Advances in Intelligent Systems and Computing 529

Plamen Angelov
Yannis Manolopoulos
Lazaros Iliadis
Asim Roy
Marley Vellasco *Editors*

Advances in Big Data

Proceedings of the 2nd INNS Conference on Big Data, October 23–25, 2016, Thessaloniki, Greece

Springer

Advances in Intelligent Systems and Computing

Volume 529

Series editor

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

About this Series

The series “Advances in Intelligent Systems and Computing” contains publications on theory, applications, and design methods of Intelligent Systems and Intelligent Computing. Virtually all disciplines such as engineering, natural sciences, computer and information science, ICT, economics, business, e-commerce, environment, healthcare, life science are covered. The list of topics spans all the areas of modern intelligent systems and computing.

The publications within “Advances in Intelligent Systems and Computing” are primarily textbooks and proceedings of important conferences, symposia and congresses. They cover significant recent developments in the field, both of a foundational and applicable character. An important characteristic feature of the series is the short publication time and world-wide distribution. This permits a rapid and broad dissemination of research results.

More information about this series at http://www.springer.com/series/11156

Plamen Angelov · Yannis Manolopoulos
Lazaros Iliadis · Asim Roy
Marley Vellasco
Editors

Advances in Big Data

Proceedings of the 2nd INNS
Conference on Big Data, October 23–25, 2016
Thessaloniki, Greece

Springer

Editors

Plamen Angelov
School of Computing and Communications
Lancaster University
Lancaster
UK

Yannis Manolopoulos
Data Engineering Lab, Department of Informatics
Aristotle University of Thessaloniki
Thessaloniki
Greece

Lazaros Iliadis
Lab of Forest Informatics (FiLAB)
Democritus University of Thrace
Orestiada
Greece

Asim Roy
WPC Information Systems Faculty
Arizona State University
Tempe, AZ
USA

Marley Vellasco
Electrical Engineering Department, (ICA)
Pontifical Catholic University of Rio de Janeiro
Rio de Janeiro, RJ
Brazil

ISSN 2194-5357 ISSN 2194-5365 (electronic)
Advances in Intelligent Systems and Computing
ISBN 978-3-319-47897-5 ISBN 978-3-319-47898-2 (eBook)
DOI 10.1007/978-3-319-47898-2

Library of Congress Control Number: 2016953664

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

The concept "Big data" is not only related to storage of and access to data. Analytics play a major role in making sense of that data and exploiting its value. Big data analytics is about examining vast volumes of data in order to discover hidden patterns or even existing correlations. With available technology, it is feasible to analyze data and get crucial answers from it, almost in real time.

A lot of research has been made by Google on this field with impressive results. It is worth mentioning "Google Cloud Machine Learning", a managed platform enabling easy construction and development of machine learning models, capable to perform on any type of data, of any size. The neural network field has historically focused on algorithms that learn in an online, incremental mode without requiring in-memory access to huge amounts of data. This type of learning is not only ideal for streaming data (as in the Industrial Internet or the Internet of Things) but could also be used on stored big data. Neural network technologies thus can become significant components of big data analytics platforms.

One of the major challenges of our era is learning from big data, something that requires the development of novel algorithmic approaches. Most of the available machine learning algorithms, find it hard to scale up to big data. Moreover there are serious challenges in the problems of high-dimensionality, velocity and variety. Thus the aim of this conference is to promote new advances and research directions in efficient and innovative algorithmic approaches to analyzing big data (e.g. deep networks, nature-inspired and brain-inspired algorithms), implementations on different computing platforms (e.g. neuromorphic, GPUs, clouds, clusters) and applications of Big Data Analytics to solve real-world problems (e.g. weather prediction, transportation, energy management).

The 2nd INNS Big Data 2016 conference, founded by the International Neural Network Society (INNS), was organized by Aristotle University of Thessaloniki Greece and Democritus University of Thrace Greece, following the inaugural event held in San Francisco in 2015. This conference aims to initiate the collaborative adventure with big data and other learning technologies.

All submitted papers in the 2nd INNS Big Data conference have passed through a peer review process by at least 2 independent academic referees. Where needed a third and a fourth referee was consulted to resolve any potential conflicts. Out of 50 submissions, totally 34 submissions have been accepted for oral presentation; 27 papers have been accepted as full ones (54 %) and 7 as short ones. Authors come from 19 different countries around the globe, namely: Brazil, China, Cyprus, Denmark, France, Germany, Greece, India, Italy, Japan, Malaysia, Portugal, Saudi Arabia, Spain, Tunisia, Turkey, USA, Ukraine and UK. Proceedings are published by Springer in the “Advances in Intelligent Systems and Computing Series”.

The program of the 2nd INNS Big Data Conference includes 5 plenary talks by distinguished keynote speakers, plus 2 tutorials.

David Bholat is Senior Analyst with the Bank of England. In particular, Dr. Bholat leads a team of ten data scientists and researchers in Advanced Analytics, a Big Data division in the Bank of England which he helped to establish in 2014. The division is recognised as a leader among central banks in the area of Big Data, as noted in a recent MIT Sloan Review article profiling the division. A former Fulbright fellow, Dr. Bholat graduated from Georgetown University’s School of Foreign Service with highest honours. He subsequently studied at the London School of Economics, the University of Chicago and London Business School. Publications in 2016 include Modelling metadata in central banks; Non-performing loans: regulatory and accounting treatments of assets; Peer-to-peer lending and financial innovation in the United Kingdom; and Accounting in central banks. Other previous publications relevant to the conference include Text mining for central banks; Big data and central banks; and The future of central bank data.

The title of the talk by Dr. Bholat is “Big Data at the Bank of England”. This talk will discuss the Bank of England’s recent forays into Big Data and other unconventional data sources. Particular attention will be given to the practicalities of embedding data analytics in a business context. Examples of the Bank’s use of Big Data will be given, including the analysis of derivatives and mortgage data, Internet searches, and social media.

Francesco Bonchi is Research Leader with the ISI Foundation of Turin and Scientific Director of the Technological Center of Catalunya at Barcelona. Before he was Director of Research at Yahoo Labs in Barcelona, Spain, where he was leading the Web Mining Research group. His recent research interests include mining query-logs, social networks, and social media, as well as the privacy issues related to mining these kinds of sensible data. In the past he has been interested in data mining query languages, constrained pattern mining, mining spatiotemporal and mobility data, and privacy preserving data mining. He will be PC Chair of the 16th IEEE International Conference on Data Mining (ICDM 2016) to be held in Barcelona in December 2016. He is member of the ECML/PKDD Steering Committee, Associate Editor of the IEEE Transactions on Big Data (TBD), the IEEE Transactions on Knowledge & Data Engineering (TKDE), the ACM Transactions on Intelligent Systems and Technology (TIST), Knowledge and Information Systems (KAIS), and member of the Editorial Board of Data Mining & Knowledge Discovery (DMKD). He has been Program Co-chair of the ECML/PKDD’2010. He is co-editor of the book “Privacy-Aware Knowledge Discovery: Novel Applications and New Techniques” published by Chapman & Hall/CRC Press. He earned his Ph.D. in Computer Science from the University of Pisa in December 2003.

The title of the talk by Dr. Bonchi is “On information propagation, social influence, and communities”. With the success of online social networks and microblogging platforms such as Facebook, Tumblr, and Twitter, the phenomenon of influence-driven propagations, has recently attracted the interest of computer scientists, sociologists, information technologists, and marketing specialists. In this talk we will take a data mining perspective, discussing what (and how) can be learned from a social network and a database of traces of past propagations over the social network. Starting from one of the key problems in this area, i.e. the identification of influential users, we will provide a brief overview of our recent contributions in this area. We will expose the connection between the phenomenon of information propagation and the existence of communities in social network, and we will go deeper in this new research topic arising at the overlap of information propagation analysis and community detection.

Steve Furber, CBE FRS FREng, is ICL Professor of Computer Engineering in the School of Computer Science at the University of Manchester, UK. After completing a BA in Mathematics and a PhD in Aerodynamics at the University of Cambridge, UK, he spent the 1980s at Acorn Computers, where he was a principal designer of the BBC Microcomputer and the ARM 32-bit RISC microprocessor. Over 75 billion variants of the ARM processor have since been manufactured, powering much of the world’s mobile and embedded computing. He moved to the ICL Chair at Manchester in 1990 where he leads research into asynchronous and low-power systems and, more recently, neural systems engineering, where the SpiNNaker project is delivering a computer incorporating a million ARM processors optimised for brain modelling applications.

The title of the talk by Professor Furber is “The SpiNNaker Project”. The SpiNNaker (Spiking Neural Network Architecture) project aims to produce a massively-parallel computer capable of modelling large-scale neural networks in biological real time. The machine has been 15 years in conception and ten years in construction, and has far delivered a 100,000-core machine in a single 19-inch rack, which is now being expanded towards the million-core full system. Although primarily intended as a platform to support research into information processing in the brain, SpiNNaker has also proved useful for Deep Networks and similar applied Big Data applications. In this talk I will present an overview of the machine and the design principles that went into its development, and I will indicate the sort of applications for which it is proving useful.

Rudolf Kruse is Professor at the Otto-von-Guericke University of Magdeburg (Germany), where he is leading the Computational Intelligence Group. His current research interests include data science and intelligent systems. His group is successful in various industrial applications in cooperation with companies such as Volkswagen, SAP, Daimler, and British Telecom. He obtained his Ph.D. and his Habilitation in Mathematics from the Technical University of Braunschweig in 1980 and 1984 respectively. Following a stay at the Fraunhofer Gesellschaft, he joined the

Technical University of Braunschweig as a professor of computer science in 1986. Since 1996 he is a full professor at the Department of Computer Science of the Otto-Von-Guericke University Magdeburg in Germany. Rudolf Kruse has coauthored 15 monographs and 25 books as well as more than 350 refereed technical papers in various scientific areas. He is associate editor of several scientific journals. He is a Fellow of the International Fuzzy Systems Association (IFSA), Fellow of the European Coordinating Committee for Artificial Intelligence (ECCAI) and Fellow of the Institute of Electrical and Electronics Engineers (IEEE).

The title of the talk by Professor Kruse is “Modeling Self-Explanatory Big Data Applications”. Big Data and Data Science have made commercial advances driven by research. Typical questions that arise during the modeling phase are whether it should be aimed to provide explanations to the user, what should be explained and how this should be done. This talk addresses these controversies using two exemplary industrial projects: “Markov Networks for Planning” and “Medical Research Insights”.

Piotr Mirowski is a Research Scientist at Google DeepMind, the research lab that focuses on solving Artificial General Intelligence and that investigates research directions such as deep reinforcement learning or systems neuroscience. Piotr has a M.Sc. in computer science (2002) from ENSEEIHT in Toulouse, France and, prior to resuming his studies, worked as a research engineer at Schlumberger Research (2002–2005). He obtained his Ph.D. in computer science (2011) at New York University under the supervision of Prof. Yann LeCun. His doctoral work on using recurrent neural networks for learning representations of time series covered applications such as inferring gene regulation networks, predicting epileptic seizures, categorizing streams of online news and statistical language modeling for speech recognition (in collaboration with AT&T Labs Research). After his PhD, Piotr was a Member of Technical Staff at Bell Labs (2011–2013), where he focused on simultaneous localization and mapping for robotics, on indoor localization and on load forecasting for smart grids. Prior to joining Google DeepMind (2014 till now), Piotr worked as an applied scientist at Microsoft Bing (2013–2014), investigating deep learning methods for search query formulation.

The title of the talk by Dr. Mirowski is “Learning Sequences”. Many data science or control problems can be qualified as sequence learning. Examples of sequences abound in fields such as natural language processing (e.g., speech recognition, machine translation, query formulation or image caption generation) or robotics (control and navigation). Their underlying challenge resides in learning long range memory of observed data. In this talk, we will look at the inner workings of recurrent neural networks and neural memory architectures for learning sequence representation and illustrate their state-of-the-art performance.

Professors Luca Oneto and Dr. Davide Anguita, of the University of Genoa, will deliver a tutorial titled “Model Selection and Error estimation without the Agonizing Pain”. Some Big Data failures, like the infamous 2013 Google Flu Trends misprediction, reveal that large volumes of data are not enough for building effective and reliable predictive models. Even when huge datasets are available, Data Science needs Statistics in order to cope with the selection of optimal models

(Model Selection) and the estimation of their quality (Error Estimation). In particular, Statistical Learning Theory (SLT) addresses these problems by deriving non-asymptotic bounds on the generalization error of a model or, in other words, by upper bounding the true error of the learned model based just on quantities computed on the available data. However, for a long time, SLT has been considered only an abstract theoretical framework, useful for inspiring new learning approaches, but with limited applicability to practical problems. The purpose of this tutorial is to give an intelligible overview of the problems of Model Selection and Error Estimation, by focusing on the ideas behind the different SLT-based approaches and simplifying most of the technical aspects with the purpose of making them more accessible and usable in practice, with particular reference to Big Data problems. We will start by presenting the seminal works of the 80's until the most recent results, then discuss open problems and finally outline future directions of this field of research.

Professor Giacomo Boracchi, of the Politecnico di Milano, will deliver a tutorial entitled "Change Detection in Data Streams: Big Data Challenges". Changes might indicate unforeseen evolution of the process generating the data, anomalous events, or faults, to name a few examples. As such, change-detection tests provide precious information for understanding the stream dynamics and activating suitable actions, which are two primary concerns in financial analysis, quality inspection, environmental and health-monitoring systems. Change detection plays also a central role in machine learning, being often the first step towards adaptation. This tutorial presents a formal description of the change-detection problem that fits sequential monitoring as well as classification and signal/image analysis applications. The main approaches in the literature are then presented, discussing their effectiveness in big data scenarios, where either data-throughput or data-dimension are large. In particular, change-detection tests for monitoring multivariate data streams will be presented in detail, including the popular approach of monitoring the log-likelihood, which will be demonstrated to suffer from detectability loss when data-dimension increases. The tutorial is accompanied by various examples where change-detection methods are applied to real world problems, including classification of streaming data, detection of anomalous heartbeats in ECG tracings and the localization of anomalous patterns in images for quality control.

Finally, we would like to thank the Artificial Intelligence Journal (Elsevier) for sponsoring the conference.

We hope that the 2nd INNS Big Data conference will stimulate the international Big Data community and that it will provide insights in opening new paths in Analytics by conducting further algorithmic and applied research.

September 2016

Lazaros Iliadis
Asim Roy
Marley Vellasco

Organization

General Chairs

Plamen Angelov	Lancaster University, UK
Yannis Manolopoulos	Aristotle University of Thessaloniki, Greece

Program Committee Co-chairs

Lazaros Iliadis	Democritus University of Thrace, Greece
Asim Roy	Arizona State University, Tempe USA
Marley Vellasco	PUC-Rio, Rio de Janeiro, Brazil

Advisory Board

Nikola Kasabov	Auckland University of Technology, New Zealand
Ali Minai	University of Cincinnati, USA
Danil Prokhorov	Toyota Tech Center, Michigan, USA
Theodore Trafalis	University of Oklahoma, USA
G. Kumar Venayagamoorthy	Clemson University, USA

Tutorials/Workshop Chairs

Apostolos Papadopoulos	Aristotle University of Thessaloniki, Greece
Bernardete Ribeiro	Coimbra University, Portugal

Poster Session Chair

Yi Lu Murphey	University of Michigan-Dearborn, USA

Special Sessions Chairs

Irwin King	Chinese University of Hong Kong, China
Luca Oneto	University of Genoa, Italy

Panel Chair

Leonid Perlovsky	Harvard University, Boston, USA

Awards Chair

Araceli Sanchis de Miguel	Carlos III University, Spain

Competitions Chair

Adel Alimi	University of Sfax, Tunisia

Publication Chairs

Mariette Awad	American University of Beirut, Lebanon
Danilo Mandic	Imperial College, London, UK

Publicity Chairs

Jose Antonio Iglesias	Carlos III University of Madrid, Spain
Simone Scardapane	Sapienza University of Rome, Italy
Teng Teck Hou	Singapore Management University, Singapore

International Liaison Chairs

De-Shuang Huang	Tongji University, Shanghai, China
Petia Georgieva	University of Aveiro, Portugal

Local Organizing Committee

Anastasios Gounaris	Aristotle University of Thessaloniki, Greece

WebMaster

Ioannis Karydis	Ionian University, Greece

Program Committee

Sherief Abdallah	British University, UAE
Anjana Agrawal	Freelance Consultant, India
James Aimone	Sandia National Laboratories, USA
Anton Akusok	The University of Iowa, USA
Asma Aldrees	King Saud University, Saudi Arabia
Adel M. Alimi	University of Sfax, Tunisia
Abdulrahman Altahhan	Coventry University, UK
George Anastasopoulos	Democritus University of Thrace, Greece
Plamen Angelov	Lancaster University, UK
Davide Anguita	University of Genoa, Italy
Mario Antunes	University of Porto, Portugal
Hanane Azzag	Université de Paris-Nord, France
Mohamed Ben Halima	University of Sfax, Tunisia
Alberto Betella	Universitat Pompeu Fabra, Spain
Lachezar Bozhkov	Technical University Sofia, Bulgaria
Stephan Clemencon	Telecom ParisTech, France
Danilo Comminiello	Sapienza University of Rome, Italy
Joana Costa	Polytechnic Institute of Leiria, Portugal
Filip Dabek	Walter Reed National Military Medical Center, USA
Konstantinos Demertzis	Democritus University of Thrace, Greece
Dejan Dovzan	University of Ljubljana, Slovenia
Felix Effenberger	Max-Planck-Institute, Leipzig, Germany
Yuantao Fan	Halmstad University, Sweden
Leonardo Ferreira	University of Sao Paulo, Brazil
Emanuele Fumeo	University of Genoa, Italy
Diego Galar	Luleå University of Technology, Sweden
Murat Ganiz	Dogus University, Turkey
Christos Georgiadis	University of Macedonia, Greece
Petia Georgieva	University of Aveiro, Portugal
Guang-Bin Huang	Nanyang Technological University, Singapore
Jose Antonio Iglesias	Carlos III University of Madrid, Spain
Alexandros Iosifidis	Tampere University of Technology, Finland
Ming Jiang	University of Sunderland, UK
Mehmed Kantardzic	University of Louisville, USA
Zied Kechaou	ENIS, Sfax, Tunisia
Vishnu Pratap Singh Kirar	Truba Institute of Engineering & IT, Bhopal, India

Mustapha Lebbah	Université Paris 13, LIPN-CNRS, France
Spiros Likothanasis	University of Patras, Greece
Konstantinos Margaritis	University of Macedonia, Greece
John Mariani	Lancaster University, UK
Trevor Martin	University of Bristol, UK
Tshilidzi Marwala	University of Johannesburg, South Africa
Ken McGarry	University of Sunderland, UK
Yi Murphey	University of Michigan-Dearborn, USA
Slawomir Nowaczyk	Halmstad University, Sweden
Luca Oneto	University of Genoa, Italy
Seiichi Ozawa	Kobe University, Japan
Massimo Panella	Sapienza University of Rome, Italy
Antonis Papaleonidas	Democritus University of Thrace
Zhiwei Qin	India Thorsteinn
Rognvaldsson	Halmstad University, Sweden
Sreela Sasi	Gannon University, USA
Ahmet Sayar	Kocaeli University, Turkey
Simone Scardapane	Sapienza University of Rome, Italy
Tegjyot Singh Sethi	University of Louisville, USA
Rahul Kumar Sevakula	Indian Institute of Technology, Kanpur, India
Catarina Silva	Instituto Politécnico de Leiria, Portugal
Kaushik Sinha	Wichita State University, USA
Igor Skrjanc	University of Ljubljana, Slovenia
David Stockwell	Central Queensland University, Australia
Navid Tafaghodi Khajavi	University of Hawaii at Manoa, USA
Anastasios Tefas	Aristotle University of Thessaloniki, Greece
Athina Vakali	Aristotle University of Thessaloniki, Greece
Marley Vellasco	Pontifical Catholic University of Rio de Janeiro, Brazil
Paul Verschure	Universitat Pompeu Fabra, Spain
Ali Wali	University of Sfax, Tunisia
Zheng Yan	The Chinese University of Hong Kong, Hong Kong
Kun Yang	Stanford University, USA
Ligang Zhang	Central Queensland University, Australia
Li Zhang	University of Northumbria, UK
Liang Zhao	University of Sao Paulo, Brazil

Contents

Predicting Human Behavior Based on Web Search Activity: Greek Referendum of 2015

Spyros E. Polykalas[1(✉)] and George N. Prezerakos[2]

[1] Department of Digital Media and Communication, TEI of Ionian Islands, Argostoli, Greece
s.polykalas@teiion.gr

[2] Department of Electronic Computing Systems, TEI of Piraeus, Aigaleo, Greece
prezerak@teipir.gr

Abstract. The enormous volumes of data generated by web users are the basis of several research activities in a new innovative field of research: online forecasting. Online forecasting is associated with the proper computation of web users' data with the aim to arrive at accurate predictions of the future in several areas of human socio-economic activity. In this paper an algorithm is applied in order to predict the results of the Greek referendum held in 2015, using as input the data generated by users of the Google search engine. The proposed algorithm allows us to predict the results of the referendum with great accuracy. We strongly believe that due to the high internet penetration, as well as, the high usage of web search engines, the proper analysis of data generated by web search users reveals useful information about people preferences and/or future actions in several areas of human activity.

