \in \mathbb{N}$ and $b_0 > 0$. It is very common to consider the "dyadic" wavelet transform which corresponds to the case where $a_0 = 2$ and $b_0 = 1$. In this case, $j = 1, 2, \ldots, n$, where n is the base-2 logarithm of the number of points forming the signal and $k = 1, 2, \ldots, 2^{j-1}$. Then, the dyadic discrete wavelet transform is:

$$x_{j,k} = 2^{-\frac{j}{2}} \int_{-\infty}^{\infty} X(t)\psi(2^{-j}t - k)dt,$$

where j is the decomposition level (or scale) and k the time lag. The maximal number of decomposition levels, n, is the log_2 of the number of points forming the signal. The discrete wavelet transform is faster than the continuous version and also allows for an exact reconstruction of the original signal by inverse transformation. The dyadic grid provides a spatial frequency representation of discrete dyadic wavelet transform (see Figure 7). In this figure, the x-axis corresponds to time, the y-axis represents the frequencies and the circles correspond to the wavelet coefficients $x_{j,k}$. The signal points are represented below the last level of decomposition. At each additional level, the frequency is doubled.

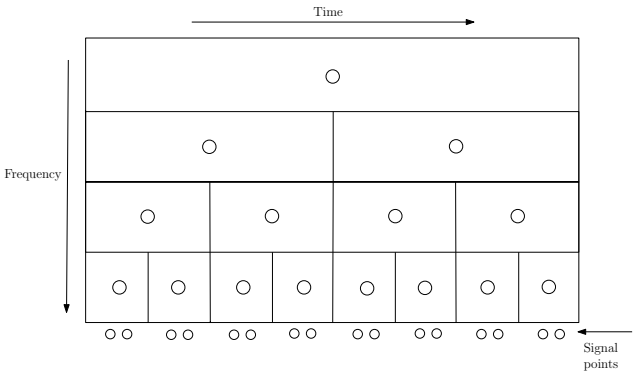


Fig. 7 Representation of the dyadic grid with 4 levels of decomposition

3.2 Signal Energy

Wavelet decomposition can also be used to calculate the energy of a signal for each level of decomposition. Thus, the energy e_j^2 of the signal X in the scale j is given by:

$$e_j^2 = \sum_{k=1}^{2^{j-1}} x_{j,k}^2, \forall j \in \{1, \ldots, 2^{j-1}\}.$$

In other words, from the dyadic grid, the energy associated with the scale j (decomposition level j) is equal to the sum of the squares of the coefficients of the line j. The use of signal leads to a loss of the temporality information. It is also possible

to obtain this result using a Fourier transform, however, the discrete wavelet decomposition provides more opportunities for further work. For example, the wavelet decomposition could be useful if the temporal evolution of the frequency content of signals is investigated in a future work.

3.3 Slope Criterion

For a given participant i ($i = 1,\ldots,13$) in a given state (normal or relaxed), each electrode m ($m = 1,\ldots,58$) provides a signal X_m. A discrete dyadic wavelet decomposition is performed on this signal by considering 15 scales ($15 = \lfloor log_2(46000 \rfloor)$, where 46000 is the number of points in each 3 minutes EEG signals and where $\lfloor . \rfloor$ is the integer part). From the coefficients obtained, the energy of the signal is calculated for each scale. Figure 8 presents these energies as a function of frequency.

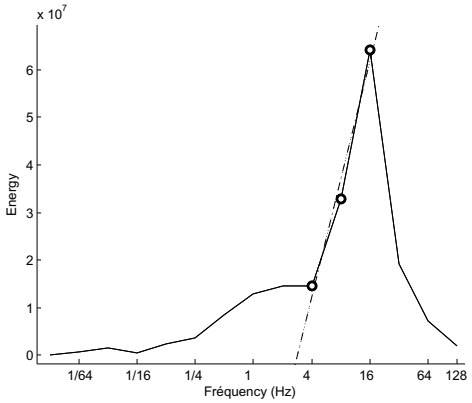


Fig. 8 Representation of the energy of signal X_m obtained using a discrete dyadic wavelet decomposition as a function of frequency. To calculate the slope criterion, a simple regression is performed (dotted line) on the energies calculated for 4, 8 and 16 Hz (circles).

The Alpha waves are between 8 and 12 Hz. Thus, according to the literature, only the energies calculated for 4, 8 and 16 Hz are used (black circles in Figure 8). Then, a simple regression is performed (dotted line in Figure 8) and the slope is retained. It seems more robust to use the slope (based on three points) than only one of these three points (minimum of the three points, maximum, etc.).

The slope coefficient is representative of the evolution of signal energy in the frequency considered. By repeating this process for each electrode, a feature of 58 coefficients (one per electrode) is obtained for an individual in a given state. Thus, a matrix of size 26×58 is obtained, representing the slope criterion. Some usual classification tools (classification and regression trees or k nearest neighbors for example) will be applied on this matrix in 2 dimensions.

4 Preliminary Results

The relevance of the slope criterion is illustrated in Figures 9 and 10. Figure 9 provides for each participant, in his state of "normal" alertness and his state of "relaxed" alertness, the sum of the slope criterions on all electrodes. It appears that for a given individual, the slope criterion is almost always lower when the individual is in the normal state than when he is in the relaxed state. Thus, by comparing, for a given individual, the values of the slope criterion for the normal and relaxed states it is possible to effectively distinguish the two states. However, for a new individual, a single record is known and the problem remains unsolved. Figure 10 shows for each electrode the sum of the slopes of the participants in a "normal" alertness state and participants in a "relaxed" state. The previous observation is also true at the electrode level. In fact, for a given electrode, the slope criterion is higher when considering the record obtained by this electrode after the relaxation.

By using the slope criterion, the signal given by one electrode is reduced to one real value. This approach implies a loss of information. Moreover, we are not interested here in the quality of the linear fit. However, despite this loss and the fact that the three points could be not aligned, Figures 9 and 10 show that the slope criterion allows discriminating the two alertness states (individual by individual). However, a strong inter-individual variability can be observed in Figure 9. Because of this strong individual variability, we cannot plot a line on Figure 9 which separates the two alertness states (represented by cross and circles). Then, for a given subject with

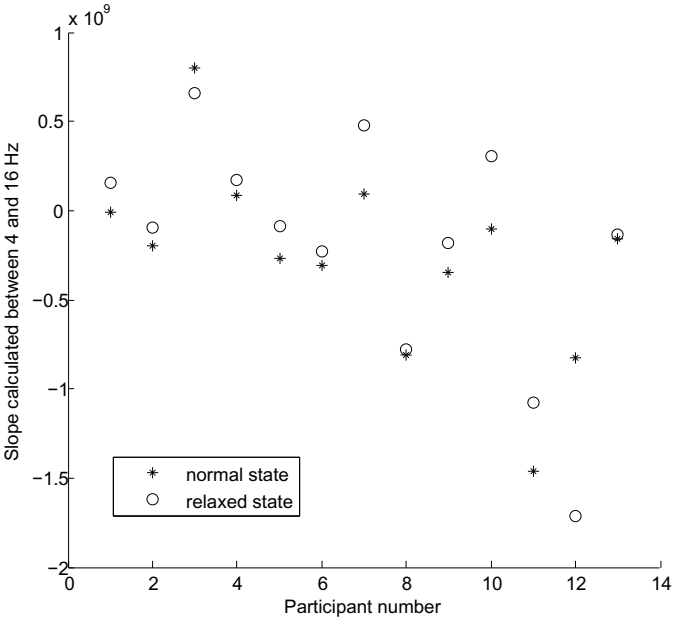


Fig. 9 Slope criterion summed over all electrodes for each of 13 participants

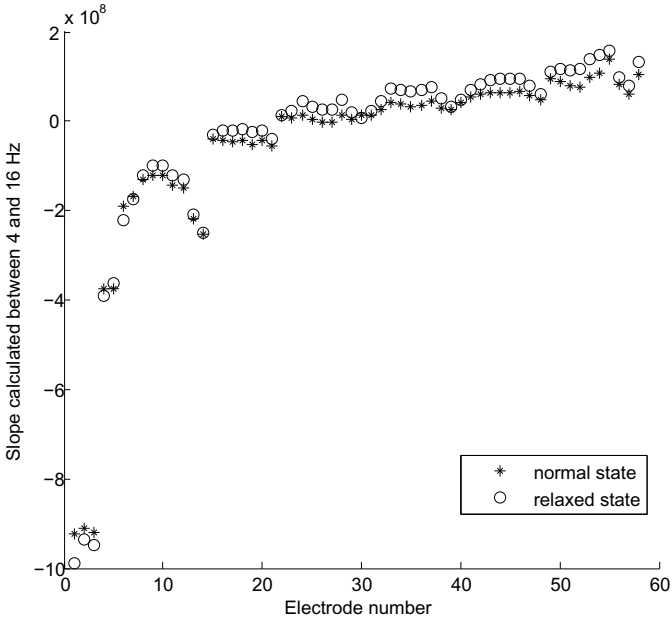


Fig. 10 Slope criterion summed over all participants for each of 58 electrodes

two EEG records, the slope criterion allows determining which record corresponds to the record done in the relaxed state. However, when only one record is known (new subject), we cannot classify it effectively.

At the beginning of this study, other approaches to obtain a summarized data matrix in two dimensions have been tested on similar signals [Vézard, 2010]. The goal was to obtain an approach which allows separating the two alertness states and reducing the inter individual variability observed. One of these approaches was based on the use of the Hölder regularity of the signal. The Hölder exponent [Jaffard and Meyer, 1996; Levy Vehel and Seuret, 2004] is a tool to measure the regularity of a signal at a given point. The smaller the Hölder exponent (respectively large) is, the more irregular (respectively smooth) is the signal. The Hölder exponent was estimated as defined in [Legrand, 2004]. The aim was to summarize the signal recorded by an electrode in its global regularity. An average of Hölder exponents for each point of the signal provided by an electrode was calculated.