Keywords: Google Trends · Online forecasting · Predictions · Forecasting · Search engines · Human behavior

1 Introduction

Almost a decade ago, Google opened to the public the web users' preferences in relation to their searching behavior. Several researchers realized that the proper processing of the web users' search behavior may allow them to reveal useful information about the users' needs, wants, concerns and in general about their feelings and preferences (Ettredge et al. 2005).

Web users generate data almost in all web activities, such as visiting a website, buying online, sending/receiving emails and participating in social networks. In cases where the popularity of such activities is high, then there is plenty of room for researchers and companies to use these data in order to reach valuable conclusions not only for web users, but for the general population. The most indicative case of user generated data is web search, since is characterized by high popularity among web users and by an almost monopolized market structure since Google Search engine holds more than 85 % of the market (source: www.statista.com).

A recent study published by Eurostat indicates that the 59 % of Europeans use web search services to find information relevant to goods and services. As the percentages

P. Angelov et al. (eds.), *Advances in Big Data*, Advances in Intelligent Systems and Computing 529, DOI 10.1007/978-3-319-47898-2_1

of internet penetration and use of web search increase, relevant generated data regarding web search behavior, become statistically significant. Thus, forecasting based on web search data is becoming increasingly more accurate.

Within this context, the aim of this paper is to explore whether there is a correlation between the users' web search preferences during a time period before the Greek referendum, held in July 2015, and the actual results of the referendum. In particular in this paper an algorithm is applied in order to analyze the data generated by users of the Google engine, aiming to predict the actual results of the referendum.

The paper is structured as follows: Sect. 2 discusses the relevant literature while in Sect. 3 the proposed algorithm is described. In the next section, the algorithm is applied in the study case and in Sect. 5 the main findings of this paper are discussed.

2 Literature Review

Online forecasting based on users' web search data is becoming as one of the most promising fields in the research area of forecasting. Several efforts have been carried out by Google's own researchers which have attempted predictions using search term popularity in a number of areas ranging from home, automobile and retail sales to travel behavior (Bangwayo-Skeete and Skeete 2015; Artola et al. 2015; Yang et al. 2015). In (Choi and Varian 2012) forecast methods are proposed in order to predict near-term values of economic indicators such as automobile sales, unemployment claims, travel destination planning and consumer confidence. Home sales predictions have also been attempted by other researchers (Wu and Brynjolfsson 2009) while the work in (Ginsberg et al. 2009) constitutes an interesting take on predicting flu epidemics before they actually emerge. The prediction of unemployment rates is another successful exercise that has been attempted both before the establishment of Google Trends for the USA job market (Ettredge et al. 2005), as well as, afterwards in regards to Spain (Vicente et al. 2015).

With respect to elections, an initial approach in (Pion and Hamel 2007) considers election predictions along with predictions in sports and economics. In addition in (Davidowitz and Seth 2013) is argued that Google searches prior to an election can be used to predict turnout in different parts of the USA. Franch (2013) provided predictions for the 2010 UK elections by applying twice the concept behind Galton's predictive "wisdom of the crowds".

3 The Proposed Algorithm

The proposed algorithm is applied on the data generated by the users of the Google search engine. Each time that a user, searches the Web with the Google search engine, the relevant data such as, the typed word or phrase, the date, the time, the location and data related to his/her profile are stored by Google. The data are analyzed by Google and some of them become publicly available by the Google Trends service. In particular Google Trends returns a normalized averaged number that corresponds to the volume of daily searches for a specific term compared to the rest of the search terms.

The proposed algorithm uses the search popularity of selected word/phrases, as provided by the Google Trends, in order to analyze the feelings, intentions and thoughts of the web users in relation to these word/phrases, aiming to predict their future behavior. Early versions of the proposed algorithm have been applied, in several elections races (Polykalas et al. 2013a, b). The algorithm consists of four main phases: initial, words set, noise elimination and runs. At the initial phase the examined time period before the event under study is determined and the geographic restrictions for the web search users is set. During the next phase the popularity of selected words/phrases relevant to the case study is examined, in order to determine the set of words/phrases that will be used as input data. As stated earlier, Google Trends returns a normalized value of the popularity of each word/phrase typed by web search users. We call this popularity the Web Interest (WI) of each typed word/phrase. The WI of each examined word/phrase should fulfill two criteria in order for the relevant word/phrase to be part of the selected words/phrases. The first one concerns the variance of the relevant WI during the determined time period, while the second one is related to the absolute values of the WI during the determined time period. If the WI varies significant during the determined period and the value of the WI is comparable with the WI of previously selected words/phrases, then the examined word/phrase is selected as algorithm input. Several words/phrases should be examined during this phase, in order to include, in the final set of word/phrases, all potential words/phrases than meet the aforementioned two conditions. Having determined the geographical restrictions, the time period and the final set of the words/phrases, the next phase is related to the elimination of potential noise. The noise elimination phase consists of three different sub-phases. The first one deals with the elimination of noise generated by indecisive or confused web users. An indecisive/confused web user is defined as the web user who is searching, at the same time, for words/phrases that show contradicted feelings or unpredictable future behavior. Further explanation is given for this sub-phase in the next section where the proposed algorithm is applied to our case study. The second sub-phase is related to the examination of previous (if any) relevant events similar to the one under study. If there are similar historical events then, the relevant data that were generated by web search engine users, as well as, the relevant historical actual results are used as feedback to the current data used for the case under study. The third sub-phase concerns the exclusion of the influence generated by non-representative facts during the determined time period. In order to determine the non-representative facts, a day to day examination of the WI of the selected words is required. If a selected word presents very high variation during a short period (high increase followed by high decrease within 1–2 days), which is not followed by a respective variation of the WI of the other selected word/phrases, then these WI values should not be considered as valid input values (in practical terms this means that another event, related to the main one such as a TV interview, a scandal etc., that drew high media attention has occurred and has skewed the respective WIs). The last phase of the proposed algorithm contains the final runs of the proposed algorithm, which in turns generate the final results. A normalization of the final results is required only if the number of different set of word/phrases used in the proposed algorithm is less than the relevant actual set of tendencies under study.

4 Applying the Proposed Algorithm

Our case study is related to the analysis of the web search behavior of users located in Greece, aiming to predict the results of the Greek referendum held in 2015. In the early morning of 27th of June 2015,the Greek prime minister announced a referendum to be held on 5th of July 2015. The referendum contained only one question, which was related to the bailout conditions proposed by European Commission (EC), International Monetary Fund (IMF) and European Central Bank (ECB). More specifically, Greeks were asked whether the Greek government should accept or not the plan proposed by EC, IMF and ECB and submitted to the Greek government on the 25th of June. The announcement of the referendum initiated endless public discussions between politicians and in general between Greek people. Public opinion was divided in two main groups: the supporters of "yes" and the supporters of "no". The discussions about the referendum had major implications on the relevant behavior of Greek web users. Several profiles were created in social media in relation to the Greek referendum, while the discussions about the referendum monopolized the messages exchanged in social media.

Taking into account the above facts, we applied the proposed algorithm to our case study. The first steps were related to the determination of the geographic restrictions and the time period. Since the referendum concerns the Greek people, we geographically restricted the web search users only to users with a Greek location (with Greek IP address) during the determined time period. As regards to the determination of the time period, the referendum was announced on 27th of June and was held on 5th of July, therefore the time period was pre-determined from 27th of June to 4th of July. The determination of the time period between 27th of June and 4th of July was verified by the WI variance, during that period, of the most characteristic words used by web users when they were searching for issues relevant to the referendum. More specifically during that period the discussions in all media were focused on whether Greek people will vote "yes" or "no" to the relevant question of the referendum. Therefore the most characteristic words that came to our mind, relevant to referendum were: "ναι" and "όχι" in Greek ("ναι" means yes and "όχι" means no). Indeed, as depicted in the following figure[1] (Fig. 1a), the variance of "οχι" and "ναι" during the 12-month period before the referendum, verifies our determination of the time period between 27th of June to 4th of July, as well as to select "ναι" and "οχι" as the initial set of words under analysis.

Google Trends allows web users to determine the time periods on a monthly basis presenting daily measures. The variation of the WI of the selected words during June – July of 2015, represents a significant increase starting from the date of referendum announcement (27th of June) until two days after the referendum (7th of July). The peak values of both WIs are on Friday 3rd of July. Thus, the variation of the WIs of both selected words ("ναι" and "οχι") meet the two criteria described in previous section, relevant to the variation and the absolute values of WI. Therefore both selected words are selected as valid words in our algorithm.

[1] In order to assist other researchers to verify our results, the graphs contained the figures are the ones provided by the Google Trends service.

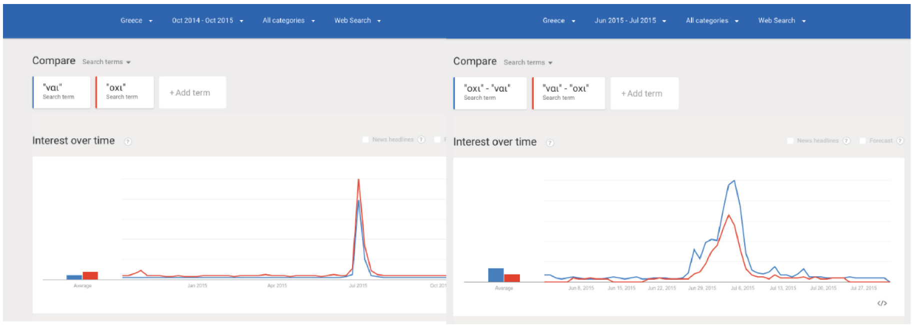


Fig. 1. (a) Determination of time period (right-figure) and (b) WI of "no" indecisive users (left-figure)

Having determined the initial set or words the next step is to examine whether there were other words/phrases, which on one hand characterize the searching behavior of supporters of "yes" or "no" and on the other hand fulfill the relevant two criteria. Therefore we also examined additional forms of the words "yes" and "no", as well as, the names/acronyms/leaders of the major political parties. In particular for "yes" we examined the relevant form with Latin letters ("nai") while for "no" we examined the relevant form with Latin letter ("oxi") and the grammatically correct (with intonation) form of the Greek "no" ("όχι"). In addition we examined the WI for names/acronyms/leaders of political parties that support "yes" (NeaDimokratia – ND, PASOK, Potami, Samaras, Venizelos, Theodorakis), and the WI for names/acronyms/leaders of political parties that support "no" (Syriza, ANEL, Tsipras, Kammenos).

Inserting the words "nai" and "ναι" at the Google Trends and having determined the time period as well as the geographical restrictions, Google Trends in the relevant graph returns WI values only for "ναι". That means that the WI values for "nai" are significantly lower than the WI of "ναι" and thus could not have been presented in the graph (in practice that means that "nai" does not meet the second algorithm criteria). Running similar scenarios for the other words/phrases which may characterize the searching behavior of the supporters of "yes" and "no", as described above, we reached the conclusion that there weren't any other words/phrase that fulfill the two criteria of the proposed algorithm. Thus the only characteristic word for supporters of "yes" and of "no" that should be included in the algorithm were the words "ναι" and "οχι" respectively.

In this case study the sub-phase of historic feedback is not applicable since there is no historical data relevant to referendums in the recent history of Greece. Therefore the next issue under consideration was related to the noise elimination generated by indecisive web users. An indecisive user is the user, whom his/her web search behavior does not indicate clearly his/her intentions, thoughts and feelings about the referendum. For example an indecisive user is the user who searches at the same time for "yes" and "no" typing for example in the web search engine the phrase "yes no referendum". The web search data generated by indecisive web users should be excluded from the valid data, analyzed by the proposed algorithm. Google Trends allows users to exclude a word from a phrase by using the symbol "–". Applying this rule to our algorithm, in order to eliminate noise, we use the followings phrases for each category: “ναι – οχι” for the supporters of “yes” and “οχι- ναι” for the supporters of “no”. In the Fig. 1b is displayed the WI variation of the selected final set of words.

The last phase of the proposed algorithm correlates the outcomes of the proposed algorithm with the actual results of the referendum. To do this, the following steps are followed: let WIi, group x, current, N be the Web Interest value on i day before the referendum for supporters of x (x takes values “no” and “yes”) during the 2015 referendum, and with N the duration (days) for the determined time period. Then the Average Web Interest (AWI) over a period of N days before the referendum date is calculated. In our scenarios N takes the value of 9 (from 27th of June to 4th of July):

$$\mathrm{AWI}_{\mathrm{group\,x,\;current,\;N}} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{WI}_{\mathrm{i,\;group\,x,\;current,\;N}} \tag{1}$$

Running the final runs we reach the conclusion that the algorithm predicts the referendum results with great accuracy. Indeed the algorithm predicts 38,41 % as the percentage of “yes” supporters and 61,59 % as the percentage of “no” supporters, while the actual results were 38,69 % and 61,33 % respectively (predictions error: 0,26 %).

5 Conclusions

We have shown that the proper computation of the data generated by web search users could reveal useful information about their future behavior, such as the will to vote against or not in a forthcoming referendum. In particular by applying the proposed algorithm to the Greek referendum held in 2015, we are able to predict the results of the referendum with high accuracy, based on data generated up to one day before the referendum. It should be noted that mainly due to the lack of historic relevant data the traditional methods of predictions (elections polls) failed to predict the actual results with acceptable accuracy[2]. The successful application of the proposed algorithm in several election races in different countries, the results of the current study, as well as, several research publications in relation to predictions based on online data, allow us to strongly believe that online forecasting, already plays and will continue to play a major role in the predictions of people/consumer behavior, allowing the prediction of human behavior based on web search activities.

[2] https://en.wikipedia.org/wiki/Greek_bailout_referendum,_2015.

References

Choi, H., Varian, H.: Predicting the present with Google Trends wiley online library. In: Economic Record Special Issue: Selected Papers from the 40th Australian Conference of Economists (2012)

Artola, C., Pinto, F., de Pedraza García, P.: Can internet searches forecast tourism inflows? Int. J. Manpower **36**(1), 103–116 (2015)

Ettredge, M., Gerdes, J., Karuga, G.: Using web-based search data to predict macroeconomic statistics. Commun. ACM **48**(11), 87–92 (2005)

Franch, F.: (Wisdom of the crowds)^2: 2010 UK election prediction with social media. J. Inf. Technol. Polit. **10**(1), 57–71 (2013)

Ginsberg, J., Mohebbi, M., Patel, R., Brammer, L., Smolinski, M., Brilliant, L.: Detecting influenza epidemics using search engine query data. Nature **457**, 1012–1014 (2009)

Vicente, M.R., López-Menéndez, A.J., Pérez, R.: Forecasting unemployment with internet search data: Does it help to improve predictions when job destruction is skyrocketing? Technol. Forecast. Soc. Change **92**(2015), 132–139 (2015)

Bangwayo-Skeete, P.F., Skeete, R.W.: Can Google data improve the forecasting performance of tourist arrivals? Mixed-data sampling approach. Tourism Manage. **46**(2015), 454–464 (2015)

Pion, S., Hamel, L.: The internet democracy: a predictive model based on web text mining. In: Proceedings of DMIN 2007, pp. 292–300 (2007)

Stephens-Davidowitz, S.I.: Using Google Data to Predict Who Will Vote (2013). Available at SSRN: http://ssrn.com/abstract=2238863

Polykalas, S.E., Prezerakos, G.N., Konidaris, A.T.: A general purpose model for future prediction based on web search data: predicting Greek and Spanish elections. In: Proceedings of International Symposium on Mining and Web, March 2013, Barcelona, Spain (2013a)

Polykalas, S.E., Prezerakos, G.N., Konidaris, A.T.: An algorithm based on Google Trends' data for future prediction. Case study: German elections. In: Proceedings of International Conference ISSPIT/IEEE 2013, Athens, Greece (2013b)

Wu, L., Brynjolfsson, E.: The future of prediction: how Google searches foreshadow housing prices and sales. In: NBER Conference Technological Progress & Productivity Measurement (2009)

Yang, X., Pan, B., Evans, J.A., Lv, B.: Forecasting Chinese tourist volume with search engine data. Tourism Manage. **46**(2015), 386–397 (2015)

Spatial Bag of Features Learning for Large Scale Face Image Retrieval

Nikolaos Passalis(✉) and Anastasios Tefas

Aristotle University of Thessaloniki, Thessaloniki, Greece
passalis@csd.auth.gr, tefas@aiia.csd.auth.gr

Abstract. In this paper a supervised codebook learning technique for the Bag-of-Features representation that optimizes the learned codebooks towards face retrieval is proposed. This allows to use significantly smaller codebooks reducing both the storage requirements and the retrieval time allowing the proposed technique to efficiently scale to large datasets. The proposed method is also combined with a spatial image segmentation technique that exploits the natural symmetry of the human face to further reduce the size of the extracted representation. It is experimentally demonstrated using one large-scale face recognition dataset, the YouTube Faces Database, as well as two smaller datasets, that the proposed technique can increase the retrieval precision, while reducing the encoding time by almost two orders of magnitude.