Another approach was to analyze the alpha wave content in signals. Alpha rhythm is the classical EEG correlate for a state of relaxed wakefulness. When the person is relaxed, the neurons are synchronized and operate at a particular and identical rhythm. This rhythm appears to be responsible for the more pronounced appearance of Alpha waves [Niedermeyer and Lopes da Silva, 2005]. When the person is forced to perform a task that can break the relaxed state, the functioning of neurons vary widely. They seem to act by groups which do not work at a similar rhythm. Alpha waves are then masked by the more pronounced appearance of other waves (like

Beta waves). Thus, the idea was to measure the proportion of alpha waves in the signal (alpha waves divided by the sum of all waves: alpha, beta, teta and delta).

These two approaches gave a data matrix in two dimensions like that obtained with the slope criterion. However, they did not seem to work as well as the matrix of slopes to discriminate the two states of vigilance. In fact, graphs similar to Figures 9 and 10 can be obtained for these approaches. However, unlike the slopes criterion, no trend would appear from these graphs [Vézard, 2010]. Therefore, the slope criterion is investigated in this paper.

Common classification methods were initially used on the slope matrix to predict the alertness state of the participants. Predictive performance of k nearest neighbors (presented in [Hastie et al., 2009]), binary decision tree [Breiman et al., 1984] (CART), random forests [Breiman, 2001], discriminant PLS (by direct extension of the regression PLS method described in [Tenenhaus, 1998] recoding the variable to explain using dummy variables) and discriminant sparse PLS [Lé Cao et al., 2008] were studied. R packages "class", "rpart", "randomForest", "pls" and "SPLS" were respectively used to test these methods. Random forests have been applied by setting the number of trees at 15000 and leaving the other settings by default. Other methods were tuned by applying a 10 folds cross-validation on the training sample (number of neighbors for k nearest neighbors, complexity of the tree for CART, number of components for the discriminant PLS, number of components and value of the thresholding parameter for discriminant sparse PLS). The PLS method has been adapted for classification by recoding the variable to predict (alertness) using a matrix formed by an indicator of the modality ("normal" or "relaxed"). To compare the results, these methods were evaluated on the same samples (learning and test). A 5 fold cross-validation was used to calculate a classification rate. This operation was repeated 100 times to study the stability of classification methods with respect to the data partitioning.

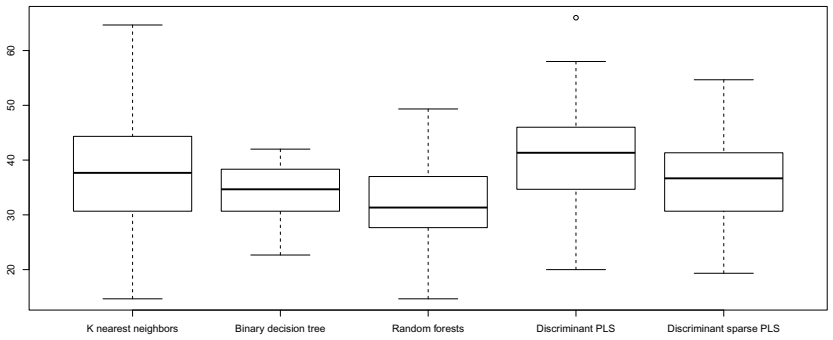


Fig. 11 Correct classification rate for the classification methods on the slope criterion

Table 1 Means and standard deviations of correct classification rate for the classification methods on the slope criterion

. .	*K* nearest neighbors	Binary decision tree	Random forests	Discriminant PLS	Sparse discriminant PLS
Mean	37.28	33.98	32.03	40.63	36.25
Standard deviation	10.47	5.15	6.46	8.55	7.96

The results are given by the boxplots in Figure 11. It appears that the median correct classification rate is very disappointing. It does not exceed 40% for most methods. Table 1 summarizes the means and standard deviations obtained using classification methods on the slope criterion. Large standard deviations reflect the influence of the data partitioning on the results. In the case of a binary prediction, these results cannot be satisfactory. It is likely that the inter-individual variability observed in Figure 9 has affected the performance of the classification methods. This inter-individual variability is very difficult to include in the classification methods with the available data for this study. Therefore, the pre-processing has been refined to obtain improved classification rates. Specifically, a genetic algorithm has been used as a feature selection process, to determine the electrode and the frequencies that provide the best discrimination for the slope criterion.

5 Feature Selection with a Genetic Algorithm

In this section, a genetic algorithm is used to improve the slope criterion. So far, previous work in the field, which suggested to focus on the alpha waves, was used. For this reason, the regression was done using frequencies between 4 and 16 Hz. Given the results, this approach will be refined. The algorithm searches for the best range of frequencies (not necessarily adjacent) to perform the regression. Similarly, so far all electrodes were kept. However, one objective of this work is to remove some electrodes to reduce the time required for the installation of the cap. Thus, the best combination electrode/frequencies based on the quality of the prediction is searched for. In this work, 58 electrodes and 15 decomposition levels are available. Thus, $58 * 2^{15} = 1900544$ ways exist to choose an electrode and a frequency range. To avoid an exhaustive search, the proposed approach is to use a genetic algorithm to perform a feature selection ([Broadhursta et al., 1997; Cavill et al., 2009]).

5.1 General Principle of a Genetic Algorithm

These optimization algorithms [De Jong, 1975; Holland, 1975] are based on a simplified abstraction of Darwinian evolution theory. The general idea is that a population of potential solutions will improve its characteristics over time, through a series of basic genetic operations called selection, mutation and genetic recombination or crossing. From an algorithmic point of view, the general principle is depicted in Figure 12.

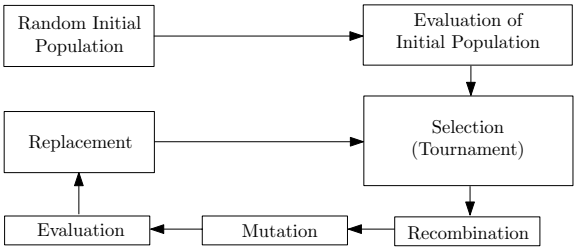


Fig. 12 Evolutionary loop of a basic Genetic Algorithm

The purpose of these algorithms is to optimize a function (fitness) within a given search space of candidate solutions. Solutions (called individuals) correspond to points within the search space, a random set of which are generated, this seeds the algorithm with an initial Population (set of individuals). They are represented by the genomes (binary codes or real numbers, with a fixed or variable size). All individuals are evaluated using a problem specific objective function called fitness. Individuals are selected based on their fitness (using a series of tournaments), these selected individuals are called Parents. These parents are used to generate new individuals using two basic genetic (search) operations, recombination (random recombination of two or more individuals) and mutation (random modification of a single individual). These newly generated individuals are called Offspring, since they share (genetic) similarities with the Parents used to generate them. Finally, the best individuals (amongst Parents and Offspring) are selected and replace the initial population. The algorithm is iterated until a stop criterion is reached; for instance, when all individuals are identical (convergence of the algorithm) or after a pre-specified number of iterations.

5.2 Algorithmic Choices

In this work, the genome is composed of 16 variables: the first, an integer ranging from 1 to 58, characterizes the number of the electrode selected, the 15 others are binary and correspond to the inclusion (or not) of each frequency to compute the slope criterion. An example of a genome is given in Figure 13. Each genome defines the electrode and the frequencies on which to perform the regression.

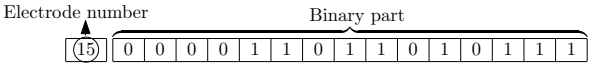


Fig. 13 Example of a genome in the genetic algorithm

5.2.1 Genetic Operators

The main search operators are mutation and crossover (recombination). To create a child, 2 parents are randomly selected. A tournament is performed to keep only the best individual (the one with the highest rating based on fitness). The selection pressure is not high (the best of 2) to maintain a high diversity in the population. The selection and the tournament is repeated twice in order to select two parents (tournament "winners"). Both parents are crossed and create a child. The child inherits the electrode which is located halfway between the electrodes of both parents. For the frequency crossover, it is a logical "AND" slightly modified in order to balance the production of 1 and 0. In fact, in this modified "and", when 1 and 0 are crossed (in this order), a 1 is obtained. Once the child is established, a mutation is applied. Each component of the genome of the child mutate with probability $1/8$. Thus, each child is, on average, affected by two mutations. When a mutation reaches the electrode number, a random number (drawn between 1 and 58) replaces the child electrode number. For the binary part, a mutation is the change of the binary variable (the 0 becomes 1 and vice versa).

5.2.2 Evaluation Functions

The genetic algorithm searches for the best combination of electrode / frequency range which achieves the highest prediction accuracy. Thus, it seems natural to rely on the correct classification rate (CCR). Then, the fitness function corresponds to the CCR obtained for each genome. These are then ranked in descending order of CCR. To compare each genome, the same samples are used to calculate the CCR using a 5 fold cross-validation. The evaluation step is done for each child at each iteration. Thus, it is necessary to use a fast classification method as evaluation function. In this work, two methods have been tested. The first is the single variable classification (SVC) [Guyon and Elisseeff, 2003], a method to predict from a single variable. The average for each modality (normal or relaxed) is calculated on the individuals in the training set for the variable (feature). Individuals of the test sample are then assigned to the class corresponding to the nearest average. The prediction is compared to ground truth which gives a CCR. The second method is the binary decision tree (CART) [Breiman et al., 1984]. Here, the algorithm is used with a single variable which guarantees fast calculation. Then, the fitness function for each genome X is written as:

$$f(x) = \frac{\#\text{ well classified participants of the test set}}{\#\text{ participants in the test set}}.$$

The genetic algorithm searches for the genome which maximizes f.

5.2.3 Stop Criterion

The algorithm stops if one of the following three conditions is satisfied:

- The number of iterations exceeds 1000.
- Parents are the same for 10 generations.

- The number of differences among the parents is less than 3.

To calculate the number of differences for a given population, denoted D, the genomes of the population at iteration i are stored in a matrix, denoted by P^i. Let P^i_j be the column j of the matrix P^i (where $j = 1, \ldots, 16$). Then $D = D_b + D_{elec}$ where:

- D_b is the number of differences for the binary part of P^i_j (columns 2 to 16). The number of differences for column P^i_j (where $j = 2, \ldots, 16$) is *min* (number of 0 in P^i_j, number of 1 in P^i_j).
- D_{elec} is the number of differences in P^i_1 (column corresponding to the electrode component). Then, D_{elec} is the number of individuals who have a electrode which is different from the electrode most selected in the population.

5.3 Results

The algorithm, programmed using Matlab, is run 100 times for each evaluation method with 300 parents and 150 children. The training and test sets are different for two different runs. Figure 14 gives CCR values for each run of the genetic algorithm with CART (stars) and SVC (circles).

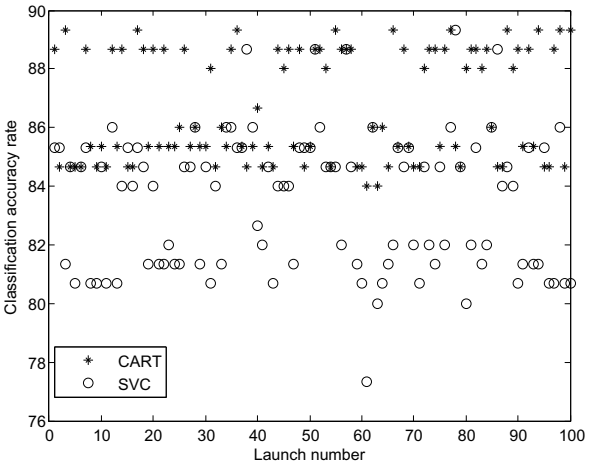


Fig. 14 Correct classification rates calculated with CART (stars) and SVC (circles) for each run of the genetic algorithm with 300 parents and 150 children

For each run, the algorithm is launched two times (one time with CART and the other time with SVC). During a run, CART and SVC use the same training and test sets in order to obtain comparable results. The correct classification rate obtained by CART (mean of 86.68% and standard deviation of 1.87%) exceed significantly (Mann-Whitney paired test with a p-value $= 5.57 * 10^{-14}$) those obtained by SVC

(mean of 83.49% and standard deviation of 2.37%), as mentioned in Table 2. At the end of the algorithm, some of the best genomes have the same evaluation (due to the low number of individuals and the evaluation method). It is therefore necessary to choose a genome (BEST) among those who have the same score. Thus, the best genomes at the end of each run of the algorithm are stored. The genome that appears most often is considered as the BEST for the evaluation method considered. The two BEST (for CART and SVC) get a correct classification rate equal to 89.33%. For CART, the BEST is obtained by performing regression between $1/8$, $1/4$, 2, 4 and 64 Hz on electrode F4 (right frontal area on Figure 1). For SVC, the BEST is obtained from electrode F2 (right frontal area) and the regression between $1/32$, $1/16$, 2, 4, 8, 64 and 128 Hz (see Table 3). Frequencies chosen for these genomes are more extensive than those used in the preliminary study.

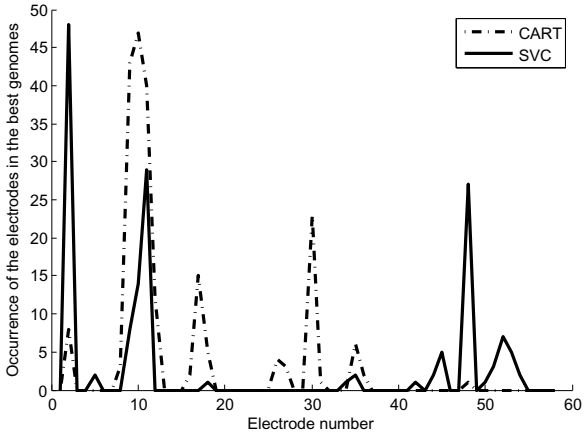


Fig. 15 Occurrence of the electrodes in the best genomes for each electrodes during the 100 runs of the genetic algorithm with 300 parents, 150 children and CART (dash-dotted curve) or SVC (solid curve)

Figure 15 gives the occurrence of the electrodes in the best genome over the 100 runs. When some genomes have the same CCR at the end of the run, we select the electrode chosen most often among the genomes with equal CCR. The algorithm running with CART selects the electrodes around the number 10 (FZ in Figure 1 and 2), 17 (FC1) or 30 (T4). With the SVC method, the electrodes around the 2 (FPZ), the 11 (F2) or the 48 (T6) are mostly chosen. Finally, on average, the population of the evolutionary algorithm converges in less than 50 iterations for both methods. Figure 16 gives the number of differences among parents for one run of the algorithm.

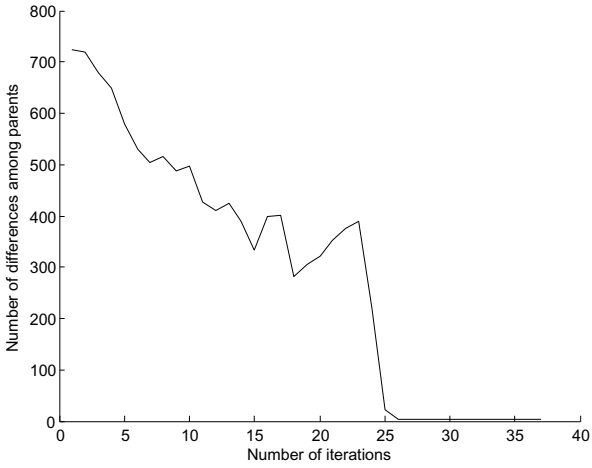


Fig. 16 Number of differences among parents for a run of the genetic algorithm with 300 parents, 150 children and SVC

Table 2 CCR for the two evaluation methods

Evaluation methods	CCR	
	mean	standard deviation
CART	86.68	1.87
SVC	83.49	2.37

Table 3 Summary table of results for best genomes

Evaluation methods	BEST genome		
	electrode selected	frequency selected (Hz)	CCR
CART	F4	1/8, 1/4, 2, 4 et 64	89,33%
SVC	F2	1/32, 1/16, 2, 4, 8, 64 et 128	89,33%

It shows that the number of differences among parents decreases very rapidly and falls below the threshold of 3 differences in less than 40 iterations. Then, one of the three stop conditions is satisfied and the algorithm stops.

Tables 2 and 3 summarize the CCR obtained by the genetic algorithm, which are better than those obtained (see Figure 11) with the criterion of the slopes calculated for frequencies between 4 and 16 Hz (alpha waves). Moreover, Table 4 shows that the genetic algorithm allows for a dimension reduction. SVC classifier can not be used with more than one variable. Then, Table 4 only shows a comparison between the results obtained in section 4 and those obtained with the genetic algorithm for the CART classifier.

Table 4 Comparison between CCR obtained in the preliminary study (1st row) and CCR obtained with the genetic algorithm (2nd row)

Evaluation methods	Number of electrodes in the predictive model	CCR	
		mean	standard deviation
CART	58	33.98	5.15
CART	1	86.68	1.87

It also appears that it is more appropriate to use a regression on frequencies of $1/8$, $1/4$, 2, 4 and 64 Hz for the signal of electrode $F4$ and the CART classifier. Then, our work allows us to accurately predict the state of alertness of a new individual. In fact, this electrode and this range of frequencies will be used to calculate the slope criterion for this individual. The CART decision tree, built on the sample formed by the 26 signals (13 study participants in both states of alertness) will be used as a classifier to predict his state of alertness.

6 Conclusion

In this paper, a method to predict the state of alertness of humans using their brain activity was studied. Initially, we proposed a criterion to obtain a summarized data matrix in two dimensions. Given the disappointing results obtained by classifying all of the available data, a genetic algorithm was used as a feature selection step to refine it. This allowed obtaining a reliable model (average of correct classification rate equal to 86.68% with a standard deviation of 1.87%). The algorithm also selects only a single electrode from the 58 that were initially available.

An exchange with neurobiologists now seems necessary to link the results obtained by the genetic algorithm to human physiology. We are performing a new campaign to collect EEG data and increase the number of participants included in the study. We believe that it will improve the precision of the estimate of CCR and so reduce the number of solutions which have the same score at the end of the genetic algorithm. In addition, an increase of the number of participants will allow using an external validation for the CCR at the end of the genetic algorithm.

It is possible to improve the genetic algorithm proposed in this paper. In fact, the improvement of the crossing and the introduction of new assessment methods are all paths that remain to be explored. A final interesting point concerns the transformation of the prediction obtained ("normal" state of alertness or "relaxed") to a probability. Using linear discriminant analysis or logistic regression as evaluation function should provide this probability directly.

Acknowledgements. The authors wish to thank Vérane Faure and Mathieu Carpentier, previously interns in the team.