1 Introduction

Large-scale face image retrieval has recently received a lot of attention, especially in the context of celebrity face image retrieval [2]. Face image retrieval is defined as the task of retrieving face images of a person given a query image that depicts the face of that person. An image must be retrieved even if the person has a different facial expression, pose, or different illumination conditions exist. Given the vast amount of face images available on the Internet, a large-scale face retrieval technique must be able to successfully handle large amounts of data. Facial image retrieval also poses challenges of high-dimensionality, velocity and variety.

A wide range of methods have been proposed for face recognition and retrieval [18]. Recently, a widely known technique for image retrieval, the Bag-of-Features (BoF) model [4], also known as Bag-of-Visual Words (BoVW) model, has been applied for face recognition/retrieval after being appropriately modified [13,16,17]. The typical pipeline of the BoF model can be summarized as follows. First, multiple features, such as SIFT descriptors [9], are extracted from each image (feature extraction). That way, the *feature space* is formed where each image is represented as an unordered set of features. Then, the extracted features are used to learn a *codebook* of representative features (also called *codewords*). This process is called *codebook learning*. Finally, each feature vector is represented using a codeword from the learned codebook and a histogram is extracted for each image. That way, the *histogram space* is formed where each image is represented by a constant dimensionality histogram vector.

The BoF model discards most of the spatial information contained in the original image, which can severely harm the face recognition precision. To overcome this limitation, the BoF-based techniques for face recognition, e.g., [13,16,17], define a grid over each image (or some regions of interest) and independently extract a histogram from each cell of the grid. These techniques tend to be invariant to facial expression, pose, and illumination variations when used for face recognition using a trained classifier [13]. However, when the extracted representation is used for face image retrieval the problem becomes ill-defined, since the user's information need is sometimes ambiguous. For example, if the query image depicts a smiling person the system might retrieve face images from the same person, but it might also retrieve images from other persons that smile. Therefore, it is critical that the extracted representation is appropriately tuned for the given retrieval task.

The main contribution of this paper is the proposal of a supervised codebook learning technique that is able to learn face retrieval-oriented codebooks using an annotated training set of face images. This allows to use significantly smaller codebooks reducing both the storage requirements and the retrieval time by almost two orders of magnitude and allowing the technique to efficiently scale to large datasets. The proposed method is also combined with a spatial image segmentation technique that exploits the natural symmetry of the human face to further reduce the size of the extracted representation. Furthermore, the proposed approach is able to learn in incremental mode without requiring in-memory access to large amounts of data.

The rest of the paper is structured as follows. The related work is discussed in Sect. 2 and the proposed method is presented in Sect. 3. The experimental evaluation using one large-scale dataset, the YouTube Faces Database and two smaller datasets, the ORL Database of Faces and the Extended Yale Face Database B, is presented in Sect. 4. Finally, conclusions are drawn in Sect. 5.

2 Related Work

This work mainly concerns codebook learning for face image retrieval using the BoF model. A rich literature exists in the field of codebook learning for the BoF representation, ranging from supervised approaches [3,8,10], to unsupervised ones [11]. Despite the fact that supervised codebook learning is well established in general computer vision tasks, such as scene classification [8], or action recognition [3], little work has been done in the area of face image retrieval. One application of supervised codebook learning for face image retrieval is presented in [16], where a simple identity based codebook construction technique is proposed. This method is combined with hamming signatures and locality-sensitive hashing (LSH) in order to be applied to large datasets. In contrast to this, the method proposed in this paper reduces the size of the codebook, instead of relying on hashing and approximate nearest neighbor search techniques, allowing the method to natively scale to large datasets. Nonetheless, these techniques can be still combined with the proposed method to further increase the retrieval performance.

All the proposed BoF-based image recognition/retrieval techniques either use a grid over each image [13,17], or define multiple grids over some points of interest [16], and process each cell independently using a separate codebook. This introduces spatial information to the extracted representation leading to the Spatial BoF (SBoF) model. In our approach each (aligned) face image is split into four horizontal strips instead of dividing each image using a grid. This allows to further reduce the length of the extracted representation, without significantly affecting the retrieval precision.

3 Proposed Method

3.1 Spatial Bag-of-Features Model

Before presenting the proposed codebook learning method, the BoF model and the SBoF model are briefly described. Let N be the number of face images that are to be encoded using the regular BoF model. The i-th image is described by N_i feature vectors: $\mathbf{x}_{ij} \in \mathbb{R}^D$ $(i = 1...N,\ j = 1...N_i)$, where D is the length of the extracted feature vectors. In this work, dense SIFT features [6], are extracted from 16×16 patches using a regular grid with spacing of 4 pixels. The BoF model represents each face image using a fixed-length histogram of its quantized feature vectors, where each histogram bin corresponds to a codeword. In hard assignment each feature vector is quantized to its nearest codeword/histogram bin, while in soft-assignment every feature vector contributes, by a different amount, to each codeword/bin.

In order to learn a codebook the set of all feature vectors $\mathscr{S} = \{\mathbf{x}_{ij} | i = 1...N, j = 1...N_i\}$ is clustered into N_K clusters and the corresponding centroids (codewords) $\mathbf{v}_k \in \mathbb{R}^D (k = 1...N_K)$ are used to form the codebook $\mathbf{V} \in \mathbb{R}^{D \times N_K}$, where each column of $\mathbf{V}$ is a centroid vector. These centroids are used to quantize the feature vectors. It is common to cluster only a subset of $\mathscr{S}$ since this can reduce the training time with little effect on the learned representation. The codebook is learned only once and then it can be used to encode any new face image.

To encode the i-th face image, the similarity between each feature vector $\mathbf{x}_{ij}$ and each codeword $\mathbf{v}_k$ is computed as: $d_{ijk} = exp(\frac{-||\mathbf{v}_k - \mathbf{x}_{ij}||_2}{g}) \in \mathbb{R}$. The parameter g controls the quantization process: for harder assignment a small value, i.e., $g < 0.01$, is used, while for softer assignment larger values, i.e., $g > 0.01$, are used. Then, the l^1 normalized membership vector of each feature vector $\mathbf{x}_{ij}$ is obtained by: $\mathbf{u}_{ij} = \frac{\mathbf{d}_{ij}}{||\mathbf{d}_{ij}||_1} \in \mathbb{R}^{N_K}$. This vector describes the similarity of feature vector $\mathbf{x}_{ij}$ to each codebook vector. Finally, the histogram $\mathbf{s}_i$ is extracted as $\mathbf{s}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} \mathbf{u}_{ij} \in \mathbb{R}^{N_K}$. The histogram $\mathbf{s}_i$ has unit l^1 norm, since $||\mathbf{u}_{ij}||_1 = 1$ for every j. These histograms describe each image and they can be used to retrieve relevant face images. The training and the encoding process are unsupervised and no labeled data are required.

As previously mentioned, the BoF model discards most of the spatial information contained in the images during the encoding process. This can severely harm the face recognition precision, since most of the discriminant face features, e.g., eyes, nose, mouth, are expected to be found in the same position when the face is properly aligned. This allows to have highly specialized codebooks for each of these regions.

In this work the following segmentation technique is used for the SBoF model. Each image is segmented into N_r equally spaced horizontal stripes. Since the human face is symmetric to a great extent, no vertical segmentation is used. A separate codebook is learned for each strip and N_r histograms are extracted (one from each strip). These histograms are fused into the final histogram vector and renormalized. The length of this vector is $N_r \times N_K$. When four regions are used, i.e., $N_r = 4$, and the images are correctly aligned, each region roughly corresponds to one of the following parts of the human face: forehead, eyes, nose and mouth. Although the images can be further split to more regions (either horizontal either vertical or both), this provides little increase in retrieval precision. This fact is experimentally validated in Sect. 4. Note that the storage requirements and the retrieval time depend on the number of regions used to segment each image, since the storage required for each image is $N_r \times N_K \times B$ bytes. This technique also provides mirror-invariance, while still preserving the discriminant non-symmetric facial features that might appear.

3.2 Spatial Bag-of-Features Learning

The goal of the proposed optimized SBoF technique, abbreviated as O-SBoF, is to learn codebooks that minimize the entropy in the histogram space using a training set of face images, where the i-th image is annotated by a label $l_i \in \{1, ..., N_C\}$ and N_C is the number of individuals (classes) in the training set. Intuitively, the entropy in the histogram space is minimized when the histograms are gathered in pure clusters, i.e., each cluster contains face images of the same person.

To simplify the presentation of the proposed method, it is assumed that only one region exists in each image, i.e., the feature vectors are extracted from the whole image. This is without loss of generality, since the method can be independently applied to optimize the codebook of each region/strip by considering each region/strip as an image.

In order to measure the entropy in the histogram space, the vectors $\mathbf{s}_i$ are clustered into N_T clusters. The centroid of the k-th cluster is denoted by $\mathbf{c}_k$ ($k = 1...N_T$). Then, the entropy of the k-th cluster can be defined as:

$$E_k = -\sum_{j=1}^{N_C} p_{jk} \log p_{jk} \tag{1}$$

where p_{jk} is the probability that an image of the k-th cluster belongs to the class j. This probability is estimated as $p_{jk} = h_{jk}/n_k$, where n_k is the number of images in cluster k and h_{jk} is the number of images in cluster k that belong to class j.

Low-entropy clusters, i.e., clusters that contain mostly vectors from images of the same person, are preferable for retrieval tasks to high-entropy clusters, i.e., clusters that contain vectors from images that belong to several different persons. Therefore the aim is to learn a codebook $\mathbf{V}$ that minimize the total entropy of a cluster configuration, which is defined as:

$$E = \sum_{k=1}^{N_T} r_k E_k \tag{2}$$

where $r_k = n_k/N$ is the proportion of images in cluster k.

By substituting r_k and p_{jk} into the entropy definitions given in (1) and (2) the following objective function is obtained:

$$E = -\frac{1}{N} \sum_{k=1}^{N_T} \sum_{j=1}^{N_C} h_{jk} \log p_{jk} \tag{3}$$

However, it is not easy to directly optimize (3) as this function is not continuous with respect to $\mathbf{s}_i$. To this end, a continuous smooth approximation of the previously defined entropy is introduced. To distinguish the previous definition from the smooth entropy approximation, the former is called *hard entropy* and the latter is called *soft entropy*.

A smooth cluster membership vector $\mathbf{q}_i \in \mathbb{R}^{N_T}$ is defined for each histogram $\mathbf{s}_i$, where $q_{ik} = exp(\frac{-||\mathbf{s}_i - \mathbf{c}_k||_2}{m})$. The corresponding smooth l^1 normalized membership vector $\mathbf{w}_i$ is defined as $\mathbf{w}_i = \frac{\mathbf{q}_i}{||\mathbf{q}_i||_1} \in \mathbb{R}^{N_T}$. The parameter m controls the fuzziness of the assignment process: for $m \to 0$ each histogram is assigned to its nearest cluster, while larger values allow fuzzy membership.

Then, the quantities n_k and h_{jk} are redefined as: $n_k = \sum_{i=1}^{N} w_{ik}$ and $h_{jk} = \sum_{i=1}^{N} w_{ik}\pi_{ij}$, where π_{ij} is 1 if the i-th image belongs to class j and 0 otherwise. These modifications lead to a smooth entropy approximation that converges to hard entropy as $m \to 0$.

The derivative of E with respect to $\mathbf{v}_m$ can be calculated as the product of two other partial derivatives: $\frac{\partial E}{\partial \mathbf{v}_m} = \sum_{l=1}^{N} \sum_{\kappa=1}^{N_K} \frac{\partial E}{\partial s_{l\kappa}} \frac{\partial s_{l\kappa}}{\partial \mathbf{v}_m}$. In order to reduce the entropy in the histogram space the histograms $\mathbf{s}_l$ that, in turn, depend on the codebook $\mathbf{V}$, must be shifted. The partial derivative $\frac{\partial E}{\partial \mathbf{s}_l}$ provides the direction in which the histogram $\mathbf{s}_l$ must be moved. Since each codeword $\mathbf{v}_m$ lies in the feature space, the derivative $\frac{\partial s_{l\kappa}}{\partial \mathbf{v}_m}$ projects the previous direction into the codebook.

The calculation of these derivatives is straightforward and it is omitted due to space constraints. Note that the histogram space derivative does not exist when a histogram vector and a centroid vector coincide. The same holds for the feature space derivative when a codebook center and a feature vector also coincide. When that happens, the corresponding derivatives are set to 0.

During the optimization process the image histograms are updated. This might invalidate the initial choice of the centers $\mathbf{c}_k$ that should be also updated during the optimization. Therefore, the derivative of the objective function with respect to each $\mathbf{c}_k$, i.e., $\frac{\partial E}{\partial \mathbf{c}_m}$, is also calculated.

The codebook and the histogram centers can be optimized using gradient descent, i.e., $\Delta\mathbf{V} = -\eta\frac{\partial E}{\partial \mathbf{V}}$ and $\Delta\mathbf{c}_m = -\eta_c\frac{\partial E}{\partial \mathbf{c}_m}$, where η and η_c are the learning rates. In this work, the adaptive moment estimation algorithm, also known as Adam [5], is used instead of the simple gradient descent for the optimization, since it provides faster and more stable convergence. To avoid simply overfitting the histogram space centers, instead of learning the codebook, a smaller learning rate is used for the histogram space centers. For all the conducted experiments the optimization process runs for 100 iterations and the following learning rates are used: $\eta = 0.01$ and $\eta_c = 0.001$. The default parameters are used for the Adam algorithm ($\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$) [5]. Also, to reduce the training time, 100 features vectors are randomly sampled from each region during the training process. This reduces the training time, without affecting the quality of the learned codebook. The proposed approach can be also adapted to work in incremental mode by using small mini-batches of, e.g., 100 face images. However, this is not always necessary since in facial image retrieval the number of annotated training images is usually significantly smaller than the number of images that are to be encoded.

Regarding the initialization of the codebook and the histogram space centers several choices exist. In this work, the codebook is initialized using the k-means algorithm, as in the regular BoF model. For the histogram space centers, N_C centers are used (one for each person) and each center is initialized using the mean histogram of each person.

Finally, the softness parameters m and g must be selected. The parameter g controls the softness of the quantization process, while m controls the histogram's membership fuzziness and should be set to small enough value in order to closely approximate the hard entropy. It was experimentally established that the method is relatively stable with regard to the selected parameters: $m = 0.01$ and $g = 0.01$ is used for all the conducted experiments.

To better understand how the proposed method works, a toy example of the optimization process is provided. Four persons are chosen from the YouTube Faces dataset, which is described in Sect. 4, 30 images are selected for each person and 64 codewords are used for each of the four strips. The histogram space is visualized during the optimization process in Fig. 1. Since the histograms lie in a space with 4×64 dimensions, the PCA technique is used to visualize the resulting histograms. The expression and illumination variations in the face images lead to a representation where two or more subclasses exist for each person. Nonetheless, the optimization of the representation using the proposed O-SBoF technique successfully reduces the overlapping of histograms that belong to different classes and brings the histograms that belong to the same person significantly closer.

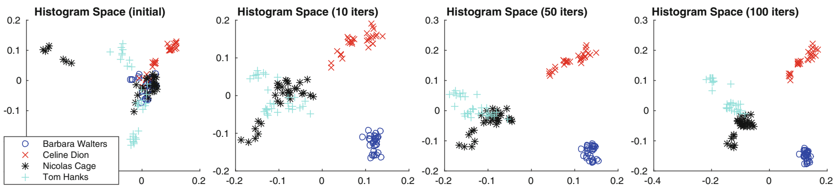


Fig. 1. Histogram space during the optimization using the proposed O-SBoF method

4 Experiments

4.1 Evaluation Setup

Two small-scale face recognition datasets, the ORL Database of Faces (ORL) [12], and the cropped variant of the Extended Yale Face Database B (Yale B) [1,7], and one large-scale face dataset, the YouTube Faces Database [14], were used for the evaluation of the proposed method.

The ORL dataset contains 400 images from 40 different persons (10 images per person) under varying pose and facial expression. The cropped Extended Yale Face Database B contains 2432 images from 38 individuals. The images were taken under greatly varying lighting conditions. The YouTube Faces dataset contains 621,126 frames of 3,425 videos and a total number of 1,595 individuals. The aligned version of this dataset is used, i.e., the face is already aligned in each image using face detection and alignment techniques. Before extracting the feature vectors each image is cropped by removing 25 % of its margins.

For the small-scale experiments each dataset is randomly split to two sets using half of the images of each person as the train set and the rest of them as the test set. The train set is used to build the database and train the O-SBoF model. The retrieval performance is evaluated using the images contained in the test set as queries. The training and the retrieval evaluation process are repeated five times and the mean and the standard deviation of the evaluated metrics are reported.

For the large-scale experiments a different evaluation strategy, similar to those of celebrity face image retrieval tasks [2], is used. The training set is formed by randomly selecting 100 images from the most popular persons that appear in the videos (5,900 training images are collected from the videos of the 59 most popular persons). A person is considered popular if it appears in at least 5 videos. To build the database, the images of persons that appear in at least 3 videos are used (the database contains 356,588 images from 226 persons). To evaluate the retrieval performance 100 queries from the popular persons (celebrities) are randomly selected. The evaluation process is repeated five times and the mean and the standard deviation of the evaluated metrics are reported.

Throughout this paper, two evaluation metrics are used: the interpolated precision (also abbreviated as 'prec.'), and the mean average precision (mAP). The mean average precision (AP) for a given query is computed at eleven equally spaced recall points (0, 0.1, ..., 0.9, 1). A more detailed definition of the used evaluation metrics can be found in [10]. Also, for all the evaluated representations 16-bit floating numbers are used, since this can reduce the storage requirements without harming the retrieval precision.

4.2 Experimental Evaluation

First, the proposed method is evaluated using the ORL and the Yale B datasets. The proposed method is also compared to two other well-established face recognition features, the FPLBP and the TPLBP features (using the code provided by the authors of [15]), and two other SBoF representations with significantly larger codebooks. The results are shown in Table 1. The proposed method can greatly increase the precision over the baseline SBoF representations. For example, in the Yale B Dataset, the mAP and the top-5 precision increase by more than 20 % over the baseline. This allows to match the precision of the other representation techniques using 24 to 64 times smaller representation size. If a slightly larger codebook is used (64 codewords instead of 16), the proposed method greatly outperforms all the other evaluated methods, while still using smaller representation size than the other methods.