References

[Anderson and Sijercic, 1996] Anderson, C., Sijercic, Z.: Classification of EEG signals from four subjects during five mental tasks. In: Proceedings of the Conference on Engineering Applications in Neural Networks, London, United Kingdom, pp. 407–414 (1996)

[Ben Khalifa et al., 2005] Ben Khalifa, K., Bédoui, M., Dogui, M., Alexandre, F.: Alertness states classification by SOM and LVQ neural networks. International Journal of Information Technology 1, 131–134 (2005)

[Breiman, 2001] Breiman, L.: Random forests. Machine Learning 45, 5–32 (2001)

[Breiman et al., 1984] Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and regression trees. Wadsworth Advanced Books and Software (1984)

[Broadhursta et al., 1997] Broadhursta, D., Goodacrea, R., Ah Jonesa, A., Rowlandb, J.J., Kelp, D.B.: Genetic algorithms as a method for variable selection in multiple linear regression and partial least squares regression, with applications to pyrolysis mass spectrometry. Analytica Chimica Acta 348, 71–86 (1997)

[Cavill et al., 2009] Cavill, R., Keun, H.C., Holmes, E., Lindon, J.C., Nicholson, J.K., Ebbels, T.M.: Genetic algorithms for simultaneous variable and sample selection in metabonomics. Bioinformatics 25, 112–118 (2009)

[Cecotti and Graeser, 2008] Cecotti, H., Graeser, A.: Convolutional neural network with embedded fourier transform for EEG classification. In: International Conference on Pattern Recognition, Tampa, Florida, pp. 1–4 (2008)

[Daubechies, 1992] Daubechies, I.: Ten Lectures on Wavelets. SIAM (1992)

[De Jong, 1975] De Jong, K.A.: An Analysis of the Behavior of a Class of Genetic Adaptive Systems. PhD thesis, University of Michigan (1975)

[Guyon and Elisseeff, 2003] Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research 3, 1157–1182 (2003)

[Hastie et al., 2009] Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd edn. Springer (2009)

[Hazarika et al., 1997] Hazarika, N., Chen, J., Tsoi, C., Sergejew, A.: Classification of EEG signals using the wavelet transform. Signal Processing 59, 61–72 (1997)

[Holland, 1975] Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)

[Jaffard and Meyer, 1996] Jaffard, S., Meyer, Y.: Wavelet methods for pointwise regularity and local oscillations of functions. Mem. Amer. Math. Soc. 123(587) (1996)

[Jasper, 1958] Jasper, H.H.: Report of the committee on methods of clinical examination in electroencephalography. Electroencephalography and Clinical Neurophysiology 10, 1–370 (1958)

[Lé Cao et al., 2008] Lé Cao, K.-A., Rossouw, D., Robert-Granié, C., Besse, P.: Sparse PLS: Variable selection when integrating omics data. Statistical Applications in Genetics and Molecular Biology 7(Article 35) (2008)

[Legrand, 2004] Legrand, P.: Débruitage et interpolation par analyse de la régularité Höldérienne. Application à la modélisation du frottement pneumatique-chaussée. PhD thesis, École Centrale de Nantes et Université de Nantes (2004)

[Levy Vehel and Seuret, 2004] Levy Vehel, J., Seuret, S.: The 2-microlocal formalism. In: Fractal Geometry and Applications: A Jubilee of Benoit Mandelbrot, Proc. Sympos. Pure Math., vol. 72, Part 2, pp. 153–215 (2004)

[Mallat, 2008] Mallat, S.: A Wavelet Tour of Signal Processing, 3rd edn. Academic Press (2008)

[Naitoh et al., 1971] Naitoh, P., Johnson, L.C., Lubin, A.: Modification of surface negative slow potential (CNV) in the human brain after total sleep loss. Electroencephalography and Clinical Neurophysiology 30, 17–22 (1971)

[Niedermeyer and Lopes da Silva, 2005] Niedermeyer, E., Lopes da Silva, F.: Electroencephalography, basic principles, clinical applications and related fields, 5th edn., ch. 9 (2005)

[Obermaier et al., 2001] Obermaier, B., Guger, C., Neuper, C., Pfurtscheller, G.: Hidden markov models for online classification of single trial EEG data. Pattern Recognition Letters 22, 1299–1309 (2001)

[Rosenblith, 1959] Rosenblith, W.: Some quantifiable aspects of the electrical activity of the nervous system (with emphasis upon responses to sensory stimuli). Revs. Mod. Physics 31, 532–545 (1959)

[Shaker, 2005] Shaker, M.: EEG waves classifier using wavelet transform and fourier transform. International Journal of Biological and Life Sciences, 85–90 (2005)

[Subasi et al., 2005] Subasi, A., Akin, M., Kiymik, K., Erogul, O.: Automatic recognition of vigilance state by using a wavelet-based artificial neural network. Neural Comput. and Applic. 14, 45–55 (2005)

[Tecce, 1979] Tecce, J.J.: A CNV rebound effect. Electroencephalography and Clinical Neurophysiology 46, 546–551 (1979)

[Tenenhaus, 1998] Tenenhaus, M.: La régression PLS, Théorie et Pratique (1998)

[Timsit-Berthier et al., 1981] Timsit-Berthier, M., Gerono, A., Mantanus, H.: Inversion de polarité de la variation contingente négative au cours d'état d'endormissement. EEG Neurophysiol. 11, 82–88 (1981)

[Vézard, 2010] Vézard, L.: Réduction de dimension en apprentissage supervisé. Applications à l'étude de l'activité cérébrale. Master's thesis, INSA de Toulouse (2010), http://www.sm.u-bordeaux2.fr/ vezard/wp-content/uploads/2012/05/rapport.pdf

[Vuckovic et al., 2002] Vuckovic, A., Radivojevic, V., Chen, A., Popovic, D.: Automatic recognition of alertness and drowsiness from EEG by an artificial neural network. Medical Engineering and Physics 24, 349–360 (2002)

[Walter et al., 1964] Walter, W.G., Cooper, R., Aldridge, V., McCallum, W.C., Winter, A.: Contingent negative variation: An electric sign of sensorimotor association and expectancy in the human brain. Nature 203, 380–384 (1964)

[Yeo et al., 2009] Yeo, M., Li, X., Shen, K., Wilder-Smith, E.: Can SVM be used for automatic EEG detection of drowsiness? Safety Science 47, 115–124 (2009)

Large Scale Image Classification: Fast Feature Extraction, Multi-codebook Approach and Multi-core SVM Training

Thanh-Nghi Doan and François Poulet

Abstract. The usual frameworks for image classification involve three steps: extracting features, building codebook and encoding features, and training the classifier with a standard classification algorithm (e.g. SVMs). However, the task complexity becomes very large when applying these frameworks on a large scale dataset like ImageNet containing more than 14 million images and 21,000 classes. The complexity is both about the time needed to perform each task and the memory and disk usage (e.g. 11TB are needed to store SIFT descriptors computed on the full dataset). We have developed a parallel version of LIBSVM to deal with very large datasets in reasonable time. Furthermore, a lot of information is lost when performing the quantization step and the obtained bag-of-words (or bag-of-visual-words) are often not enough discriminative for large scale image classification. We present a novel approach using several local descriptors simultaneously to try to improve the classification accuracy on large scale image datasets. We show our first results on a dataset made of the ten largest classes (24,807 images) from ImageNet.

1 Introduction

Image classification is one of the important research topics in the areas of computer vision, object recognition, and machine learning. Low-level local image features and the bag-of-words model (BoW) are the core of state-of-the-art image classification systems. The usual frameworks for image classification involve three steps:

Thanh-Nghi Doan
IRISA, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France
e-mail: thanh-nghi.doan@irisa.fr

François Poulet
Université de Rennes I, IRISA, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France
e-mail: francois.poulet@irisa.fr

F. Guillet et al. (eds.), *Advances in Knowledge Discovery and Management,*
Studies in Computational Intelligence 527,
DOI: 10.1007/978-3-319-02999-3_9,

1) extracting features, 2) building codebook and encoding features, and 3) training classifiers. Step 1 is to extract low-level local invariant features from images: the usual choices are SIFT [Lowe, 2004], SURF [Bay et al., 2008], and dense SIFT (DSIFT) [Bosch et al., 2007]. Step 2 is to build codebook and encode features: k-means clustering algorithm is the usual choice for building codebook, BoW model is the state-of-the-art of feature encoding. The image representation is obtained by applying the clustering algorithm and then constructing the histogram of each image SIFT distribution in the previously obtained set of clusters. Step 3 is to train classifiers: many systems choose either linear or non-linear kernel SVM classifiers. All these frameworks are evaluated on small datasets, e.g. Caltech 101 [Li et al., 2007], Caltech 256 [Griffin et al., 2007], and PASCAL VOC [Everingham et al., 2010] that can fit into desktop memory. However, the emergence of ImageNet [Deng et al., 2009] with more than 14 million images and 21,000 classes makes the complexity of image classification very large and difficult to deal with. This challenge motivates us to study an efficient framework in both computation time and classification accuracy. In this paper, we show how to address the challenge and achieve promising results over the state-of-the-art classification algorithms on ImageNet. We propose a fast and efficient framework for large scale image classification, as shown in Fig. 1. Our key contributions include:

1. A parallel version of LIBSVM to deal with a large scale dataset in reasonable time.
2. A novel approach using several different local robust descriptors and how to combine them efficiently by using multi-feature and multi-codebook approach.

The remainder of this paper is organized as follows. Section 2 briefly reviews the related work on large scale classification and image representation. The benchmark datasets in computer vision are introduced in section 3. In section 4, we present the efficient low-level local image features for many vision tasks. Our multi-feature and multi-codebook approach and parallel LIBSVM are described in section 5. Section 6 presents numerical test before the conclusion and future work.