Next, the proposed method is evaluated using the large-scale YouTube Faces dataset. The results are shown in Table 2. The precomputed CSLBP and FPLBP features are used [14]. Again, the proposed method outperforms all the other evaluated methods using significantly smaller representation size, leading to better retrieval precision, faster image encoding and retrieval, and lower storage requirements. The O-SBoF method increase the mAP over the best performing SBOF method by 6 %, while reducing the representation size by 16 to 64 times. Regarding the (offline) training time of the proposed O-SBoF method, less than 3 h were required for learning 4 × 64 codewords. However, the proposed method can significantly reduce the online encoding time. The histogram encoding time was reduced from 2 days when using 4096 codewords (12 h for 1024 codewords) to 20 min when using 16 codewords (less than 1 h for 64 codewords). For all the conducted experiments a workstation with two 10-core 2.8 GHz CPUs was used.

Table 1. Small-scale evaluation using the ORL and the Extended Yale B datasets

Method	# codewords	# bytes	ORL Dataset		Yale B Dataset	
			mAP	top-5 prec.	mAP	top-5 prec.
FPLBP	-	896 / 3072	79.56 ± 0.41	70.52 ± 0.65	35.96 ± 0.25	80.03 ± 0.88
TPLBP	-	14336 / 49152	80.78 ± 0.72	71.63 ± 0.81	32.15 ± 0.24	77.21 ± 1.04
SBoF	4×1024	8192	93.29 ± 0.70	88.69 ± 0.70	25.78 ± 0.26	68.76 ± 1.05
SBoF	4×4096	32768	93.70 ± 0.63	89.37 ± 0.42	28.65 ± 0.30	73.10 ± 0.95
SBoF	4×16	128	81.69 ± 0.65	71.95 ± 0.40	18.24 ± 0.28	51.75 ± 1.01
O-SBoF	4×16	128	$\mathbf{93.24 \pm 1.52}$	$\mathbf{88.92 \pm 1.12}$	$\mathbf{37.72 \pm 0.90}$	$\mathbf{79.05 \pm 1.08}$
SBoF	4×64	512	88.90 ± 0.74	81.60 ± 0.94	20.89 ± 0.25	59.43 ± 0.93
O-SBoF	4×64	512	$\mathbf{97.42 \pm 0.72}$	$\mathbf{95.47 \pm 1.00}$	$\mathbf{43.13 \pm 0.64}$	$\mathbf{83.69 \pm 1.14}$

Table 2. Large-scale evaluation using the YouTube Faces dataset

Method	# codewords	# bytes	mAP	top-20 prec.	top-50 prec.	top-100 prec.
CSLBP	-	960	37.06 ± 1.21	98.23 ± 1.21	94.47 ± 2.34	87.93 ± 2.56
FPLBP	-	1120	37.90 ± 1.34	99.15 ± 0.57	96.74 ± 1.61	90.99 ± 2.35
SBoF	4×1024	8192	41.19 ± 1.04	99.99 ± 0.02	99.05 ± 0.26	91.50 ± 1.41
SBoF	4×4096	32768	40.95 ± 1.07	99.90 ± 0.12	98.54 ± 0.51	90.99 ± 1.48
SBoF	4×16	128	32.13 ± 1.35	98.93 ± 0.46	94.72 ± 1.25	84.47 ± 2.25
O-SBoF	4×16	128	**40.89 ± 1.31**	**99.71 ± 0.28**	**97.45 ± 0.70**	**88.78 ± 1.77**
SBoF	4×64	512	38.60 ± 1.32	99.84 ± 0.17	98.18 ± 0.43	89.55 ± 1.31
O-SBoF	4×64	512	**47.19 ± 1.24**	**99.93 ± 0.13**	**99.41 ± 0.31**	**92.17 ± 1.29**

Finally, to justify the choice of the spatial segmentation into 4 horizontal strips a set of experiments using different grid layouts was performed. The results are shown in Table 3. The 4×1 grid, i.e., using 4 horizontal strips, achieves the best precision in the ORL Dataset, while only slightly decreases the precision over the 4×4 grid in the Yale B Dataset. However, the 4×4 grid uses 16 codebooks instead of 4, quadrupling the size of the extracted histograms. Therefore, a 4×1 grid was used for all the conducted experiments using the SBoF and the O-SBoF methods.

Table 3. Effect of varying the grid layout to the mAP

Grid layout	1×1	4×1	1×4	4×2	2×4	4×4
ORL dataset	79.58 ± 0.65	**88.90 ± 0.74**	76.17 ± 0.82	85.75 ± 0.51	78.73 ± 0.44	81.34 ± 0.40
Yale B dataset	13.30 ± 0.16	20.89 ± 0.25	15.50 ± 0.09	20.11 ± 0.18	18.53 ± 0.16	**22.14 ± 0.22**

5 Conclusions

In this paper, a supervised codebook learning technique for face retrieval was presented. It was demonstrated using one large-scale dataset and two smaller datasets that the proposed technique can improve the retrieval precision and, at the same time, reduce the storage requirements and the retrieval time by almost two orders of magnitude.

There are several future research directions. In this work it was assumed that the face images were already aligned using face detection and alignment techniques and a fixed grid was used for the O-SBoF technique. However, the proposed method can be combined with facial feature detectors to more accurately define the regions used for feature extraction and further improve the retrieval precision and reduce the representation size. Furthermore, using more robust feature extractors is expected to improve the face recognition precision even more.

References

1. Georghiades, A., Belhumeur, P., Kriegman, D.: From few to many: illumination cone models for face recognition under variable lighting and pose. IEEE Trans. Pattern Anal. Mach. Intell. **23**(6), 643–660 (2001)
2. Guo, Y., Zhang, L., Hu, Y., He, X., Gao, J.: Ms-celeb-1m: challenge of recognizing one million celebrities in the real world. In: IS&T International Symposium on Electronic, Imaging (2016)
3. Iosifidis, A., Tefas, A., Pitas, I.: Discriminant bag of words based representation for human action recognition. Pattern Recogn. Lett. **49**, 185–192 (2014)
4. Jégou, H., Douze, M., Schmid, C.: Improving bag-of-features for large scale image search. Int. J. Comput. Vis. **87**(3), 316–336 (2010)
5. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. Computing Research Repository, abs/1412.6980 (2014)
6. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, 2169–2178 (2006)
7. Lee, K., Ho, J., Kriegman, D.: Acquiring linear subspaces for face recognition under variable lighting. IEEE Trans. Pattern Anal. Mach. Intell. **27**(5), 684–698 (2005)
8. Lian, X.-C., Li, Z., Lu, B.-L., Zhang, L.: Max-margin dictionary learning for multiclass image categorization. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 157–170. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15561-1_12
9. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the 7th IEEE International Conference on Computer Vision, vol. 2, pp. 1150–1157 (1999)
10. Passalis, N., Tefas, A.: Entropy optimized feature-based bag-of-words representation for information retrieval. IEEE Trans. Knowl. Data Eng. **28**, 1664–1677 (2016)
11. Passalis, N., Tefas, A.: Spectral clustering using optimized bag-of-features. In: Proceedings of the 9th Hellenic Conference on Artificial Intelligence, p. 19 (2016)
12. Samaria, F.S., Harter, A.C.: Parameterisation of a stochastic model for human face identification. In: Proceedings of the Second IEEE Workshop on Applications of Computer Vision, pp. 138–142 (1994)
13. Wang, C., Wang, Y., Zhang, Z.: Patch-based bag of features for face recognition in videos. Biometric Recogn. **7701**, 1–8 (2012)
14. Wolf, L., Hassner, T., Maoz, I.: Face recognition in unconstrained videos with matched background similarity. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 529–534 (2011)
15. Wolf, L., Hassner, T., Taigman., Y.: Descriptor based methods in the wild. In: Real-Life Images workshop at the European Conference on Computer Vision (ECCV) (2008)
16. Wu, Z., Ke, Q., Sun, J., Shum, H.-Y.: Scalable face image retrieval with identity-based quantization and multireference reranking. IEEE Trans. Pattern Anal. Mach. Intell. **33**(10), 1991–2001 (2011)
17. Yang, S., Bebis, G., Chu, Y., Zhao, L.: Effective face recognition using bag of features with additive kernels. J. Electron. Imaging **25**(1), 013025 (2016)
18. Zhang, X., Gao, Y.: Face recognition across pose: a review. Pattern Recogn. **42**(11), 2876–2896 (2009)

Compact Video Description and Representation for Automated Summarization of Human Activities

Ioannis Mademlis(✉), Anastasios Tefas, Nikos Nikolaidis, and Ioannis Pitas

Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece
imademlis@aiid.csd.auth.gr

Abstract. A compact framework is presented for the description and representation of videos depicting human activities, with the goal of enabling automated large-volume video summarization for semantically meaningful key-frame extraction. The framework is structured around the concept of per-frame visual word histograms, using the popular Bag-of-Features approach. Three existing image descriptors (histogram, FMoD, SURF) and a novel one (LMoD), as well as a component of an existing state-of-the-art activity descriptor (Dense Trajectories), are adapted into the proposed framework and quantitatively compared against each other, as well as against the most common video summarization descriptor (global image histogram), using a publicly available annotated dataset and the most prevalent video summarization method, i.e., frame clustering. In all cases, several image modalities are exploited (luminance, hue, edges, optical flow magnitude) in order to simultaneously capture information about the depicted shapes, colors, lighting, textures and motions. The quantitative evaluation results indicate that one of the proposed descriptors clearly outperforms the competing approaches in the context of the presented framework.

Keywords: Video summarization · Video description

1 Introduction

Several applications exist nowadays where large-scale video footage depicting human activities needs to be analyzed, possibly on a frame-by-frame basis, requiring human intervention. Examples include professional capture sessions, where the action described in the script is typically filmed using multiple cameras, or streams from surveillance cameras which may be capturing continuously for many days. A very large volume of data are usually produced in such scenarios, which may well exceed 6TB per day [1]. This amount of data is difficult to

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement number 316564 (IMPART).

be efficiently assessed and analysed manually, demanding a great deal of human effort.

Video summarization can be employed as an automated solution to such problems, by generating a condensed version of the video that only contains the most important content [2]. Subsequently, these summaries may be used instead of the original video streams in order to alleviate storage or computational requirements, or the necessary human labour, e.g., in case manual annotation is needed. Most summarization methods initially select a subset of important video frames (*key-frames*) that compactly represent the entire video content. The abstracted content that needs to be included in the target summary can be represented either as an ordered set of static key-frames, or as a dynamic video skim, with the former being more suitable for indexing, browsing and retrieval applications [3]. Evaluation of the success of a summarization method is typically subjective, due to the inherently subjective nature of the task.

Video frames are initially described by low-level image descriptors, such as global color-based, texture-based or shape-based features [4]. In general, the most commonly employed frame descriptors are variants of global joint image histograms in the HSV color space [5,6]. In order to bring down the computational requirements of the subsequent summarization process, dimensionality reduction on such color histograms has been attempted [7]. In [8] the low-level Frame Moments Descriptor (FMoD) is introduced, a video descriptor designed for compactly capturing statistical characteristics of several image modalities, both in a global and in various local scales. In a number of works [3,9], local, recognition-oriented image descriptors have been employed for video description (e.g., Scale-Invariant Feature Transform (SIFT) [10] or Speeded-Up Robust Features (SURF) [11]), using the popular Bag-of-Features (BoF) representation model [12]. In general, video description and representation methods specifically suited to video summarization have not been studied extensively.

Key-frame extraction typically includes clustering the frame descriptors into groups. Subsequently, a set of frames that are closest to each cluster centroid are initially selected as key-frames. In many cases, information about the way a video is naturally segmented into shots (e.g., in movies [8]) is also exploited to assist the summarization process [6,13], e.g. by applying clustering at shot-level. Typically, a number of the extracted key-frames are filtered out to reduce redundancy and the rest are presented in temporal order.

Although the above approaches are oriented towards generic video input, methods exploiting video type-specific information have also been proposed. In surveillance videos, temporal segmentation (shot boundaries detection [14]) is not a viable option due to the lack of cuts, therefore motion detection is employed in order to create summaries that contain sets of object actions, like pedestrian walking. Detected actions taking place in different direction and speed, are fused into a single scene to form a short length video or graphical cue containing as many actions as possible [15]. However, in unedited videos from professional capture sessions (e.g., in TV/movie production), which are also filmed with a static camera and not clearly segmented into shots, such an approach is not applicable. The preferred summarization goal would naturally be to select one key-frame per depicted activity. Moreover, while in both cases the most important visual clue is human motion, state-of-the-art human activity descriptors such as Dense Trajectories [16] cannot be readily employed, due to the exceptionally high memory/computational requirements and their unsuitability for per-frame descriptions, given that accurate activity descriptions extend temporally in multiple neighbouring video frames.

This work attempts to investigate video description/representation specifically suited to video summarization tasks. It presents a compact framework for the description and representation of videos depicting human activities, with the goal of enabling automated large-volume video summarization for semantically meaningful key-frame extraction. The goal is to succinctly summarize a video by, ideally, selecting one key-frame per depicted activity segment. The framework is structured around the concept of per-frame visual word histograms, using the established BoF approach. This scheme successfully captures the distribution of elemental visual building blocks at each video frame and has been proven suitable for discriminative representation of human activities, in tasks such as activity recognition [17] or temporal activity segmentation [18].

Three existing image descriptors (histogram, FMoD, SURF), a novel one (LMoD), as well as a component of Dense Trajectories, are adapted into the proposed framework. The image descriptors are quantitatively compared against each other, as well as against the most common video summarization descriptor, i.e., global image histogram, a variant of which was shown in [3] to outperform all competing approaches when no shot boundaries information was available. The two local descriptors (SURF and LMoD) are evaluated with and without the addition of the implemented Dense Trajectories variant. A publicly available annotated dataset [19] and the most common video summarization method, i.e., frame clustering, are adopted. In all cases, several image modalities are being exploited (luminance, hue, edges, optical flow magnitude) in order to simultaneously capture information about the depicted shapes, colors, lighting, textures and motions. A simple, objective evaluation metric is employed for comparing the competing descriptors.

2 A Framework for Describing and Representing Activity Videos

In the proposed approach, each video is assumed to be composed of a temporally ordered sequence $\mathcal{V}$ of N_f video frames, each one being a set $\mathcal{V}_i$ of K matrices $\mathbf{V}_{ik} \in \mathbb{R}^{M \times N}$, where $0 \leq i < N_f$ and $k \in l, h, o, e$. K is the number of available image modalities: l stands for luminance, h for color hue, o for optical flow magnitude and e for edge map. Each $\mathbf{V}_{ik}$ is a digitized 8-bit image with a resolution of $M \times N$ pixels.

The presented framework operates by initially computing a set of low-level, L-dimensional description vectors at each $\mathbf{V}_{ik}$. For a given i, a single set of descriptors $\mathcal{D}_i$ is subsequently derived from all image modalities, by simply concatenating corresponding vectors computed for different values of k. The correspondence among modalities is established in terms of spatial pixel coordinate matching. Each $\mathcal{D}_i$, composed of P_i LK-dimensional, multimodal description vectors, is then transformed into a single histogram feature vector $\mathbf{d}_i$ by following a Bag-of-Features representation approach [12]. That is, all multimodal description vectors from the entire video are clustered into Kc representative groups, called visual words. The set of all cluster centroids is called a *codebook* and c is the codebook size parameter. Each of the P_i vectors in $\mathcal{D}_i$ is subsequently assigned to the nearest visual word, in terms of Euclidean distance. The number of description vectors assigned to each of the Kc clusters is an entry in a Kc-dimensional vector. This vector is followingly transformed into a histogram by L_1-normalization, in order to produce the final Kc-dimensional video frame feature vector $\mathbf{d}_i$. The histogram construction process is repeated for all N_f values of i.

Any type of low-level descriptor can be employed during the first step of the algorithm. This is trivial in the case of local descriptors, such as SIFT or SURF, but not when global descriptors are to be used. In the context of the proposed framework, a variant of the approach presented in [8] has been adopted for any global descriptor. That is, each $M \times N$ video frame $\mathbf{V}_{ik}$, is partitioned in small blocks of $m \times n$ pixels, where $m < M$ and $n < N$. A description vector is then computed separately for each such block. The process is successively repeated d times, for larger values of m and n, until $m = M$ and $n = N$ during the last iteration. From an implementation point-of-view, this is executed recursively, in a top-down manner, with the image region that is currently being described at each time, subsequently being partitioned into 4 quadrants. These quadrants serve as input blocks to the 4 recursive function calls of the next step. Thus, the total number S of produced description vectors is given by the sum of the first d terms of a geometric progression:

$$S(d) = 1 \cdot 4^0 + 1 \cdot 4^1 + \cdots + 1 \cdot 4^{d-1} = (4^d - 1)/3 \tag{1}$$

In the end, a set of description vectors over various image regions and for various scales is produced, including one truly global description vector (computed over the entire $\mathbf{V}_{ik}$). Available local information is more spatially focused for higher values of d, at the cost of higher computational requirements. In general, however, the main advantage of global image descriptors, i.e., rapid computation [20], is mostly retained with this simple spatial partitioning scheme, in comparison to more complicated alternative approaches, such as image segmentation.

Below, the descriptors used for the evaluation of the proposed framework are presented.

2.1 Global Descriptors

Global histograms computed in various image modalities are the most commonly used feature descriptors for video summarization. For instance, in [6], 16-bin hue histograms derived from the frame representation in the HSV color space are employed. In the presented framework, a histogram resolution of 16 bins is also adopted in the context of the multimodal, frame partitioning scheme previously described.

FMoD [8] was also adopted and adapted to our multimodal, frame partitioning scheme. FMoD operates at each $m \times n$ block by computing one *profile vector* for the horizontal dimension and one for the vertical dimension, through averaging pixel values across block columns/rows, respectively. The result is an n-dimensional and an m-dimensional profile vector. Each of the two vectors is summarized by their first 4 statistical moments (mean, standard deviation, skewness, kurtosis). The resulting 8-dimensional vector $\mathbf{f}_i = [m_{1H}, m_{2H}, m_{3H}, m_{4H}, m_{1V}, m_{2V}, m_{3V}, m_{4V}]^T$ compactly captures the statistical properties of the block.

In this work, FMoD was extended by adding first-order statistical texture analysis components to the summary of each profile vector, i.e., energy and entropy. Moreover, on top of the horizontal and the vertical profile vector, a third *block vector* is constructed by vectorizing the actual block in row-major order. The same statistical synopsis is also applied on this vector, resulting in an 18-dimensional block description vector $\mathbf{f}_i = [m_{1H}, \cdots, m_{6H}, m_{1V}, \cdots, m_{6V}, m_{1B}, \cdots, m_{6B}]^T$. Using this notation, H, V and B refer to the extracted statistical properties of horizontal profile vectors, vertical profile vectors and block vectors, respectively.

2.2 Local Descriptors

The most commonly employed local image descriptor is SIFT, with the less computationally costly SURF being a close second choice. Both produce histograms of edge orientations for carefully selected image interest points, in a scale- and rotation-invariant manner. This stems from their design with object recognition tasks in mind, but is not necessarily an ideal approach for the domain of activity video summarization. Since the video description process is not meant to enable successful video classification, but salient key-frame extraction, and given that the subsequent BoF representation step provides (to a degree) several invariances, local descriptors can be of a *holistic* nature, i.e., they would only need to compactly capture major characteristics of untransformed image patches, covering most of (or even the entire) video frame.

In the presented framework, the SURF detector and descriptor [11] was adopted, for reasons of computational speed. Interest point detection occurs on the luminance video frame modality and the detected key-point coordinates are used for SURF description vectors computation on all employed modalities.

Additionally, the global, extended FMoD descriptor discussed in Subsect. 2.1, provided the basis for a novel local descriptor, called Local Moments Descriptor (LMoD). LMoD operates in the manner presented below.

To describe a given image block $\mathbf{B}$ with a dimension of $b \times b$ pixels, $\mathbf{B}$ is recursively partitioned into quadrant sub-blocks using the frame partitioning scheme of the global descriptor case. For each sub-block, the 18-dimensional description vector of the extended FMoD descriptor is computed and all such vectors are concatenated into an $18S(d)$-dimensional block description vector, where $S(d)$ is given by Eq. (1).

Instead of sparsely detecting interest points, as in the case of SIFT or SURF, the luminance modality of the i-th video frame ($\mathbf{V}_{il}$) is densely sampled on a rectangular grid to extract the block centers where LMoD vectors are to be computed, using a sampling step of s pixels. Subsequently, each candidate block is checked for luminance homogeneity, in order to dismiss blocks conveying minimal information. To achieve rapid computation, this is simply implemented using a threshold t_l on the standard deviation of the block luminance.

Dense sampling of interest points allows the background of a depicted activity to be taken into account and complements the holistic nature of LMoD descriptors. As in the case of SURF, description vectors are constructed on all employed modalities at the spatial coordinates computed in $\mathbf{V}_{il}$.

2.3 Activity Descriptor

The multimodal global and local descriptors previously presented are able to describe each video frame in several ways. However, motion information is provided only from the optical flow magnitude image modality and, therefore, is too temporally localized. Since the focus is on activity videos, an additional descriptor may be employed along with local descriptors, which attempts to capture consistent motion in wide temporal windows. This descriptor, called Trajectories, has been adopted and adapted from one component of the state-of-the-art Dense Trajectories description algorithm (designed for activity recognition) [16].

Below, a set of $\mathcal{D}_i$ computed description vectors (corresponding to D_i detected interest points) is assumed for each $\mathbf{V}_{il}$. Trajectories operate by tracking each local interest point along consecutive video frames, using the estimated optical flow magnitude image modality, for a temporal window of T_w frames, in a sliding window approach. Thus, for each local descriptor, a temporally ordered sequence of T_w coordinate pairs (x, y) (referring to horizontal and vertical pixel positions, respectively) is produced, which is equivalent to a set of spatiotemporal coordinate triplets of the form (x, y, t). If a coordinate triplet is shared among different sequences, one sequence is retained and the rest are discarded. A $2T_w$-dimensional vector of spatial displacements is then computed from each sequence, by subtracting the corresponding spatial coordinates among all pairs of subsequent video frames. Static vectors of displacements (derived from interest points with no motion) are eliminated, through comparing their L_1 norm with a threshold t_s. Finally, each retained vector is normalized by the sum of its displacement magnitudes. The result is a *trajectory* description vector $\mathbf{t}_j$, $0 \leq j < P$, where P is the total number of estimated trajectories across the entire video. Each $\mathbf{t}_j$ encodes relative motion direction patterns across a wide temporal window, in a partially scale-invariant manner. For each trajectory starting at video frame b, b is also recorded and, thus, it is trivial to determine which frames $\mathbf{t}_j$ passes through (i.e., which ones are *contained* in it). The starting frame of $\mathbf{t}_j$ is hereafter denoted by b_j.

The set of all trajectory vectors from all video frames is employed to construct a codebook of size c, which is subsequently used to compute a trajectory histogram $\mathbf{h}_i$ per video frame $\mathbf{V}_{il}$. Unlike in the traditional BoF approach, computing $\mathbf{h}_i$ includes a simple weighting scheme: the contribution of each trajectory $\mathbf{t}_j$ is weighted based on the relation between temporal positions b_j and i. That is, the corresponding weight w_j is derived from a discrete Gaussian over the temporal axis with its peak at position i, where each $\mathbf{t}_j$ is assigned to position $b_j + \lceil (T_w/2) \rceil$. Obviously, trajectories not containing position i are completely disregarded. In the end, a c-dimensional trajectory histogram $\mathbf{h}_i$ has been produced for each video frame $\mathbf{V}_{il}$, encoding spatiotemporal activity information.

By employing the approach described above, activity motion descriptions are computed as video features complementary to local description vectors, in a manner that allows per-frame activity representation.

3 Quantitative Evaluation

3.1 Evaluation Dataset

In order to experimentally evaluate the proposed framework and descriptors, a subset of the publicly available, annotated IMPART video dataset [19] was employed. It depicts three subjects/actors in two different settings: one outdoor and one indoor. A living room-like setting was set-up for the latter, while two scripts were executed during shooting, prescribing human activities by a single human subject: one for the outdoor and one for the indoor setting. In each shooting session, the camera was static and the script was executed three times in succession, one time per subject/actor. This was repeated three times per script, for a total of 3 indoor and 3 outdoor shooting sessions. Thus each script was executed three times per actor. Three main actions were performed, namely “Walk”, “Hand-wave” and “Run”, while additional distractor actions were also included and jointly categorized as “Other” (e.g., “Jump-up-down”, “Jump-forward”, “Bend-forward”). During shooting, the actors were moving along predefined trajectories defined by three waypoints (A, B and C). Summing up, the dataset consists of 6 MPEG-4 compressed video files with a resolution of 720×540 pixels, where each one depicts three actors performing a series of actions one after another. The mean duration of the videos is about 182 s, or 4542 frames.

The fact that ground truth annotation data provided along with the IMPART dataset describe not key-frames pre-selected by users, as in [6] (which would be highly subjective), but obvious activity segment frame boundaries, was exploited to evaluate the proposed framework as objectively as possible. Given the results of each summarization algorithm for each video, the number of extracted key-frames derived from actually different activity segments (hereafter called *independent key-frames*) can be used as an indication of summarization success. Therefore, the ratio of extracted independent key-frames by the total number of requested key-frames K, hereafter called *Independence Ratio* (IR) score, is a practical evaluation metric.

3.2 Experimental Results

The proposed framework and video frame descriptors, as well as a multimodal variant of the global image histogram (without the BoF representation stage) which is popular in the relevant literature (e.g., in [5,6]), were evaluated on the presented IMPART dataset, using the IR metric and the K-Means++ algorithm [21] for frame clustering, as the main summarization method. Other clustering algorithms have been tested and shown to provide similar results. The method in [22] was employed for optical flow estimation. The Laplace operator was used for deriving the edge map image modality, after 3×3 median filtering for noise suppression.

The number of clusters K, i.e., the number of requested extracted key-frames per video, is a user-provided parameter which controls the grain of summarization. Typically, in clustering-based summarization approaches, K is set proportionally to video length and in accordance with the desired summary conciseness. In order to most effectively compare the different description and representation schemes in terms of the achieved IR score, the actual number Q of different activity segments (known from the ground truth) was used as K for each video. Codebook size c was set to 80, while frame partitioning depth d was set to 6 for global descriptors and to 2 for LMoD. Block dimension, interest point sampling step and luminance dispersion threshold were set to 25 pixels, 20 pixels and 20 standard deviation units, respectively, for LMoD. Interest point tracking temporal window width T_w was set to 15 video frames for Trajectories, as in [16]. The experiments were performed on a high-end PC, with a Core i7 @ 3.5 GHz CPU and 32 GB RAM, while the codebase was developed in C++.

Table 1 presents the IR scores, averaged over the entire employed dataset, that were achieved by the competing approaches. Two global descriptors (Framework Histogram, extended FMoD), two local descriptors (SURF, LMoD) and two composite schemes consisting of the local descriptors and the presented per-frame activity descriptor (SURF+Trajectories, LMoD+Trajectories) were compared. In the last case, the BoF-derived histograms from the local and the activity descriptors were concatenated before frame clustering. Additionally, a traditional 16-bin global image histogram descriptor (omitting the frame partitioning and the BoF representation stages) was employed, for a total of 7 competing approaches. In all cases, all discussed video frame modalities (luminance, color hue, optical flow magnitude map, edge map) were exploited through description vector concatenation.

Table 2 presents the mean required execution times per-frame (in milliseconds), over the entire employed dataset, that were achieved by the competing approaches. These measurements include the time necessary for all description and representation stages for all image modalities, as well as the time needed for image modality computation per-frame.

As it can be seen, the proposed framework is outperformed by the typically used global image histogram only when local SURF descriptors are used, which confirms the findings of [3]. In all other cases, the presented description and representation schemes achieve higher performance, with Extended FMoD being more

Table 1. A comparison of the mean IR scores for different video description and representation methods, using K-Means++ summarization.

Method	Mean IR
Framework Histogram	0.685
Extended FMoD	0.723
LMoD	0.740
SURF	0.484
LMoD+Trajectories	**0.802**
SURF+Trajectories	0.553
Global Histogram	0.571

Table 2. A comparison of the mean execution time requirements per-frame for different video description and representation methods.

Method	Mean time (msecs)
Framework Histogram	1077
Extended FMoD	1508
LMoD	1405
SURF	1208
LMoD+Trajectories	2204
SURF+Trajectories	1998
Global Histogram	**706**

successful than Framework Histogram and LMoD providing the best results in both metrics. Additionally, the Trajectories per-frame activity descriptor seems to beneficially enrich the informational content of both employed local descriptors (LMoD and SURF), resulting in the combination LMoD+Trajectories being the best choice. Obviously, it is reasonable that activity description aids the summarization of activity videos. Not unexpectedly, this comes at the cost of a three-fold increase in required computational time in comparison to the traditional global image histograms. This indicates a typical trade-off between summarization quality and computational requirements, with better performing descriptors being more appropriate for off-line/non real-time applications.

The very low performance of SURF is of high interest, since it validates our assumption that sparsely sampled and highly invariant descriptors designed for recognition tasks are not necessarily suitable for video summarization. This fits with previous results in [3], which indicated that in the absence of clear shot boundary information, global image color histograms produced better results than SIFT and SURF. A local descriptor that is holistic, according to our definition, and densely sampled, i.e., LMoD, outperforms both approaches, possibly because it captures spatial image properties lost in the case of simple global histograms and information content discarded by recognition-oriented local descriptors. Overall, the presented framework seems to be more efficient when employing holistic local descriptors.

4 Conclusions

We have proposed a consistent video description and representation framework that accommodates per-frame local, global and activity descriptors, with the goal of assisting successful automated summarization of human activity videos. The framework has been objectively evaluated on a publicly available dataset (IMPART), using the most common video summarization method, i.e., frame clustering. In all cases, several image modalities are being exploited (luminance, hue, edges, optical flow magnitude) in order to simultaneously capture information about the depicted shapes, colors, lighting, textures and motions. In this context, the introduced Extended FMoD, LMoD and Trajectories descriptors (specially adapted novel extensions of pre-existing descriptors) outperform competing approaches, with the LMoD+Trajectories combination proving to be the most effective.

References

1. Evans, A., Agenjo, J., Blat, J.: Combined 2D and 3D web-based visualisation of on-set big media data. In: IEEE International Conference on Image Processing (ICIP), pp. 1120–1124 (2015)
2. Money, A.G., Agius, H.: Video summarization: a conceptual framework and survey of the state of the art. J. Vis. Commun. Representation **19**(2), 121–143 (2008)
3. Cahuina, E.J., Chavez, G.C.: A new method for static video summarization using local descriptors and video temporal segmentation. In: Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 226–233. IEEE (2013)
4. Hu, W., Xie, N., Li, L., Zeng, X., Maybank, S.: A survey on visual content-based video indexing and retrieval. IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev. **41**(6), 797–819 (2011)
5. Zhuang, Y., Rui, Y., Huang, T., Mehrotra, S.: Adaptive key frame extraction using unsupervised clustering. In: International Conference on Image Processing (ICIP), pp. 866–870. IEEE (1998)
6. De Avilla, S.E.F., Lopes, A.P.B., Luz Jr., A.L., Araujo, A.A.: VSUMM: a mechanism designed to produce static video summaries and a novel evaluation method. Pattern Recogn. Lett. **32**(1), 56–68 (2011)
7. Wan, T., Qin, Z.: A new technique for summarizing video sequences through histogram evolution, pp. 1–5. IEEE (2010)
8. Mademlis, I., Nikolaidis, N., Pitas, I.: Stereoscopic video description for key-frame extraction in movie summarization. In: European Signal Processing Conference (EUSIPCO), pp. 819–823. IEEE (2015)
9. Li, J.: Video shot segmentation and key frame extraction based on SIFT feature. In: International Conference on Image Analysis and Signal Processing (IASP), pp. 1–8. IEEE (2012)

10. Lowe, D.G.: Object recognition from local scale-invariant features. In: International Conference on Computer Vision (ICCV), pp. 1150–1157. IEEE (1999)
11. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (SURF). Comput. Vis. Image Underst. **110**(3), 346–359 (2008)
12. Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: European Conference on Computer Vision (ECCV), pp. 1–2 (2004)
13. Tian, Z., Xue, J., Lan, X., Li, C., Zheng, N.: Key object-based static video summarization. In: ACM International Conference on Multimedia, pp. 1301–1304 (2011)
14. Cernekova, Z., Pitas, I., Nikou, C.: Information theory-based shot cut/fade detection and video summarization. IEEE Trans. Circuits Syst. Video Technol. **16**(1), 82–91 (2006)
15. Fu, W., Wang, J., Gui, L., Lu, H., Ma, S.: Online video synopsis of structured motion. Neurocomputing **135**, 155–162 (2014)
16. Wang, H., Kläser, A., Schmid, C., Liu, C.L.: Action recognition by Dense Trajectories. In: IEEE Conference on Computer Vision & Pattern Recognition (CVPR), pp. 3169–3176 (2011)
17. Mademlis, I., Iosifidis, A., Tefas, A., Nikolaidis, N., Pitas, I.: Exploiting stereoscopic disparity for augmenting human activity recognition performance. Multimedia Tools Appl. **75**, 1–20 (2015)
18. Kourous, N., Iosifidis, A., Tefas, A., Nikolaidis, N., Pitas, I.: Video characterization based on activity clustering. In: International Conference on Electrical and Computer Engineering (ICECE), pp. 266–269. IEEE (2014)
19. Kim, H., Hilton, A.: Influence of colour and feature geometry on multi-modal 3D point clouds data registration. In: International Conference on 3D Vision (3DV), pp. 202–209 (2014)
20. Penatti, O., Valle, E., da Silva Torres, R.: Comparative study of global color and texture descriptors for Web image retrieval. J. Vis. Commun. Image Representation **23**(2), 359–380 (2012)
21. Arthur, D., Vassilvitskii, S.: K-Means++: the advantages of careful seeding. In: Symposium on Discrete Algorithms, pp. 1027–1035. Society for Industrial and Applied Mathematics (2007)
22. Farnebäck, G.: Two-frame motion estimation based on polynomial expansion. In: Bigun, J., Gustavsson, T. (eds.) SCIA 2003. LNCS, vol. 2749, pp. 363–370. Springer, Heidelberg (2003). doi:10.1007/3-540-45103-X_50

Incremental Estimation of Visual Vocabulary Size for Image Retrieval

Ilias Gialampoukidis(✉), Stefanos Vrochidis, and Ioannis Kompatsiaris

Centre for Research and Technology Hellas, Information Technologies Institute, 6th Km Charilaou-Thermi Road, 57001 Thermi-Thessaloniki, Greece
{heliasgj,stefanos,ikom}@iti.gr

Abstract. The increasing amount of image databases over the last years has highlighted our need to represent an image collection efficiently and quickly. The majority of image retrieval and image clustering approaches has been based on the construction of a visual vocabulary in the so called Bag-of-Visual-words (BoV) model, analogous to the Bag-of-Words (BoW) model in the representation of a collection of text documents. A visual vocabulary (codebook) is constructed by clustering all available visual features in an image collection, using k-means or approximate k-means, requiring as input the number of visual words, i.e. the size of the visual vocabulary, which is hard to be tuned or directly estimated by the total amount of visual descriptors. In order to avoid tuning or guessing the number of visual words, we propose an incremental estimation of the optimal visual vocabulary size, based on the DBSCAN-Martingale, which has been introduced in the context of text clustering and is able to estimate the number of clusters efficiently, even for very noisy datasets. For a sample of images, our method estimates the potential number of very dense SIFT patterns for each image in the collection. The proposed approach is evaluated in an image retrieval and in an image clustering task, by means of Mean Average Precision and Normalized Mutual Information.

Keywords: DBSCAN-Martingale · Visual vocabulary size estimation · Clustering

1 Introduction

Image retrieval and image clustering are related tasks because of their need to efficiently and quickly search for nearest neighbors in an image collection. Taking into account that image collections are dramatically increasing (e.g. Facebook and Flickr), both tasks, retrieval and clustering, become very challenging and traditional techniques show reduced functionality. Nowadays, there are many applications of image retrieval and clustering, to assist personal photo organization and search.

Searching in an image collection for similar images is strongly affected by the representation of all images. Spatial verification techniques for image representation, like RANSAC and pixel-to-pixel comparisons, are computationally expensive and have been outperformed by the Bag-of-Visual-words (BoV) model, which is based on the construction of a visual vocabulary, known also as visual codebook, using a vocabulary of visual words [17], by clustering all visual features. The visual vocabulary construction is motivated by the Bag-of-Words (BoW) model in a collection of text documents. The set of all visual descriptors in an image collection is clustered using k-means clustering techniques, which are replaced by approximate k-means methods [15], in order to reduce the computational cost of visual vocabulary construction. However, both k-means techniques require as input the number of visual words k, which we shall estimate incrementally.

The visual vocabulary is usually constructed, using an empirical number of visual words k, such as $k = 4000$ in [18]. The optimal number k is hard to be tuned in very large databases, and impossible when ground truth does not exist. An empirical guess of k may lead to the construction of visual codebooks, which are not optimal when involved in an image retrieval of image clustering task. To that end, we propose a scalable estimation of the optimal number of visual words k in an incremental way, using a recent modification of DBSCAN [3], which also has a scalable and parallel implementation [7]. In the proposed framework, the final number of visual words is incrementally estimated on a sample of images and therefore, it can easily scale up to very large image collections, in the context of Big Data.