2 Related Work

Large Scale Classification: Many previous works on image classification have relied on BoW models [Csurka et al., 2004], local feature quantization, and support vector machines. These models can be enhanced by multi-scale spatial pyramids (SPM) [Lazebnik et al., 2006] on BoW or histogram of oriented gradient (HoG) [Dalal and Triggs, 2005] features. Fergus *et al.* [Fergus et al., 2009] study semi-supervised learning on 126 hand labeled Tiny Images categories, Wang *et al.* [Wang et al., 2009] show classification on a maximum of 315 categories. Li *et al.* [Li et al., 2009] do research with landmark classification on a collection of 500 landmarks and 2 million images. On a small subset of 10 classes, they could improve BoW classification by increasing the visual vocabulary up to 80K visual words. To make large scale learning more practical, many researchers are beginning to study strategies where the original data in low-dimensional space is often

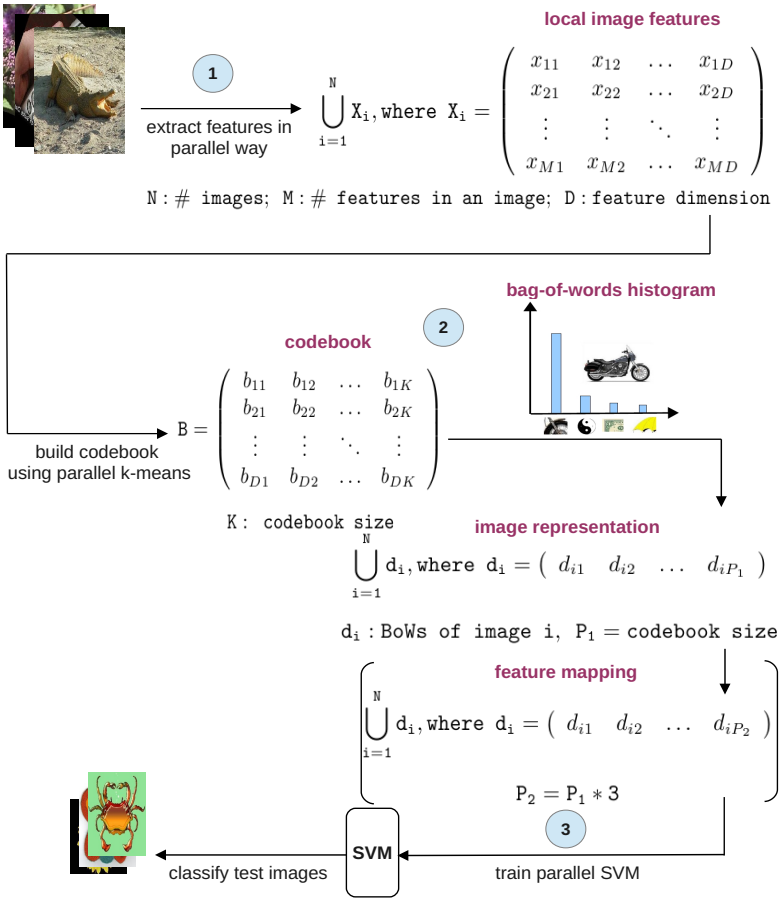


Fig. 1 The overview of our framework for large scale image classification

transformed to high- dimensional space by a nonlinear mapping induced by a particular kernel and then efficient linear classifiers are trained on the resulting space [Deng et al., 2010], [Perronnin et al., 2010]. Some recent works consider exploiting the hierarchical structure of dataset for image recognition and achieve impressive improvements in accuracy and efficiency, but has not evaluated classification minimizing hierarchical cost. Related to classification is the problem of detection, often treated as repeated 1-vs-all classification in sliding windows. In many cases, such localization of objects might be useful to improve classification, but even the most efficient of state-of-the-art techniques [Vedaldi et al., 2009; Everingham et al., 2010] take a lot of computation time and thus it is very difficult to deal with large scale datasets. The difference between our work and previous studies is to take into account parallel algorithms to speedup two processes: extracting features and training classifiers. Our experiments show first promising results

improving both classification time and accuracy and confirm that parallel algorithms are very essential for large scale image classification in terms of time efficiency.

Image Representation: Local image features and BoW model are the core of state-of-the-art image classification systems. Representing an image based on BoW model includes the three following steps: 1) feature detection, 2) feature description, and 3) codebook generation. Recent works have studied these steps and achieved impressive improvements. However, in each processing step there exists a significant amount of lost information, and the resulting visual-words are often not discriminative enough for large scale image classification applications. Many different approaches have been proposed to improve the discriminative power during these steps. At the feature detection step, multiple local features are grouped to obtain a more global and discriminative feature. At the feature description step, high-dimensional descriptors or descriptors enhanced by other information have been studied to get more image information [Winder and Brown, 2007]. At the codebook generation step, many previous works have proposed efficient quantizers or codebooks that reduce quantization errors and preserve more information of feature descriptors [Moosmann et al., 2006], [Philbin et al., 2008]. We have a more general view for all these three steps and propose a novel approach that combines both multi-feature and multi-codebook approach to construct the final image representation. Our approach aims to increase the discriminative power of image representation by embedding more useful information from the original image features. In multi-feature and multi-codebook approach, first BoW histograms of images for each feature channel is constructed based on their corresponding codebook. The result is a bag-of-BoW for all different feature types extracted in step 1 and we call it a bag-of-visual packets or a bag-of-packets (BoP). Finally, all BoW histograms in BoP are concatenated to form the final image representation, as shown in Fig. 3. In our novel approach the final image representation is constructed by using parallel multi-feature and multi-codebook computation, improving the discriminative power of image representation for large scale image classification. These are the major differences between our approach and previous studies.

3 Datasets

There are quite a few benchmark datasets for image classification, such as MNIST (`http://yann.lecun.com/exdb/mnist`), Caltech 101, Caltech 256, PASCAL VOC, etc. However, there are very few multi-class image datasets with many images for more than 300 categories. In recent years, there is an agreement that it is necessary to build a large scale dataset for studying object retrieval and recognition systems. One is Tiny Images [Torralba et al., 2008], 32×32 pixel versions of image collected by performing web queries for nouns in the WordNet hierarchy [Fellbaum, 1998], without verifying the content. The other one is ImageNet, a large-scale ontology of images built upon the backbone of the WordNet structure. The images are also collected from web searches for the nouns in WordNet, but the content of images are verified by human labelers. ImageNet is much larger in scale and diversity

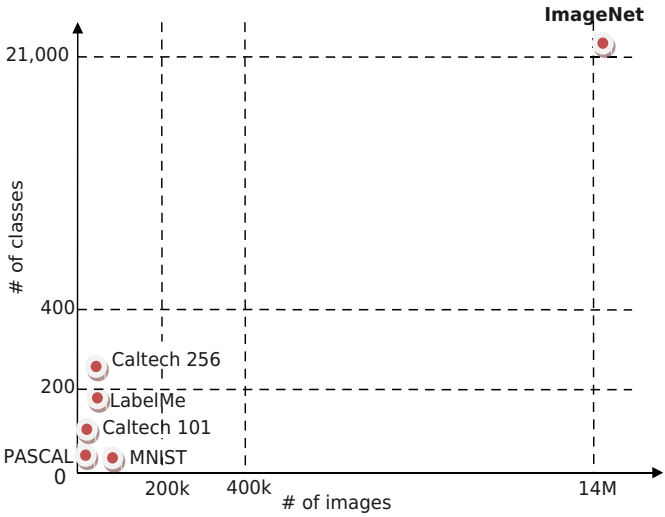


Fig. 2 A comparison of ImageNet with other benchmark datasets

and much more accurate than the current image datasets. The current released ImageNet has grown a big step in terms of the number of images and the number of classes, as shown in Fig. 2 - it has 21,841 classes with more than 1000 images for each class on average. Positively, it is necessary to have many images in the same class to cover visual variance, such as illumination, view point changes, and different appearance, even if in the dataset, some classes have only one or less than 10 images so machine learning algorithm cannot learn anything.

4 Low-Level Local Image Features

As shown in Fig. 1, given a set of input images, our system first extracts SIFT, SURF, and DSIFT features. These features have been proven to be efficient in various vision tasks such as object recognition, texture analysis, scene classification, etc.

4.1 SIFT

SIFT (Scale-invariant feature transform) is an algorithm proposed by [Lowe, 2004] to detect and describe local features in images. Extracting SIFTs consists of four key stages: scale-space extrema detection, keypoint localization, orientation assignment and keypoint descriptor. The first stage uses Difference-of-Gaussian function (DoG) to identify candidate interest points that are invariant to scale and orientation. DoG is used instead of Gaussian to speedup the computation.

In the keypoint localization stage, they reject the candidate points that have low contrast or are poorly localized along an edge. Hessian matrix is used to compute

the principal curvatures and eliminate the keypoints that have a ratio between the principal curvatures greater than the threshold. An orientation histogram is formed from the gradient orientations of sample points within a region around the keypoint in order to get an orientation assignment. According to the paper's experiments, the best results are achieved with a 4×4 array of histograms with 8 orientation bins in each. So the SIFT descriptor used is $4 \times 4 \times 8 = 128$ dimensions.

4.2 SURF

SURF (Speeded Up Robust Feature) is a robust image detector and descriptor presented by [Bay et al., 2008]. The standard version of SURF is several times faster than SIFT and claimed by its authors to be more robust against different image transformations than SIFT. SURF is based on sums of 2D Haar wavelet responses and makes an efficient use of integral images.

SURF is partly inspired by the SIFT descriptor and has slightly different ways of detecting features. It uses an integer approximation to the determinant of Hessian blob detector, which can be computed extremely quickly with an integral image. For features, it uses the sum of the Haar wavelet response around the point of interest. Again, these can be computed with the aid of the integral image.

4.3 DSIFT

DSIFT is a variant of SIFT descriptors that is extracted at multiple scales. It is roughly equivalent to running SIFT on a dense grid of locations at a fixed scale and orientation. This type of feature descriptors is often used for object categorization.

- **Bin size vs. keypoint scale.** DSIFT specifies the descriptor size by a single parameter, size, which controls the size of a SIFT spatial bin in pixels. In the standard SIFT descriptor, the bin size is related to the SIFT keypoint scale by a multiplier, denoted magnif, which defaults to 3. As a consequence, a DSIFT descriptor with bin size equal to 5 corresponds to a SIFT keypoint of scale $5/3 = 1.66$.
- **Smoothing.** The SIFT descriptor smoothes the image according to the scale of the keypoints (Gaussian scale space). By default, the smoothing is equivalent to a convolution by a Gaussian of variance s^2 where s is the scale of the keypoint and 0.25 is a nominal adjustment that accounts for the smoothing induced by the camera CCD.

5 Classifiers

In various applications, kernel machines such as Support Vector Machines (SVM) have been used with impressive success often delivering state-of-the-art results. Using the kernel trick, they are applied in several domains and even enable heterogeneous data fusion by concatenating feature spaces or multiple kernel learning. Before performing image classification, we apply multi-feature and multi-codebook

approach to construct the final image representation for all images in dataset. To stick to the efficient linear classifier, we use the explicit feature mapping approach from [Vedaldi and Zisserman, 2012] to improve the accuracy performance of image classification.