The most prominent visual features are SIFT descriptors [9], but several other methods have been proposed to represent an image, such as VLAD [8], GIST [12], Fisher vectors [13] or DCNN features [10]. In this work, we will restrict our study on the estimation of the visual vocabulary size, based on SIFT descriptors, and comparison in terms of the optimal visual features is beyond the scope of this work.

The main research contributions of this work are:

- Estimate the optimal size of a visual vocabulary
- Build the size estimation incrementally

Therefore, we are able to build efficient visual vocabularies without tuning the size or guessing a value for k. Our proposed method is a hybrid framework, which combines the recent DBSCAN-Martingale [6] and k-means clustering. The proposed hybrid framework is evaluated in the image retrieval and image clustering problems, where we initially provide an estimation for the number of visual words k, using the DBSCAN-Martingale, and then cluster all visual descriptors by k, as traditionally done by k-means clustering.

In Sect. 2 we present the related work in visual vocabulary construction and in Sect. 3 we briefly present the DBSCAN-Martingale estimator of the number of clusters. In Sect. 4, our proposed hybrid method for the construction of visual vocabularies is described in detail, and finally, in Sect. 5, it is evaluated under the image retrieval and image clustering tasks.

2 Related Work

The Bag-of-Visual-Words (BoV) model initially appeared in [17], in which k-means clustering is applied for the construction of a visual vocabulary. The constructed visual vocabulary is then used for image retrieval purposes and is similar to the Bag-of-Words model, where a vocabulary of words is constructed, mainly for text retrieval, clustering and classification. In the BoV model, the image query and each image of the collection are represented as a sparse vector of term (visual word) occurrences, weighted by tf-idf scores. The similarity between the query and each image is calculated, using the Mahalanobis distance or simply the Euclidean distance. However, there is no obvious value for the number of clusters k in the k-means clustering algorithm.

Other approaches for the construction of visual vocabularies include Approximate k-means (AKM) clustering, which offers scalable construction of visual vocabularies. Hierarchical k-means (HKM) was the first approximate method for fast and scalable construction of a visual vocabulary [11], where data points are clustered by $k = 2$ or $k = 10$ using k-means clustering and then k-means is applied to each one of the newly generated clusters, using the same number of clusters k. After n steps (levels), the result is k^n clusters. HKM has been outperformed by AKM [15], where a forest of 8 randomized k-d trees provides approximate nearest neighbor search between points and the approximately nearest cluster centers. The use of 8 randomized k-d trees with skewed splits have recently been proposed, in the special case of SIFT descriptors [5]. However, all AKM clustering methods require as input the number of clusters k, so an efficient estimation of k is necessary.

The need to estimate the number of visual words emerges from the computational cost of k-means algorithm, either in exact or approximate k-means clustering [19]. Apart from being a time consuming process, tuning the number of clusters k may affect significantly the performance of the image retrieval task [15]. Some studies assume a fixed value of k, such as $k = 4000$ in [18], but in general the choice of k varies from 10^3 up to 10^7, as stated in [12]. In another approach, 10 clusters are extracted using k-means for each one of the considered classes (categories), which are then concatenated in order to form a global visual vocabulary [20]. In contrast, we shall estimate the number of clusters using the DBSCAN-Martingale [6], which automatically estimates the number of clusters, based on an extension of DBSCAN [3], without a priori knowledge of the density parameter $minPts$ of DBSCAN. DBSCAN-Martingale generates a probability distribution over the number of clusters and has been applied to news clustering, in combination with LDA [6]. Contrary to the previous text clustering approach, DBSCAN-Martingale shall be presented in the context of visual vocabulary construction for the estimation of the optimal number of visual words, to support any image retrieval or clustering task.

3 The DBSCAN-Martingale Estimation of the Number of Clusters

In this section, we briefly describe the DBSCAN-Martingale, which has been introduced for the estimation of the number of clusters in a collection of text documents. DBSCAN [3] uses two parameters ϵ and $minPts$ to cluster the points of a dataset without knowing the number of clusters. DBSCAN-Martingale overcomes the tuning of the parameter ϵ and shows robustness to the variation of the parameter $minPts$ [6].

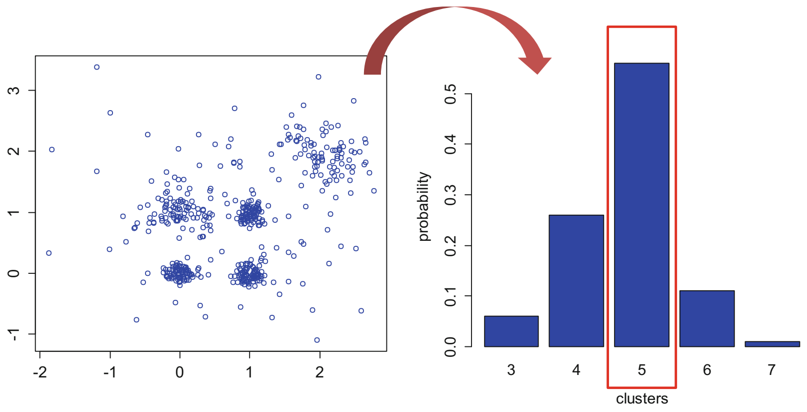


Fig. 1. The number of clusters as estimated by the DBSCAN-Martingale on an illustrative dataset. The generated probability distribution states that it is more likely to have 5 clusters, although they appear in different density levels and there are points which do not belong to any of the clusters.

The estimation of the number of clusters is a probabilistic method and assigns a probability distribution over the number of clusters, so as to extract all clusters for all density levels. For each randomly generated density level ϵ, density-based clusters are extracted using the DBSCAN algorithm. The density levels $\epsilon_t, t = 1, 2, \ldots, T$ are generated from the uniform distribution in the interval $[0, \epsilon_{max}]$ and sorted in increasing order.

Each density level ϵ_t provides one partitioning of the dataset, which then formulates a $N \times 1$ clustering vector, namely $C_{DBSCAN(\epsilon_t)}$ for all stages $t = 1, 2, \ldots, T$, where N is the number of points to cluster. The clustering vector takes as value the cluster ID κ of the j-th point, i.e. $C_{DBSCAN(\epsilon_t)}[j] = \kappa$.

In the beginning of the algorithm, there are no clusters detected. In the first stage ($t = 1$), all clusters detected by $C_{DBSCAN(\epsilon_1)}$ are kept, corresponding to the lowest density level ϵ_1. In the second stage ($t = 2$), some of the detected clusters by $C_{DBSCAN(\epsilon_2)}$ are new and some of them have also been detected at previous stage ($t = 1$). DBSCAN-Martingale keeps only the newly detected clusters of the second stage ($t = 2$), by grouping the numbers of the same cluster

ID with size greater than $minPts$. After T stages, we have progressively gained knowledge about the final number of clusters $\hat{k}$, since all clusters have been extracted with high probability.

The estimation of number of clusters $\hat{k}$ is a random variable, because of the randomness of the generated density levels $\epsilon_t, t = 1, 2, \ldots, T$. For each realization of the DBSCAN-Martingale one estimation $\hat{k}$ is generated, and the final estimation of the number of clusters have been proposed [6] as the majority vote over 10 realizations of the DBSCAN-Martingale. The percentage of realizations where the DBSCAN-Martingale outputs exactly $\hat{k}$ clusters is a probability distribution, as the one shown in Fig. 1.

4 Estimation of the Visual Vocabulary Size Using the DBSCAN-Martingale

Motivated by the DBSCAN-Martingale, which has been applied in several collections of text documents in the context of news clustering [5], we propose an estimation of the total number of visual words in an image collection, as shown in Fig. 2. The proposed method is incremental, since the estimation of the final number of visual words is progressively estimated and updated when a new image is added to the collection.

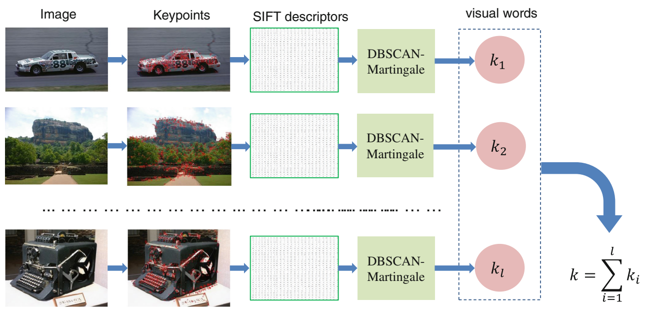


Fig. 2. The estimation of the number of visual words in an image collection. Each image i contributes with k_i visual words to the overall estimation of the visual vocabulary size.

Starting from the first image, keypoints are detected and SIFT descriptors [9] are extracted. Each visual feature is represented as a 128-dimensional vector, hence the whole image i is a matrix M_i with 128 columns, but the number of rows is subject to the number of detected keypoints. On each matrix M_i, the 128-dimensional vectors are clustered using the DBSCAN-Martingale, which outputs the number of dense patterns in the set of visual features, as provided by several density levels. Assuming that the application of 100 realizations of the DBSCAN-Martingale has output k_1 for the first image, k_2 for the second image and k_l for the l-th image (Fig. 2), the proposed optimal size of the visual vocabulary is:

$$k = \sum_{i=1}^{l} k_i \tag{1}$$

DBSCAN-Martingale extracts clusters sequentially, combines them into one single clustering vector and outputs the most updated estimation of the number of clusters, in each realization. The DBSCAN-Martingale requires T iterations of the DBSCAN algorithm, which runs in $\mathscr{O}(n \log n)$, when kd-tree data structures are employed for fast nearest neighbor search and in $\mathscr{O}(n^2)$ without tree-based spatial indexing [1]. Approximate solutions of DBSCAN [4] may reduce its computational complexity up to $\mathscr{O}(n)$, without significant accuracy reduction. We adopt the implementation of DBSCAN-Martingale in R[1], which is available on Github[2], because the R-script utilizes the dbscan[3] package, which runs DBSCAN in $\mathscr{O}(n \log n)$. Thus, the overall complexity of the DBSCAN-Martingale is $\mathscr{O}(Tn \log n)$, where n is the number of visual descriptors per image. Assuming r iterations of the DBSCAN-Martingale per image and given an image collection of l images, the overall estimation of the size of a visual vocabulary is $\mathscr{O}(lrTn \log n)$.

In order to reduce the complexity, we sample l' out of l images to get an average number of visual words per image. The final number of visual words is estimated from a sample of images $S = \{i_1, i_2, \ldots, i_{l'}\}$ of size l', so the overall complexity becomes $\mathscr{O}(l'rTn \log n)$. The final estimation for the number of visual words $\hat{k}$ of Eq. (1) becomes:

$$\hat{k} = \frac{l}{l'} \sum_{i \in S} k_i \tag{2}$$

We utilize the estimation $\hat{k}$, provided by Eq. (2), in order to cluster all visual features by $\hat{k}$ using k-means clustering. Therefore, a visual vocabulary of size $\hat{k}$ is constructed. After the construction of a visual vocabulary, as shown in Fig. 3, images are represented using term-frequency scores with inverse document frequency weighting (tf-idf) [17]:

$$\text{tfidf}_{ij} = \frac{n_{id}}{n_d} \log \frac{D}{n_i} \tag{3}$$

where n_{id} is the number of occurrences of visual word i in image d, n_d is the number of visual words in image d, n_i is the number of occurrences of visual word i in the whole image collection and D is the total number of images in the database.

In the following section, we test our hybrid visual vocabulary construction in the image retrieval and image clustering problems.

[1] https://www.r-project.org/.
[2] https://github.com/MKLab-ITI/topic-detection/blob/master/DBSCAN_Martingale.r.
[3] https://cran.r-project.org/web/packages/dbscan/index.html.

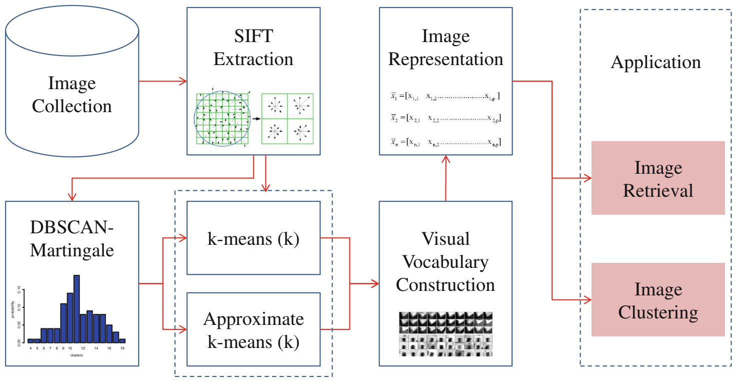


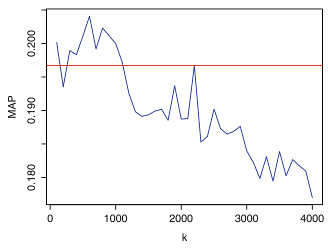
Fig. 3. The hybrid visual vocabulary construction framework using the DBSCAN-Martingale for the estimation of k and either exact or approximate k-means clustering by k. After the visual vocabulary is constructed, the collection of images is efficiently represented for any application, such as image retrieval or clustering.

5 Experiments

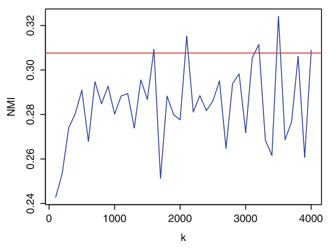
We evaluate our method in the image retrieval and image clustering tasks, in which nearest neighbor search is performed in an unsupervised way. The datasets we have selected are the WANG[4] 1K and Caltech[5] 2.5 K with 2,516 images, with queries as described in [5] for the image retrieval task. The number of extracted visual descriptors (SIFT) is 505,834 and 769,546 128-dimensional vectors in the WANG 1K and Caltech 2.5 K datasets, respectively. The number of topics is 10 for the WANG dataset and 21 for the Caltech, allowing also image clustering experiments with the considered datasets. We selected these datasets because they are appropriate for performing both image retrieval and image clustering experiments and tuning the number of visual words k may be done in reasonable processing time, so as to evaluate the visual vocabulary construction in terms of Mean Average Precision (MAP) and Normalized Mutual Information (NMI).

Keypoints are detected and SIFT descriptors are extracted using the LIP-VIREO toolkit[6]. For the implementation of DBSCAN-Martingale we used the R-script, which is available on Github[7] for $\epsilon_{max} = 200$ and 100 realizations. We build one visual vocabulary for several number of visual words k, which is tuned in $k \in \{100, 200, 300, \ldots, 4000\}$. The parameter $minPts$ is tuned from 5 to 30 and the final number of clusters per image is the number which is more robust to the variations of $minPts$. In k-means clustering, we allow a maximum of 20 iterations.

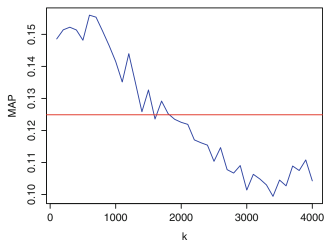
4 http://wang.ist.psu.edu/docs/related/.
5 http://www.vision.caltech.edu/Image_Datasets/Caltech101/.
6 http://pami.xmu.edu.cn/~wlzhao/lip-vireo.htm.
7 https://github.com/MKLab-ITI/topic-detection/blob/master/DBSCAN_Martingale.r.



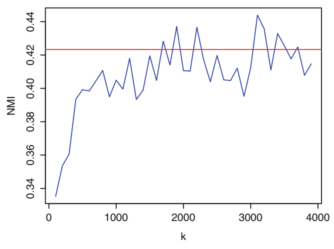
(a) MAP for the WANG dataset



(b) NMI for the WANG dataset



(c) MAP for the Caltech dataset



(d) NMI for the Caltech dataset

Fig. 4. Evaluation using MAP and NMI in image retrieval and image clustering tasks for the WANG and Caltech datasets. The MAP and NMI scores which are obtained by our k estimation is the straight red line.

Our estimations for the number of visual words is $\hat{k} = 2180$ and $\hat{k} = 1840$ for the WANG and Caltech datasets, respectively, given a sample of 200 images. The corresponding MAP and NMI are compared to the best MAP and NMI scores in $k \in \{100, 200, 300, \dots, 4000\}$. The results are reported in Table 1, where apart from the best MAP and NMI scores, we also present the ratio of MAP (NMI) provided by our $\hat{k}$-estimation to the maximum observed MAP (NMI) score, denoted by r_{MAP} (r_{NMI}). In particular, in the WANG dataset, MAP is 96.42 % of the best MAP observed and NMI is 94.91 % of the best NMI. Similar behavior is observed in the Caltech dataset, where NMI is approached at 95.36 % and MAP at 80.06 %, respectively. In Fig. 4 we observe that our

Table 1. Evaluation in image retrieval and image clustering tasks.

Dataset	k visual words	MAP	r_{MAP}	NMI	r_{NMI}
WANG	best k	0.2040		0.3241	
	DBSCAN-Martingale $\hat{k}$	0.1967	0.9642	0.3076	0.9491
Caltech	best k	0.1560		0.4439	
	DBSCAN-Martingale $\hat{k}$	0.1249	0.8006	0.4233	0.9536

incremental estimation method, when combined with k-means, approaches the highest observed MAP and NMI scores in all cases examined.

6 Conclusion

We presented an incremental estimation of the optimal size of the visual vocabulary, which efficiently estimates the number of visual words, and evaluated the performance of the constructed visual vocabulary in an image retrieval and an image clustering task. Increasing the visual vocabulary size does not guarantee that the performance of the corresponding retrieval and clustering tasks will also increase as shown in Fig. 4 (c) for the Mean Average Precision. Our proposed hybrid framework utilizes the output of DBSCAN-Martingale on a sample of SIFT descriptors, to estimate the visual vocabulary size, in order to be used as input in any k-means or approximate k-means clustering for the construction of a visual vocabulary. The estimation is incremental and can be distributed to several threads. A potential limitation of our approach could appear in the case where an image exists more than once in the image collection and therefore needlessly contributes with extra visual words the final estimation. However, if the sample of images on which the DBSCAN-Martingale is applied does not have duplicate images, the overall estimation will not be affected. In the future, we plan to test our method using other visual features and in the context of multimedia retrieval, where multiple modalities are employed.

Acknowledgements. This work was supported by the project MULTISENSOR (FP7-610411), funded by the European Commission.

References

1. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: OPTICS: ordering points to identify the clustering structure. ACM Sigmod Rec. **28**(2), 49–60 (1999)
2. Devroye, L.: Sample-based non-uniform random variate generation. In: Proceedings of the 18th Conference on Winter Simulation, pp. 260–265. ACM, December 1986
3. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. Kdd 96(34), 226–231 (1996)
4. Gan, J., Tao, Y.: DBSCAN revisited: mis-claim, un-fixability, and approximation. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp. 519–530. ACM, May 2015
5. Gialampoukidis, I., Vrochidis, S., Kompatsiaris, I.: Fast visual vocabulary construction for image retrieval using skewed-split kd trees. In: MultiMedia Modeling, pp. 466–477. Springer International Publishing, January 2016
6. Gialampoukidis, I., Vrochidis, S., Kompatsiaris, I.: A Hybrid framework for news clustering based on the DBSCAN-Martingale and LDA. In: Machine Learning and Data Mining in Pattern Recognition, pp. 170–184. Springer International Publishing, July 2016
7. He, Y., Tan, H., Luo, W., Feng, S., Fan, J.: MR-DBSCAN: a scalable MapReduce-based DBSCAN algorithm for heavily skewed data. Front. Comput. Sci. **8**(1), 83–99 (2014)

8. Jégou, H., Douze, M., Schmid, C., Pérez, P.: Aggregating local descriptors into a compact image representation. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3304–3311. IEEE, June 2010
9. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Intl. J. Comput. Vis. **60**(2), 91–110 (2004)
10. Markatopoulou, F., Mezaris, V., Patras, I.: . Cascade of classifiers based on binary, non-binary and deep convolutional network descriptors for video concept detection. In: 2015 IEEE International Conference on Image Processing (ICIP), pp. 1786–1790. IEEE, September 2015
11. Mikolajczyk, K., Leibe, B., Schiele, B.: Multiple object class detection with a generative model. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 26–36. IEEE, June 2006
12. Mikulik, A., Chum, O., Matas, J.: Image retrieval for online browsing in large image collections. In: Brisaboa, N., Pedreira, O., Zezula, P. (eds.) SISAP 2013. LNCS, vol. 8199, pp. 3–15. Springer, Heidelberg (2013). doi:10.1007/978-3-642-41062-8_2
13. Perronnin, F., Sánchez, J., Mensink, T.: Improving the fisher kernel for large-scale image classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 143–156. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15561-1_11
14. Philbin, J.: Scalable object retrieval in very large image collections. Doctoral dissertation, Oxford University (2010)
15. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: IEEE Conference on Computer Vision and Pattern Recognition, 2007, CVPR 2007, pp. 1–8. IEEE, June 2007
16. Rawlings, J.O., Pantula, S.G., Dickey, D.A.: Applied regression analysis: a research tool. Springer Science & Business Media (1998)
17. Sivic, J., Zisserman, A.: Video Google: a text retrieval approach to object matching in videos. In: Ninth IEEE International Conference on Computer Vision, 2003, Proceedings, pp. 1470–1477. IEEE, October 2003
18. Van De Sande, K.E., Gevers, T., Snoek, C.G.: Evaluating color descriptors for object and scene recognition. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9), 1582–1596 (2010)
19. Wang, J., Wang, J., Ke, Q., Zeng, G., Li, S.: Fast approximate k-means via cluster closures. In: Multimedia Data Mining and Analytics, pp. 373–395. Springer International Publishing (2015)
20. Zhang, J., Marszalek, M., Lazebnik, S., Schmid, C.: Local features and kernels for classification of texture and object categories: a comprehensive study. Intl. J. Comput. Vis. **73**(2), 213–238 (2007)

Attribute Learning for Network Intrusion Detection

Jorge Luis Rivero Pérez(✉) and Bernardete Ribeiro

CISUC - Department of Informatics Engineering, University of Coimbra, Coimbra, Portugal
{jlrivero,bribeiro}@dei.uc.pt

Abstract. Network intrusion detection is one of the most visible uses for Big Data analytics. One of the main problems in this application is the constant rise of new attacks. This scenario, characterized by the fact that not enough labeled examples are available for the new classes of attacks is hardly addressed by traditional machine learning approaches. New findings on the capabilities of Zero-Shot learning (ZSL) approach makes it an interesting solution for this problem because it has the ability to classify instances of unseen classes. ZSL has inherently two stages: the attribute learning and the inference stage. In this paper we propose a new algorithm for the attribute learning stage of ZSL. The idea is to learn new values for the attributes based on decision trees (DT). Our results show that based on the rules extracted from the DT a better distribution for the attribute values can be found. We also propose an experimental setup for the evaluation of ZSL on network intrusion detection (NID).

Keywords: Attribute Learning · Network Intrusion Detection · Zero-Shot Learning

1 Introduction

Nowadays society is becoming increasingly dependent on the use of computer systems in various fields such as finance, security and many aspects of everyday life. On the other hand threats and attacks are increasingly growing. Cyber-Security research area looks at the ability to act pro-actively in order to mitigate or prevent attacks. In that sense Network Intrusion Detection (NID) is one of the (possible) solutions. This task is basically carried out under two approaches: (i) Missuse Detection and (ii) Anomaly Detection. These approaches have advantages and disadvantages associated with their suitability to various scenarios [1]. There are machine learning based solutions to these approaches [2,3]. Despite the extensive academic research, their deployment in operational NID environments has been very limited [4]. On the other hand new attacks are constantly occurring, often as variants of known attacks.

Traditional machine learning approaches are unable to tackle challenging scenarios in which new classes may appear after the learning stage. This scenario is present many real world situations [5]. Specifically in NID related tasks where one of the main problems is the emergence of new attacks which corresponds with new classes. In that case the detection algorithm should identify the new classes which is important in real environments. Recently, there has been an increasing interest in the study of ZSL approaches, which might be a possible solution to this problem [5–7]. In this paper we propose an Attribute Learning for Network Intrusion Detection (ALNID) algorithm. We present the preliminary results as an initial step prior to the detection of new attacks on networks. A proposal of the experimental data setup for the application of ZSL in NID is also given. The proposed attribute learning algorithm can be generalized to many problems. It is simple and based on criteria such as attributes frequency and entropy, achieving to learn new values for the original attributes. The results show a significant improvement in the representation of the attributes for classes. This is encouraging since we expect to achieve higher accuracy in the inference stage of ZSL. The rest of the paper is organized as follows. In Sect. 2 we briefly review the background, related work on ZSL as well as the attribute learning prior stage. In Sect. 3 we describe the ALNID algorithm. In Sect. 4 we present the data preprocessing and propose an experimental data setup for the application of ZSL in NID. In Sect. 5 we discuss the results on KDD Cup 99 dataset. Finally, we conclude and address some lines of future work.

2 Zero-Shot Learning

ZSL is inspired by the way human beings are capable of identifying new classes when a high-level description is provided. It consists of identifying new classes without training examples, from a high-level description of the new classes (unseen classes) that relate them to classes previously learned during the training (seen classes). This is done by learning the attributes as an intermediate layer that provides semantic information about the classes to classify. ZSL has two stages: the training or attribute learning stage in which knowledge about the attributes is captured, and then the inference stage where this knowledge is used to classify instances among a new set of classes. This approach has been widely applied to images classification tasks [5–8]. There are different solutions for both attribute learning and for the inference stage, which models the relationships among features, attributes, and classes in images. Such strategy makes it difficult to apply ZSL to other problems. This, coupled with the need of identify new attacks on computer networks motivated us to development this research.

2.1 Attribute Learning Stage

The attribute learning focuses on learning to recognize several properties from objects, which allow to learn new classes based only on their description [5]. Recently there have been applications of automatic recognition of attributes on images [8–10]. In [8] the authors propose the extensive Scene UNderstanding (SUN) database that contains 899 categories and $130,519$ images. The database is built by human scene classification using Amazon Mechanical Turk (AMT). The authors firstly defined the attributes and then collected votes from AMT evaluating the presence or not of the previously defined attributes on each image. The procedure then selected or designed several state-of-art features that are potentially useful for scene classification. To the best of our knowledge this approach has never been applied on network intrusion detection.

2.2 Inference Stage

In the inference stage the predicted attributes are combined to infer the classes. There are basically three approaches [5] for this second stage: k–nearest neighbour (k–NN), probabilistic frameworks [7] and energy function [5,6]. One interesting technique is the cascaded probabilistic framework proposed in [7,10] where the predicted attributes on the first stage are combined to determine the most likely target class. It has two variants: the Directed Attribute Prediction (DAP), in which during the training stage, for each attribute, a probabilistic classifier is learned. Then, at inference stage the classifiers are used to infer new classes from their attributes signatures. The other variant is the Indirected Attribute Prediction (IAP) which learns a classifier for each training class and then combines predictions of training and test classes. The third ZSL approach uses energy function [5,6], which is the same as the penalty function where given x: data, and v: category vector, the energy function $E_w(x, v) = x'Wv$ is trained, with the metric W being $E_w(x, v)$ positive if $x = v$ and negative if $x! = v$. In [11] a framework is proposed to predict both seen and unseen classes. Unlike other approaches this proposal not only classifies unknown classes but also classifies known classes. In the attribute learning stage they consider the set of all classes Y during training and testing. Some classes y are available as seen classes in training data (Y_s) and the others are the Zero-Shot classes, without any training data, as unseen classes (Y_u). Then they define $W = W_s \cup W_u$ as the word vectors for both, seen and unseen classes. All training images $x^{(i)} \in X_y$ of a seen class $y \in Y_s$ are mapped to the word vector w_y. Then, to train this mapping a two-layer neural network to minimize the following objective function (Eq. 1) is trained [11]:

$$J(\Theta) = \sum_{y \in Y_s} \sum_{x^{(i)} \in X_y} \left\| w_y - \theta^{(2)} f(\theta^{(1)} x^{(i)}) \right\|^2 \tag{1}$$

Other recent and simple energy function based approach is the ESZSL proposal [5]. It is based on [6] which models the relationships among features, attributes, and classes. The authors assume that at training stage there are z classes, which have a signature composed of a attributes. That signatures are represented in a matrix $S \in [0, 1]^{axz}$. This matrix contains for each attribute any value in $[0, 1]$ representing the relationship between each attribute and the classes. However, how the matrix S is computed is not addressed. The instances available at training stage are denoted by $X \in \mathbb{R}^{dxm}$, where d is the dimensionality of the data, and m is the number of instances. The authors also compute the matrix $Y \in \{-1, 1\}^{mxz}$ to denote to which class belongs each instance. During the training they compute the matrix $V \in \mathbb{R}^{dxa}$ as in Eq. 2:

$$V = (XX^T + \gamma I)^{-1} XYS(SS^T + \lambda I)^{-1} \tag{2}$$

In inference stage they distinguish between a new set of z' classes by their attributes signatures, $S' \in [0, 1]^{axz'}$. Then, given a new instance x the prediction is given by Eq. 3:

$$\arg\max_i (x^T V S'_i) \tag{3}$$

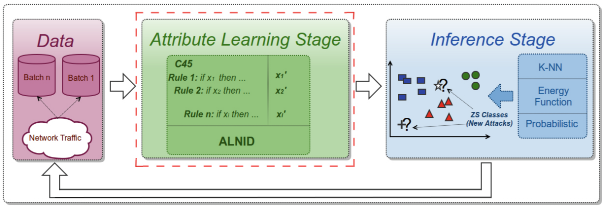


Fig. 1. Zero-Shot Learning for Network Intrusion Detection

3 Proposed Attribute Learning Algorithm

Most of the attribute learning algorithms have been implemented for automatic recognition of attributes on images based on the features extraction [8–10]. It is hardly applicable to other kind of problems like NID. Firstly we evaluate different machine learning algorithms on the preprocessed dataset (which is described in next section) and the C45 decision tree algorithm shows the higher classification accuracy of 99.54 %. This result together the lack of attribute learning algorithms for non computer vision related tasks leads us to think in the extracted rules as a solution to build or relearn the attributes. This method could be extended not only to cover a wider area of Information Technology security but also to different kind of problems with labeled and structured data in which the C45 algorithm could be applied with good results.

Our proposal is for the Attribute Learning Stage of the ZSL approach depicted in Fig. 1. ALNID - Attribute Learning for Network Intrusion Detection Algorithm - is a rule-based algorithm which weights the attributes according its entropy and frequency in the rules. We begin with a set of X instances composed of $A = \{a_1, \cdots, a_n\}$ attributes. For each $a_n \in A$ we compute the quantity of information (I), the entropy (E) and the information gain (G). With these values we compute the C45 algorithm for decision trees. Furthermore, during each iteration we record how many times each $a_i \in A$ attribute is evaluated by each rule of the set $R = \{r_1, \cdots, r_m\}$. This is what we call frequency. Then, a new set of attributes $A' = \{a'_1, \cdots, a'_n\}$ is created. The values of A' are the frequency count, increasing by each time that an attribute is evaluated by the rule r_m. As output the algorithm returns the set of new valued instances $X' = \{A'_i\}_{i=1}^{N}$ composed of the learned attributes, where N is the number of instances. The pseudo-code of the proposed method is listed below:

```
{attributes weighted w.r.t. their frequency in each rule match};
X: Examples, A: Attributes (Input)
X': Examples with learned attributes (Output)
var
    I: Quantity of information
    E: Entropy
    G: Information Gain
begin
    if all examples are from the same class then
        repeat
            Weight attribute;
            Add examples to X'
        until attribute in A
        return X';
     else
         I = Compute quantity of information of the examples;
         repeat
             E = Compute the entropy of attribute;
             G = Compute the information gain of attribute;
         until attribute in A
         a = attribute that maximizes G;
         v = value of a;
         Delete a from A;
         Generate a root node for the attribute a;
         Weight attribute;
         Add examples to X'
         repeat
             ALNID (examples from X with a == v, A);
             generate a new branch with a==v;
         until partition in Partitions generated by values of attribute a
         return X';
end.
```

4 Data and Experimental Setup

KDD Cup 99[1] is the most used in research as data to test and compare intrusion detection algorithms. It contains about 5 millions of instances representing 39 types of attacks which are grouped into 4 categories and also one category for normal traffic. Some preprocessing variants applied on the dataset are based on the reduction of the classes replacing the attack labels by their corresponding

[1] KDD Cup is the annual Data Mining and Knowledge Discovery competition organized by ACM Special Interest Group on Knowledge Discovery and Data Mining.

Table 1. Data setup for zero-shot learning on network intrusion detection

Class	Nr. of Examples	Zero-Shot Class	Category
smurf	280, 790	no	DOS (D)
neptune	107, 201	no	DOS (D)
back	2203	no	DOS (D)
teardrop	979	yes	DOS (D)
pod	264	no	DOS (D)
land	21	yes	DOS (D)
normal	97, 277	no	NORMAL (N)
satan	1589	no	PROBE (P)
ipsweep	1247	yes	PROBE (P)
portsweep	1040	no	PROBE (P)
nmap	231	yes	PROBE (P)
warezclient	1020	no	R2L (R)
guess_passwd	53	yes	R2L (R)
warezmaster	20	no	R2L (R)
imap	12	yes	R2L (R)
ftp_write	8	no	R2L (R)
multihop	7	no	R2L (R)
phf	4	no	R2L (R)
spy	2	no	R2L (R)
buffer_overflow	30	no	U2R (U)
rootkit	10	yes	U2R (U)
loadmodule	9	no	U2R (U)
perl	3	yes	U2R (U)

categories, thus reducing the number of classes to 5 [1]. Each instance represents a TCP/IP connection composed of 41 attributes both quantitative and qualitative[2]. Despite our proposal holds for the attribute learning stage, we propose herein a scheme to apply ZSL approach in NID tasks (see Fig. 1). Table 1 shows that classes such as *spy* and *perl* have 2 and 3 instances respectively while classes as *smurf* and *normal* have 280,790 and 97,277 respectively. Then, considering the study in [1] we selected the 12 attributes from the original 41 ones. Its statistical description are shown in Table 2. We modified the dataset using the five categories as classes. Later, for each category we selected two attacks as Zero-Shot classes. Table 1 illustrates these values and the Zero-Shot classes. This setup is very practical for the application at hand because we can classify

[2] Current research uses a small portion that represents the 10 % of the original dataset containing 494, 021 instances.

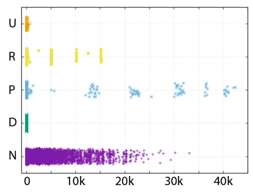
Table 2. Statistical description of the attributes

Attributes	Minimum	Maximum	Mean	StdDev
duration	0	58,329	47.979	707.747
duration'	0	3	0.013	0.117
protocol_type	1	3	2.189	0.961
protocol_type'	0	1	0.845	0.362
src_bytes	0	693,375,640	3025.616	988,219.101
src_bytes'	0	7	0.762	1.526
dst_bytes	0	5155,468	868.531	33,040.035
dst_bytes'	0	3	0.057	0.297
urgent	0	3	0	0.006
urgent'	0	1	0	0.016
count	0	511	332.286	213.147
count'	1	4	1.821	0.425
srv_count	0	511	292.907	246.323
srv_count'	0	2	0	0.017
same_srv_rate	0	1	0.792	0.388
same_srv_rate'	0	1	0.784	0.411
dst_host_count	0	255	232.471	64.745
dst_host_count'	0	2	0.033	0.238
dst_host_srv_count	0	255	188.666	106.04
dst_host_srv_count'	0	3	0.21	0.469
dst_host_same_srv_rate	0	1	0.754	0.411
dst_host_same_srv_rate'	0	1	0.024	0.154
dst_host_same_src_port_rate	0	1	0.602	0.481
dst_host_same_src_port_rate'	0	3	0.585	0.779

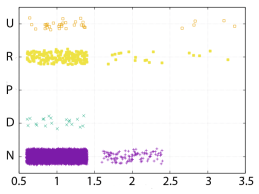
the Zero-Shot classes. In this case, new attacks can be classified in the categories to which they belong to.