5.1 Multi-feature and Multi-codebook

As mentioned in section 3, the images in the same class of ImageNet usually have high intraclass variability. This variability poses more challenges for image classification systems. Many previous works want to design a robust image feature which is invariant to image transformation, illumination and scale change [Lowe, 2004; Bay et al., 2008; Bosch et al., 2007; Tola et al., 2010]. There are some improvements when using these robust features for image classification, but it is easy to realize that none of the feature descriptors have the same discriminative power for all classes. For instance, the features based on texture analysis and shape might be useful when classifying the photos with the same geometric direction. However, it will not be sufficient when the images are rotated or the objects are taken a shot in different camera angles. In this case, the appropriate choice should be the features based on interesting keypoints (e.g. SIFT). Obviously, instead of using a single feature type for all classes we can combine many different feature types to get higher improvement in classification accuracy. In this section, we present a novel multi-feature and multi-codebook approach and demonstrate how to combine these features.

Let a set of all different feature descriptor types extracted from an image i be $F = \{f_i^j\}$, where f_i^j are the descriptors of feature type j extracted from image i, M is the number of feature types, and $j = 1,..,M$. Our approach is that BoW histograms of each feature type are constructed based on their corresponding codebook, as shown in Fig. 3. Instead of using a single codebook for constructing the final image presentation, we use multiple codebooks $\{C^1, C^2, \ldots, C^M\}$ that are built from different feature types. More specifically, the codebook C^j is used to construct BoW histogram h_i^j for feature descriptors $f_i^j \in F$. Then all BoW histograms h_i^j are concatenated to form the final image representation H_i. As a result, for each image i, we obtain H_i with M elements $H_i = \{h_i^1, h_i^2, \ldots, h_i^M\}$. For simplicity, we call H_i a ¨bag- of- packets¨ (BoP) that is the final image representation constructed based on different codebooks of the original image i. A BoP is more discriminative than an usual BoW because two BoPs H_i and H_j are considered identical if and only if their corresponding BoWs are identical. Formally, it takes the intersection of the BoWs elements from multiple features:

$$(H_i = H_j) \equiv (h_i^1 = h_j^1) \wedge (h_i^2 = h_j^2) \wedge \ldots \wedge (h_i^M = h_j^M) \quad (1)$$

Obviously, this approach improves the discriminative power of the final image representation more than the classical approach with a single codebook.

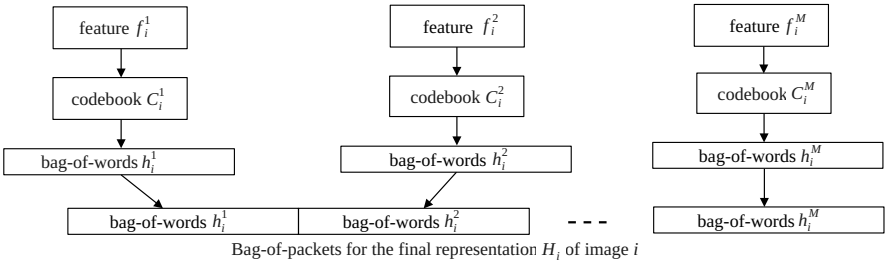


Fig. 3 Constructing bag-of-packets based on multi-feature and multi-codebook approach

5.2 Parallel LIBSVM (pLIBSVM)

LIBSVM is an integrated software for support vector classification, (C-SVC, nu-SVC), regression (epsilon-SVR, nu-SVR) and distribution estimation (one-class SVM). It supports multi-class classification. Since version 2.8, it implements an SMO-type algorithm [Chang and Lin, 2001]. LIBSVM provides a simple interface where users can easily link it with their own programs. Main features of LIBSVM include: different SVM formulations, efficient multi-class classification, cross validation for model selection.

Keerthi *el al.* [Keerthi and Lin, 2003] present the theoretical proof that SVMs with RBF kernel and suitable parameters give at least as good accuracy as linear kernel. Yuan *et al.* [Yuan et al., 2012] show empirically that LIBSVM (RBF kernel) often offers better and more stable results than LIBLINEAR [Fan et al., 2008] on many benchmark datasets. However, the training cost of LIBSVM is too high in terms of computation time. It would take many days when performing on large scale datasets like ImageNet. Therefore, speedup the training process of LIBSVM become a very essential task in the context of large scale image classification.

In the multi-core era, computers with multi-cores or multiprocessors bring to us many advantages. Advanced technologies designed for the systems where several processing cores have access to a single memory space are becoming popular choice for high performance computing systems. OpenMP Application Program Interface (API) is a multi-platform shared-memory parallel programming model working on these systems [OpenMP Architecture Review Board, 2008]. It has been proven to work effectively on shared memory systems by the Board of OpenMP Architecture Review Board, 2008. Therefore, it motivates us to investigate parallel algorithms and demonstrate how LIBSVM can benefit from these modern platforms. In the original implementation of LIBSVM, computing kernel values in the matrices of various formulations is a very time-consuming step, especially when performing on datasets with a very large number of instances. Fortunately, the values in these matrices can be computed independently allowing to apply parallel algorithms. In this paper, we use OpenMP to parallelize this computation on a multi-core computer. With this modification, we significantly reduce the training time of LIBSVM.

To evaluate the performance of pLIBSVM, we compare it with LIBLINEAR and OCAS [Franc and Sonnenburg, 2008]. Franc *et al.* [Franc and Sonnenburg, 2008] have shown in their experiments that OCAS even in the early optimization steps shows often faster convergence than the so far in this domain prevailing approximative methods. So we chose OCAS to compare with our pLIBSVM instead of Pegasos (Primal Estimated sub-GrAdient SOlver for SVM) or SGD SVM (Stochastic Gradient Descent SVM).

6 Experiments and Results

6.1 Datasets

The full size of ImageNet dataset is 1TB. In the experiments, we evaluated our framework on 10 largest classes that contains 24,807 images with data size 2.4GB. The specific names of these classes are n00483313, n01882714, n02086240, n02087394, n02094433, n02100583, n02100735, n02138441, n02279972, n09428293. There are more than 2000 diversified images per class. In each class, we sample 90% of images for training and 10% of images for testing.

6.2 Parallel Feature Extraction

We perform our experiments on an Intel(R) Xeon(R) CPU E5345, 2.33GHz computer. Depending on parameters setting, the computation time of extracting feature (e.g. SIFT) of an image ranges from 0.46 to 1 second (single thread is used in computation). To process the 10 largest classes, it would take from 3 to 7 hours. Therefore, it is very difficult to scaleup to full ImageNet because if it takes 1 second per image for feature extraction then we need 14M $\times$ 1 second $\simeq$ 162 days. To deal with this challenge, we apply parallel solutions to reduce the computation time.

SIFT/DSIFT: VLFeat, a free version for extracting SIFTs, can be downloaded from the author's homepage (www.vlfeat.org). It has been developed by Andrea Vedaldi from the Vision Lab of the University of California. The original implementation of SIFT descriptors are integer vectors in 128 dimensions. There is no interdependence in feature extraction tasks, so we can extract features in parallel way. In this experiments, we use 8 CPU cores on our computer to extract features. As shown in Table 1, we need 56 minutes to extract more than 639M DSIFTs from the 10 largest classes. That means it takes 0.14 second to extract DSIFTs from an image on average. Therefore, with full ImageNet dataset, extracting DSIFTs would take 0.14s $\times$ 14M $\simeq$ 22 days.

Parallel SURF: Parallel SURF is a fast parallel version of SURF maintained by David Gossow [Gossow et al., 2010]. The local image descriptors extracted from original implementation of Parallel SURF are floating vectors in 64 dimensions. In this experiment, we also use 8 CPU cores for extracting feature. As shown in Table 1, we need 54 minutes to extract more than 47M SURFs from the 10 largest classes. That means it takes 0.13 second to extract SURFs from an image on average.

So, with full dataset, extracting SURFs will take $0.13s \times 14M \simeq 21$ days. Obviously, we can speedup the process of extracting features by using more resources (CPU cores, computer, etc.).

Table 1 Extract features from the 10 largest classes ImageNet using 8 CPU cores

Features	Time	# keypoints	Size
SIFT	17m46s	18,923,756	6GB
SURF	54m	47,308,685	24.4GB
DSIFT	56m	639,904,650	201.3GB

6.3 Fast Codebook Building

In BoW model, one of the steps that takes a long time is to build codebook. With a large scale dataset we need to get a large amount of datapoints to build a discriminative codebook, so this task becomes very large in terms of time complexity. One of the popular choices to build codebook is the k-means clustering algorithm. However, the original implementation of k-means takes many days to converge when performing on a large scale dataset like ImageNet. So reducing the execution time for this task is becoming an essential task when we study an efficient framework for large scale image classification. In this experiment, we have used the parallel version of k-means from Wei Dong [Dong,]. This program is a re-implementation of the k-means clustering algorithm. It has the following features:

1. An out-of-core k-means that allows clustering data larger than main memory,
2. Support parallel reading from multiple input files,
3. Accelerate L2 distance calculation with BLAS or KD-tree.

To perform the k-means algorithm, we use 8 CPU cores on the same computer as in section 6.2. We sample all visual descriptors of the images in training dataset to build codebooks with 5,000 codewords. We set the maximum iteration of the k-means to 40 and the convergence threshold to 0.001. By using parallel k-means, we build codebooks from very large datasets in reasonable time, as shown in Table 2.

Table 2 Parallel k-means on the 10 largest classes ImageNet using 8 CPU cores

Features	# datapoints	Dimension	Size	Time
SIFT	17,032,522	128	5.4GB	5h21m
SURF	42,610,816	64	22GB	4h03m
DSIFT	575,790,745	128	181.2GB	8 days

6.4 Parallel Bag-of-Packets Constructing

To speedup the construction of BoW histograms of images, we take into account the implementation of randomized kd-tree forests from VLFeat toolbox. It not only improves the effectiveness of the representation in high dimensions, but enables fast medium and large scale nearest neighbor queries among high dimensional data points. Once k-means is performed, we build a hierarchical structure for codebooks by using *vl_kdtreebuild*. By this way, we can use *vl_kdtreequery* to speedup the process of mapping visual descriptors to visual words (or codewords). The computation time for applying each codebook is similar to classical approach (single codebook). When we use n codebooks for constructing BoP of images, it means we need n more times to finish this process. To achieve the same computation time as single codebook approach, we perform the process of constructing BoP in a parallel way. Consequently, the whole computation time of this process is the same as the largest individual standard approach. As shown in Table 3 and 4, we can reduce the computation time for constructing BoP of DSIFT+SURF+SIFT with multi-codebooks to the same amount of time that the one of DSIFT with a single codebook.

Table 3 Parallelize bag-of-packets construction using 8 CPU cores. The image representation is normalized by L2-Norm.

Features	Dimension	Time	Size
SIFT	5,000	3m18s	179MB
SURF	5,000	7m48s	320MB
DSIFT	5,000	1h01s	934MB
DSIFT+SURF	10,000	1h02s	1.2GB
DSIFT+SURF+SIFT	15,000	1h05s	1.5GB

Table 4 Parallelize bag-of-packets construction using 8 CPU cores. The image representation is converted to high-dimensional space by using homogeneous kernel map.

Features	Dimension	Time	Size
SIFT	15,000	3m06s	560MB
SURF	15,000	7m02s	1GB
DSIFT	15,000	58m03s	2.9GB
DSIFT+SURF	30,000	59m01s	4GB
DSIFT+SURF+SIFT	45,000	1h05s	4.5GB

6.5 Classification Accuracy

The linear kernel on the classical histogram based feature gives very poor accuracy on image classification. Therefore, once BoW histogram is constructed, some recent image classification systems use feature map to convert BoW histogram from linear space to non-linear space. This step is useful when one want to stick to the efficient linear classifiers [Chatfield et al., 2011]. The result is the image representation in high-dimensional space, that ensures linear separability of the classes. Notice that before training classifiers, we should normalize BoW histograms, so that the image size does not influence histogram counts. The popular normalization methods used in recent image classification systems are L1-Norm and L2-Norm:

$$L1-norm: f(x) = \frac{x}{||x||_1} = \frac{x}{\sum_{i=1}^{N} |x_i|} \tag{2}$$

$$L2-Norm: f(x) = \frac{x}{||x||_2} = \frac{x}{\sqrt{\sum_{i=1}^{N} |x_i|^2}} \tag{3}$$

In the experiments, we want to evaluate our approach in two different cases. In the first case, we use L2-Norm to normalize BoW histograms of all images in dataset, as shown in Table 5 and 6. In the second one, we use L1-Norm to normalize BoW histograms and then the final image representation is converted to high-dimensional space by using homogeneous kernel map from Vedaldi, as shown in Table 7 and 8. By using this feature map, we obtain a significant improvement in image classification accuracy (from +6.24% to +16.93% with different feature types).

Multi-feature and multi-codebook. To evaluate the performance of multi-feature and multi-codebook approach on the ten largest classes from ImageNet, we perform the experiments for each single feature SIFT, SURF and DSIFT. Then we perform classification by using simultaneously different feature types DSIFT+SURF and DSIFT+SURF+SIFT. As shown in Fig. 4, in the case of training LIBSVM (RBF kernel) on the combination of three different feature types, we significantly improve the performance of overall classification accuracy up to 1.82 times, compared to single feature SIFT (Table 5). The picture of the improvements is the same when we use homogeneous kernel map (Table 7 and 8).

Parallel LIBSVM. To evaluate the performance of our pLIBSVM, we compare it with LIBLINEAR, OCAS, and the original implementation of LIBSVM (a non-parallel version) in terms of both classification accuracy and training time. In the experiments, we use 8 CPU cores on the same computer as in section 6.2. We also evaluate our implementation with different SVM (linear kernel and RBF kernel). In the case of training pLIBSVM with RBF kernel, we use cross validation on training data to find the best parameters C and gamma of SVM classifiers.

As aforementioned, the major challenge of large scale image classification is on training classifiers. This paper proposed a parallel version of LIBSVM that was efficient on the ten largest classes of ImageNet. As shown in Fig. 5, in the case of

Table 5 Overall classification accuracy and training time. The image representation is normalized by L2-Norm. Training LIBSVM with linear kernel.

Features	LIBLINEAR	OCAS	LIBSVM	pLIBSVM
SIFT	34.91% (3m05s)	38.94% (28m39s)	**46.31%** (45m55s)	**46.31%** (8m57s)
SURF	35.79% (3m34s)	44.7% (43m40s)	**50.83%** (1h13m)	**50.83%** (13m36s)
DSIFT	52.64% (11m52s)	59.09% (1h23m)	**64.57%** (1h50m)	**64.57%** (17m20s)
DSIFT+SURF	60.42% (15m06s)	64.05% (1h52m)	**67.63%** (3h26m)	**67.63%** (31m46s)
DSIFT+SURF+SIFT	63.80% (23m03s)	65.70% (4h45m)	**70.09%** (4h58m)	**70.09%** (43m27s)

Table 6 Overall classification accuracy and training time. The image representation is normalized by L2-Norm. Training LIBSVM with RBF kernel.

Features	LIBSVM	pLIBSVM	Accuracy
SIFT	1h01m	7m33s	50.42%
SURF	2h00m	15m31s	56.47%
DSIFT	3h01m	19m02s	68.36%
DSIFT+SURF	5h48m	37m10s	70.25%
DSIFT+SURF+SIFT	6h03m	47m44s	**71.46%**

Table 7 Overall classification accuracy and training time. The image representation is converted to high-dimensional space by using homogeneous kernel map. Training LIBSVM with linear kernel.

Features	LIBLINEAR	OCAS	LIBSVM	pLIBSVM
SIFT	41.15% (3m58s)	43.62% (55m31s)	**47.52%** (1h21m)	**47.52%** (16m41s)
SURF	47.84% (6m30s)	50.63% (1h23m)	**55.62%** (2h17m)	**55.62%** (25m19s)
DSIFT	69.57% (15m05s)	72.07% (3h07m)	**74.32%** (3h42m)	**74.32%** (43m44s)
DSIFT+SURF	71.22% (49m59s)	72.72% (6h53m)	**75.17%** (5h15m)	**75.17%** (1h08m)
DSIFT+SURF+SIFT	72.95% (1h30m)	73.97% (20h39m)	**75.98%** (5h50m)	**75.98%** (1h17m)

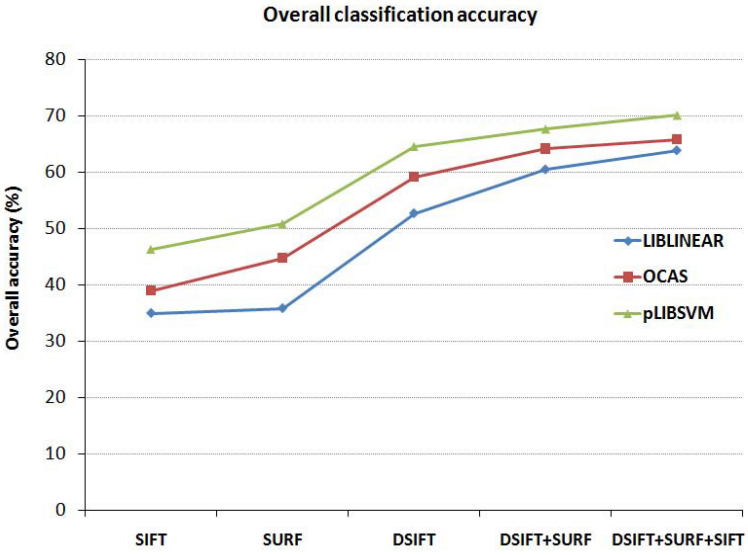


Fig. 4 Overall accuracy of SVM classifiers with different feature types

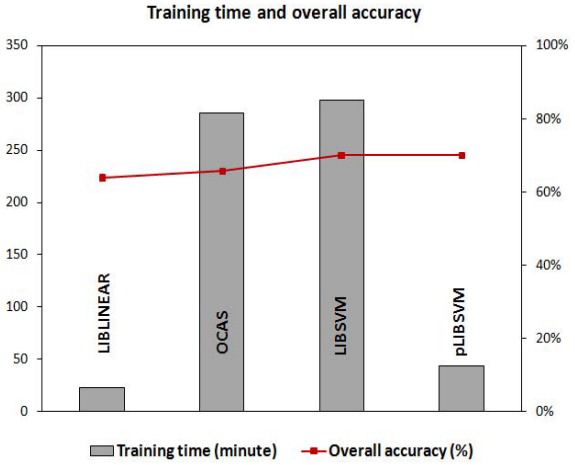


Fig. 5 Overall accuracy and training time of SVM classifiers

the combination of three different feature types (DSIFT+SURF+SIFT), the accuracy performance of pLIBSVM and LIBSVM are higher than the other classifiers from +4.39% to +6.29% (Table 5). Table 6 to 8 also show a high performance of pLIBSVM and LIBSVM on all different feature types. Furthermore, in the case of

training SVM classifiers with RBF kernel, pLIBSVM achieves better results than those of linear kernel classifiers, as shown in Table 6 and 8.

About the training time, all the results presented in Table 5 to 8 are user time. We have chosen this instead of cpu time because cpu time is the sum of all threads cpu time when we use multi-threading, so we could not show the improvement of our approach this way. We know the drawback of user time: we cannot be sure we are the only user during the whole execution process. As shown in Fig. 5, we can see we significantly speedup the training time of classifiers with our approach. Particularly, our pLIBSVM use 43 minutes with 8 CPU cores to train classifiers, compared to 4 hours 58 minutes of LIBSVM and 4 hours 45 minutes of OCAS (Table 5). The accuracy of all the linear kernel classifiers is increased when we transform the data in a high-dimensional space by using homogeneous kernel map (Table 7), but the training time is increased too, due to the higher dimension of the input space. Finally the best result is obtained by using the same transformation and RBF kernel with LIBSVM/pLIBSVM as shown in Table 8. The time reported here is with only 8 CPU cores and the 10 largest classes from ImageNet, of course the training time can easily be reduced by using more resources (CPUs, cores, computers). This is what we plan to do for the 1000 largest classes of the same dataset.

Table 8 Overall classification accuracy and training time. The image representation is converted to high-dimensional space by using homogeneous kernel map. Training LIBSVM with RBF kernel.

Features	LIBSVM	pLIBSVM	Accuracy
SIFT	3h17m	19m22s	51.47%
SURF	4h54m	29m51s	58.57%
DSIFT	7h31m	53m11s	76.86%
DSIFT+SURF	8h51m	1h19m	77.03%
DSIFT+SURF+SIFT	9h50m	1h33m	**78.15%**

Among the different GPU-based approaches, the only one that could be used for very large datasets is the incremental SVM with CUDA [Poulet and Pham, 2010]. This method is 3 to 4 orders of magnitude faster than usual classification algorithms like libSVM, but it is only a linear kernel so we know the accuracy will be less than the one with RBF kernel. To the best of our knowledge, the other GPU-based SVMs with non linear kernel require to load the whole dataset into main memory. Most of the GPU architectures today have up to 6 GB memory size. Here with only 5k vocabulary size and the 10 largest classes from ImageNet, we already need 4.5GB for the data and the 1000 largest classes from the same dataset require 25GB memory so no GPU-based SVM can be used in this context.

7 Conclusion and Future Work

We have proposed a fast and efficient framework for large scale image classification and show how to address this challenge by using ImageNet dataset as an example. In this framework, we have developed a parallel version of LIBSVM to efficiently deal with very large datasets in reasonable time. To speedup the process of extracting features, we have presented how to use a multi-core computer to reduce the computation time of feature extraction. We have also presented a novel approach using several different local features simultaneously to improve the classification accuracy on a large scale image dataset (the relative increase is up to 82%). In the near future, we plan to study how to combine effectively the global features (e.g. contour, texture, shape, etc.) with the local features to get more discriminative power of image representation. About the computation time our approach allows us to get the classification results with a RBF kernel in almost the same time as with usual algorithms and linear kernel by using only 8 cores. The next step is the classification of the 1000 largest classes of ImageNet (more than 1.4 million images). Furthermore, the current version of libOCAS only offer parallel version of the binary solver, so we intend to parallelize it for multi-class problem. That will be a promising research for large scale image classification. Encoding spatial information of the interesting keypoints of image will be also studied.

Acknowledgements. This work was partially funded by Region Bretagne (France).

References

[Bay et al., 2008] Bay, H., Ess, A., Tuytelaars, T., Gool, L.J.V.: Speeded-up robust features (surf). Computer Vision and Image Understanding 110(3), 346–359 (2008)

[Bosch et al., 2007] Bosch, A., Zisserman, A., Muñoz, X.: Image classification using random forests and ferns. In: International Conference on Computer Vision, pp. 1–8 (2007)

[Chang and Lin, 2001] Chang, C.C., Lin, C.J.: LIBSVM – a library for support vector machines (2001), `http://www.csie.ntu.edu.tw/~cjlin/libsvm`

[Chatfield et al., 2011] Chatfield, K., Lempitsky, V., Vedaldi, A., Zisserman, A.: The devil is in the details: an evaluation of recent feature encoding methods. In: British Machine Vision Conference, pp. 76.1–76.12 (2011)

[Csurka et al., 2004] Csurka, G., Dance, C.R., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: Workshop on Statistical Learning in Computer Vision, ECCV, pp. 1–22 (2004)

[Dalal and Triggs, 2005] Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 886–893. IEEE Computer Society (2005)

[Deng et al., 2010] Deng, J., Berg, A.C., Li, K., Fei-Fei, L.: What does classifying more than 10, 000 image categories tell us? In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 71–84. Springer, Heidelberg (2010)

[Deng et al., 2009] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Li, F.-F.: Imagenet: A large-scale hierarchical image database. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 248–255 (2009)

[Dong,] Dong, W.: A parallel out-of-core k-means clusterer, http://www.cs.princeton.edu/~wdong/kmeans

[Everingham et al., 2010] Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision 88(2), 303–338 (2010)

[Fan et al., 2008] Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., Lin, C.-J.: Liblinear: A library for large linear classification. Journal of Machine Learning Research 9, 1871–1874 (2008)

[Fellbaum, 1998] Fellbaum, C.: WordNet: An Electronic Lexical Database. Bradford Books (1998)

[Fergus et al., 2009] Fergus, R., Weiss, Y., Torralba, A.: Semi-supervised learning in gigantic image collections. In: Advances in Neural Information Processing Systems, pp. 522–530 (2009)

[Franc and Sonnenburg, 2008] Franc, V., Sonnenburg, S.: Optimized cutting plane algorithm for support vector machines. In: International Conference on Machine Learning, pp. 320–327 (2008)

[Gossow et al., 2010] Gossow, D., Decker, P., Paulus, D.: An evaluation of open source surf implementations. In: Ruiz-del-Solar, J. (ed.) RoboCup 2010. LNCS (LNAI), vol. 6556, pp. 169–179. Springer, Heidelberg (2010)

[Griffin et al., 2007] Griffin, G., Holub, A., Perona, P.: Caltech-256 Object Category Dataset. Technical Report CNS-TR-2007-001, California Institute of Technology (2007)

[Keerthi and Lin, 2003] Keerthi, S.S., Lin, C.-J.: Asymptotic behaviors of support vector machines with gaussian kernel. Neural Computation 15(7), 1667–1689 (2003)

[Lazebnik et al., 2006] Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2169–2178 (2006)

[Li et al., 2007] Li, F.-F., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. Computer Vision and Image Understanding 106(1), 59–70 (2007)

[Li et al., 2009] Li, Y., Crandall, D.J., Huttenlocher, D.P.: Landmark classification in large-scale image collections. In: IEEE 12th International Conference on Computer Vision, pp. 1957–1964. IEEE (2009)

[Lowe, 2004] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)

[Moosmann et al., 2006] Moosmann, F., Triggs, B., Jurie, F.: Fast discriminative visual codebooks using randomized clustering forests. In: Advances in Neural Information Processing Systems, pp. 985–992 (2006)

[OpenMP Architecture Review Board, 2008] OpenMP Architecture Review Board. OpenMP application program interface version 3.0 (2008)

[Perronnin et al., 2010] Perronnin, F., Sánchez, J., Liu, Y.: Large-scale image categorization with explicit data embedding. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2297–2304 (2010)

[Philbin et al., 2008] Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: IEEE Conference on Computer Vision and Pattern Recognition (2008)

[Poulet and Pham, 2010] Poulet, F., Pham, N.-K.: High dimensional image categorization. In: Cao, L., Feng, Y., Zhong, J. (eds.) ADMA 2010, Part I. LNCS, vol. 6440, pp. 465–476. Springer, Heidelberg (2010)

[Tola et al., 2010] Tola, E., Lepetit, V., Fua, P.: Daisy: An efficient dense descriptor applied to wide-baseline stereo. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(5), 815–830 (2010)

[Torralba et al., 2008] Torralba, A., Fergus, R., Freeman, W.T.: 80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(11), 1958–1970 (2008)

[Vedaldi et al., 2009] Vedaldi, A., Gulshan, V., Varma, M., Zisserman, A.: Multiple kernels for object detection. In: IEEE 12th International Conference on Computer Vision, pp. 606–613. IEEE (2009)

[Vedaldi and Zisserman, 2012] Vedaldi, A., Zisserman, A.: Efficient additive kernels via explicit feature maps. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(3), 480–492 (2012)

[Wang et al., 2009] Wang, C., Yan, S., Zhang, H.-J.: Large scale natural image classification by sparsity exploration. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 3709–3712. IEEE (2009)

[Winder and Brown, 2007] Winder, S.A.J., Brown, M.: Learning local image descriptors. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2007)

[Yuan et al., 2012] Yuan, G.-X., Ho, C.-H., Lin, C.-J.: Recent advances of large-scale linear classification. Proceedings of the IEEE 100(9), 2584–2603 (2012)

About the Editors

Fabrice Guillet is a CS professor at Polytech'Nantes, the graduate engineering school of University of Nantes, and a member of the "KnOwledge and Decision" team (COD) of the LINA laboratory. He received a PhD degree in CS in 1995 from the "École Nationale Supérieure des Télécommunications de Bretagne", and his Habilitation (HdR) in 2006 from Nantes university. He is a co-founder of the International French-speaking "Extraction et Gestion des Connaissances (EGC)" society. His research interests include knowledge quality and knowledge visualization in the frameworks of Data Mining and Knowledge Management. He has recently co-edited two refereed books of chapter entitled "Quality Measures in Data Mining" and "Statistical Implicative Analysis — Theory and Applications" published by Springer in 2007 and 2008.

Bruno Pinaud received the PhD degree in Computer Science in 2006 from the University of Nantes. He is currently assistant professor at the University of Bordeaux in the Computer Science Department since September 2008. His current research interests are visual data mining, graph rewriting systems, graph visualization and experimental evaluation in HCI (Human Computer Interaction). He successfully organized the 2012 edition of the EGC Conference.

Gilles Venturini is a CS Professor at François Rabelais University of Tours (France). His main researches interests concern visual data mining, virtual reality, 3D acquisition, biomimetic algorithms (genetic algorithms, artificial ants). He is co-editor in chief of the French New IT Journal (Revue des Nouvelles Technologies de l'Information) and was recently elected as President of the EGC society.

Djamel Abdelkader Zighed is a CS Professor at the Lyon 2 University. He is the head of the Human Sciences Institute and he was Director of the ERIC Laboratory (University of Lyon). He is also the coordinator of the Erasmus Mundus Master Program on Data Mining and Knowledge Management (DMKM). He is also member of various international and national program committees.

Author Index