5 Results and Discussion

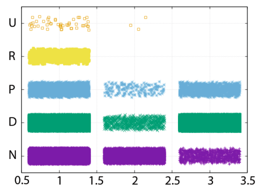
The ALNID algorithm was evaluated on the preprocessed dataset computing a DT of 132 leaves and 263 rules ($R = \{r_1, ..., r_{263}\}$). The classification accuracy for the seen classes was 99.94 %. Table 2 summarizes the minimum, maximum, mean and standard deviation values for each of the original attributes and the learned ones by our proposal. The Fig. 2 depicts the distribution per class of the original attribute values ($A = \{a_1, ..., a_{12}\}$) and the learned ones ($A' = \{a'_1, ..., a'_{12}\}$). These were listed by decreasing order of the entropy (E) of the learned attributes. In general the graphical plots show better separated



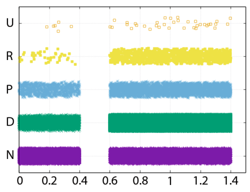
(a) duration



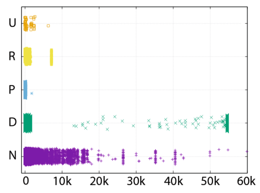
(b) *duration'*



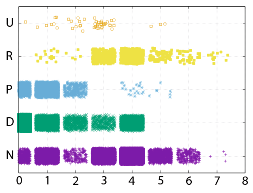
(c) protocol_type



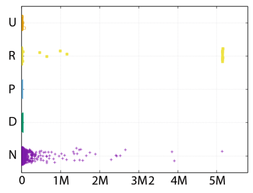
(d) *protocol_type'*



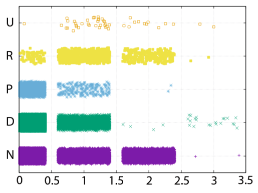
(e) src_bytes



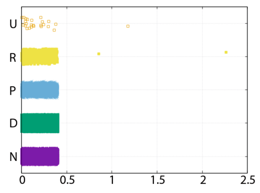
(f) *src_bytes'*



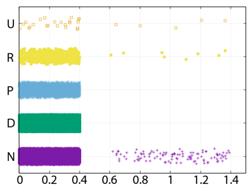
(g) dst_bytes



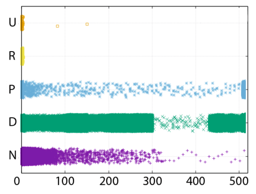
(h) *dst_bytes'*



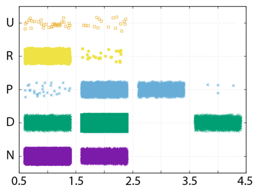
(i) urgent



(j) *urgent'*

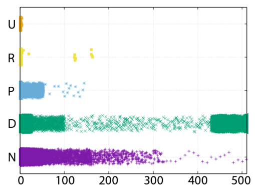


(k) count

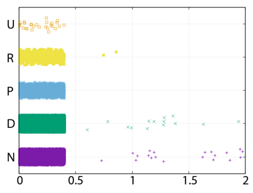


(l) *count'*

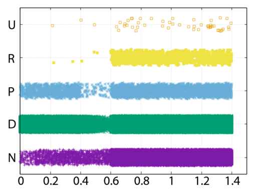
Fig. 2. Original (left) and learned (right) attributes.



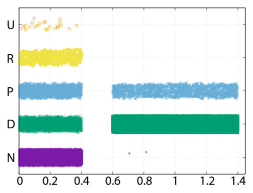
(m) srv_count



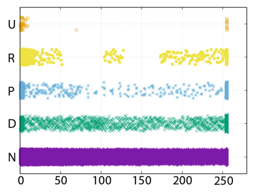
(n) *srv_count'*



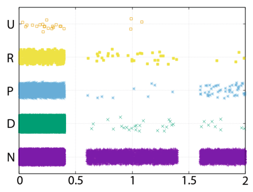
(o) sam_srv_rate



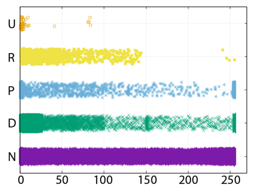
(p) *sam_srv_rate'*



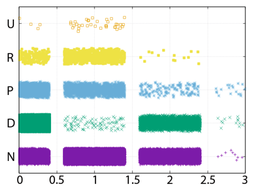
(q) dst_host_count



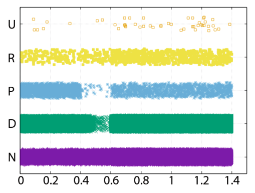
(r) *dst_host_count'*



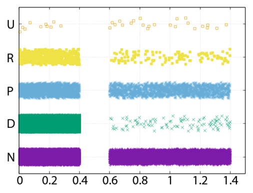
(s) dst_host_srv_count



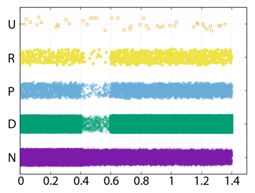
(t) *dst_host_srv_count'*



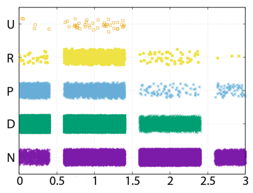
(u) dst_host_same_srv_rate



(v) *dst_host_same_srv_rate'*



(w) dst_host_same_port_rate



(x) *dst_host_same_port_rate'*

Fig. 2. (*continued*)

distributions per class for the learned attributes with our proposal than for the originals ones. The learned attribute *duration'* in Fig. 2(a) and (b) improves distribution at least w.r.t. one class, e.g. the NORMAL (N) one. The rest of the learned attributes by ALNID achieves a higher separability than the original ones in their distribution regarding the classes (see Fig. 2(c)–(x)). This new representation of the learned attributes are expected to improve the k–NN classification during the inference stage. Also we found a way to compute the signature matrix $S \in [0,1]^{axz}$ required by the energy function used in the inference approach in [5]. The range of values of the learned attributes is discrete which can easily be integrated in the inference stage.

6 Conclusions

ZSL is a two-stage approach that addresses the classification of new classes without any example during the training. This, joined with the need to detect new attacks on traffic networks motivated us to research in the application of ZSL to NID. In this paper we proposed ALNID, a new algorithm for the attribute learning stage of ZSL. The algorithm builds a DT and combines entropy and frequency of the original attributes in a weighted function setting. Our evaluation proposal on KDD Cup 99 dataset showed better distribution of the attribute values per class. The class separability is convenient for the inference stage based on k–NN. Future work will extend the work to the inference stage based on the learned attributes by this proposal and will validate ZSL to other NID Big Data.

References

1. Rivero Pérez, P.L.: Técnicas de aprendizaje automático para la detección de intrusos en redes de computadoras. Revista Cubana de Ciencias Informáticas **8**(4), 52–73 (2014)
2. Sangkatsanee, P., Wattanapongsakorn, N., Charnsripinyo, C.: Practical real-time intrusion detection using machine learning approaches. Comput. Commun. **34**(18), 2227–2235 (2011)
3. Tsai, C.F., Hsu, Y.F., Lin, C.Y., Lin, W.Y.: Intrusion detection by machine learning: a review. Expert Syst. Appl. **36**(10), 11994–12000 (2009)
4. Sommer, R., Paxson, V.: Outside the closed world: on using machine learning for network intrusion detection. In: 2010 IEEE Symposium on Security and Privacy (SP), pp. 305–316. IEEE (2010)
5. Romera-Paredes, B., Torr, P.: An embarrassingly simple approach to zero-shot learning. In: Proceedings of The 32nd International Conference on Machine Learning, pp. 2152–2161 (2015)
6. Akata, Z., Perronnin, F., Harchaoui, Z., Schmid, C.: Label-embedding for attribute-based classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 819–826 (2013)
7. Lampert, C.H., Nickisch, H., Harmeling, S.: Attribute-based classification for zero-shot visual object categorization. IEEE Trans. Pattern Anal. Mach. Intell. **36**(3), 453–465 (2014)

8. Patterson, G., Xu, C., Su, H., Hays, J.: The sun attribute database: beyond categories for deeper scene understanding. Int. J. Comput. Vis. **108**(1–2), 59–81 (2014)
9. Ferrari, V., Zisserman, A.: Learning visual attributes. In: NIPS, pp. 433–440 (2007)
10. Lampert, C.H., Nickisch, H., Harmeling, S.: Learning to detect unseen object classes by between-class attribute transfer. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, pp. 951–958. IEEE (2009)
11. Socher, R., Ganjoo, M., Manning, C.D., Ng, A.: Zero-shot learning through cross-modal transfer. In: NIPS, pp. 935–943 (2013)

Sampling Methods in Genetic Programming Learners from Large Datasets: A Comparative Study

Hmida Hmida[1,2(✉)], Sana Ben Hamida[2], Amel Borgi[1], and Marta Rukoz[2]

[1] Université Tunis El Manar, LIPAH, Tunis, Tunisia
hhmida@gmail.com
[2] Université Paris Dauphine, PSL Research University, CNRS, UMR 7243, LAMSADE, 75016 Paris, France

Abstract. The amount of available data for data mining and knowledge discovery continue to grow very fast with the era of Big Data. Genetic Programming algorithms (GP), that are efficient machine learning techniques, are face up to a new challenge that is to deal with the mass of the provided data. Active Sampling, already used for Active Learning, might be a good solution to improve the Evolutionary Algorithms (EA) training from very big data sets. This paper present a review of sampling techniques already used with active GP learner and discuss their ability to improve the GP training from very big data sets. A method in each sampling strategy is implemented and applied on the KDD intrusion detection problem using very close parameters. Experimental results show that sampling methods outperforms results obtained with full dataset but some of them cannot be scaled to large datasets.

1 Introduction

Genetic Programming (GP) [9] is an Evolutionary Algorithm (EA) considered as a universal solver. This paradigm have shown its potential by reinventing previously invented patents and creating new patentable inventions.

Applied to supervised learning and classification problems, GP generates a population of classifiers by means of genetic operators. The performance of each classifier is measured by a fitness function that needs the evaluation (execution) of every program against the complete training dataset. This leads to a computational overhead increased specially with large datasets and proportional to the product of the number of instances in the dataset, the population size and the number of generations that will be carried out during the evolution process. Otherwise, using the full learning data might be impossible when the input data doesn't fit within the main memory which is often the case with Big Datasets.

The problem of reducing this calculation cost can be addressed mainly with two approaches:

© Springer International Publishing AG 2017
P. Angelov et al. (eds.), *Advances in Big Data*, Advances in Intelligent Systems and Computing 529, DOI 10.1007/978-3-319-47898-2_6

– Speeding evaluation by means of hardware acceleration and parallelization.
– Reducing the number of evaluations by selecting a training subset much smaller than the entire dataset: training set sampling.

A large variety of sampling methods were investigated in many classification problems using GP and demonstrated a success comparing to the use of entire training set [4,5,8,10,17]. However, the results obtained from the earlier experiments cannot be cross-compared since they are based on different GP variants (Standard, Linear, Cartesian...), different datasets and GP parameters. In this paper, we will try to put a representative set of sampling methods in a very close experimental conditions, to show their impact on learning speed and accuracy.

This paper is organized as follows. Section 2 reviews the main sampling methods already implemented for active GP learners. A brief description of the design issues for the comparative study is given in Sect. 3. The last section discusses the obtained results under these experimental conditions.

2 Sampling Methods: A Review

The main objective of the studied sampling methods, in this paper, is to reduce the original training dataset size by substituting it with a representative subset much smaller and hence the evaluation cost of GP algorithm. Two major classes of sampling techniques can be layed out: static sampling and dynamic sampling.

2.1 Static Sampling Methods

With static sampling, the learner obtains all the input training set at once and stills unmodified across the learning process. Training subset is created by selecting a given number of patterns, independently from the training process in which they will be used. Some methods use a unique subset during all GP runs. Other ones can assign different subsets for each run. For example, the *Historical Subset Selection* **HSS** [6] gathers misclassified instances from previous GP runs (using full dataset) at each generation using the best individual.

Other well known static sampling methods are the Bagging/Boosting techniques. These techniques were applied to GP in [8] but not as a speeding method but rather to improve GP quality by evolving several subpopulations with different training sets generated by selecting the base dataset with replacement.

2.2 Dynamic Sampling Methods

Dynamic sampling methods are also referred to as active data selection methods as derived from *Active Learning* [2]. The underlying sampling techniques are tightly related to the learning process evolution. Generally, it depends on a particular feature like unresolved cases, the number of generations reached, ...

Random Sampling. The selection of fitness cases is based on a uniform probability among the training subset. This stochastic selection helps to reduce any bias within the full dataset on evolution. In *Random Subset Selection* **RSS** [6], at each generation g, the probability of selecting any case i is equal to $P_i(g)$:

$$\forall i : 1 \leq i \leq T, \quad P_i(g) = \frac{S}{T}. \tag{1}$$

where T is the size of the full dataset and S is the target subset size. The sampled subset have a fluctuating size around S. Fixed Random Selection (**FRS**) [17] is a very similar method with a fixed number of cases selected at every generation. *Stochastic Sampling* **SS** [13] is another method using the same probabilistic selection to construct subsets for each individual per generation.

Weighted Sampling. The selection of fitness cases to be added to the subset is based on a calculated weight that measures how much a pattern is worthy and can help to sharpen the population quality.

The very first algorithm in this category is *Dynamic Subset Selection* **DSS** [5,6]. This algorithm is intended to preserve training set consistency while alleviating its size by keeping only the *d*ifficult cases with ones not selected for several generations. Each dataset record is assigned a difficulty degree $D_i(g)$ and an age $A_i(g)$ starting with 0 at first generation and updated at every generation. The difficulty is incremented at each misclassification and reset to 0 if the fitness case is selected. The age is equal to the number of generations since last selection, so it is incremented when the pattern has not been selected and reset to 0 otherwise. The resulting weight W of the i^{th} case is calculated with this sum:

$$\forall i : 1 \leq i \leq T, \quad W_i(g) = D_i(g)^d + A_i(g)^a. \tag{2}$$

The selection probability is biased by fitness case difficulty and age:

$$\forall i : 1 \leq i \leq T, \quad P_i(g) = \frac{W_i(g) * S}{\sum_{j=1}^{T} W_j(g)}. \tag{3}$$

DSS needs three parameters to be tuned: difficulty exponent (d), age exponent (a) and target size (S). Gathercole [6] tried different subset and population sizes with The thyroid classification problem[1] using a full training set size of 3772. With 10000 individuals, GP with DSS realizes better results than neural networks. DSS reduced evaluations by 20 % whilst obtained similar classifiers.

Hierarchical Sampling. This kind of sampling is based on multiple levels of sampling methods inspired by the concept of a memory hierarchy. It combines several sampling algorithms that are applied in different levels.

Curry and Heywood conceived an extension to DSS into a 3 levels hierarchy [3]. At level 0, the dataset was first partitioned into blocks that were sufficiently small to reside within RAM alone. Then, at level 1, blocks were chosen

[1] See UCI Machine Learning Repository at http://archive.ics.uci.edu/ml/.

from this partition based on RSS or DSS. Finally, at level 2, the selected block is considered as the full dataset on which DSS was applied for several rounds. Depending on the level 1 algorithm, we have **RSS-DSS** hierarchy or **DSS-DSS** hierarchy. Besides DSS parameters (see Sect. 2.2), new parameters are added: level 0 block size, level 1 number of iterations, level 2 iterations, maximum level 2 iterations and tournament iterations. Only level 2 iterations is calculated by:

$$I_b(i) = I_{max} * E_b(i-1). \tag{4}$$

$I_b(i)$: number of level 2 iterations on block b in i^{th} level 1 iteration.
I_{max}: maximum level 2 iterations.
$E_b(i-1)$: error rate over block b for the best case at previous iteration.

A modified DSS in level 2 is used where two roulette wheels exist per block, one is used to control the selection of patterns with respect to age and the other to difficulty, the roulette wheels being selected in proportion to the corresponding probability for age and difficulty (2 additional parameters).
For the DSS-DSS hierarchy, Eqs. 2 and 3 are extended to blocks in [3].
Tested against KDD-99 [15] and Adult dataset [3], both algorithms realize competitive results in very shorter time and DSS-DSS outperforms RSS-DSS.

An extension to this work is made in [4] with the *Balanced Block DSS* by altering mainly the level 0 block selection with the goal of obtaining a balanced block in level 1 with respect to a fixed ratio.

Incremental Sampling. The training subset starts with a few cases and acquires more cases at every generation to reach the full dataset size.

Zhang [16,17] proposes to perform a uniform data crossover simultaneously with genetic programs evolution. Data crossover means that when crossover operation is applied to genetic programs to produce new programs, their subsets are crossed to obtain new offspring subsets inducing data inheritance through generations. This helps to preserve the knowledge acquired by the parents. This method called *Incremental Data Inheritance* **IDI** starts with n_0 sized subsets, and increases by λ at every generation with respect to an import rate depending on a third parameter ρ. In addition to that Zhang uses an *Adaptive Fitness Function* varying from an individual to another [17]. This method was applied to the evolution of collective behaviors for multiple robotic agents [16] and time series prediction problem [17]. It was compared to the standard GP, GP with Incremental Random Selection (**IRS**) and FRS.

Experimental evidence supports that evolving programs on an incrementally selected subset of fitness cases can significantly reduce the fitness evaluation time without sacrificing generalization accuracy of the evolved programs.

Topology Based Sampling. Inspired by the idea that structure influences the efficiency of heuristics working on it, this method consists at building a topology of the problem from the knowledge acquired by individuals about fitness cases. Lazarczyk [10] suggests a *Fitness case Topology Based Sampling* **TBS** in which relationship between fitness cases in the training set is represented by an undirected weighted graph. Vertices are fitness cases and edges have a weight measuring a similarity or a distance induced from individuals' performance. Then, cases having a tight relationship with respect to a threshold cannot be selected together in the same subset assuming that they have an equivalent difficulty for the population. Edge values start at 0 and are updated after evaluation phase by increasing the weight of all edges between solved cases by each individual and decreasing the weight of all edges by a loss rate λ.

This algorithm uses a unique subset for all the individuals renewed each generation after updating the topology graph. This subset is constructed by selecting randomly a case from a candidates set (initially equal to full dataset) until reaching the target size or the candidates set is empty. All fitness cases connected to it with an edge weight exceeding the threshold are removed from candidates.
The threshold is calculated using a binary-search like algorithm, to have a value adapted to the current topology: it must not be too high or too low.

Experimental results on classification problems (intertwined spirals and the Thyroid problems) demonstrated improvements in mean fitness value through generations compared to DSS and SSS.

Other Sampling Techniques. *Rational Allocation of Trials* [14] **RAT** is an algorithm that does not sample the dataset but uses a minimal set of fitness cases for evaluation and then decides for each individual whether to carry on evaluation with the next fitness case according to the probability that they might win some tournament that they are losing or lose some tournament they are winning.

Balanced sampling [7] is a method aiming to improve classifiers accuracy by correcting the original dataset imbalance within majority and minority class instances. It has some methods based on the minority class size and thus reduce the number of instances like the methods studied in this paper. The following sampling methods are used with GP:

– **Static Balanced Sampling:** selects cases randomly from each class without replacement until obtaining a balanced subset with equal number of majority and minority class instances of the desired size at every generation.
– **Basic Under-sampling (BUSS):** selects all minority class instances and then an equal number from majority class randomly.
– **Basic Over-sampling (BOSS):** selects all majority class instances and then an equal number from minority class randomly with replacement.
– **Under-sampling A/B** and **Over-sampling A/B:** Multiple balanced samples created at each generation with BUSS (respectively BOSS). Then, individuals are attributed the average fitness realized across all the sample sets (and the minimum fitness for version B).

3 Design Issues for the Comparative Study

3.1 Cartesian GP (CGP)

CGP is a form of GP, proposed by Julian Miller [12], that represents genetic programs as directed acyclic graphs and mostly inspired by digital circuits called FPGA. In this graph, nodes representing functions have many inputs and one output, and are layed into two dimensional matrix. This form of GP [1] is highly competitive with other GP methods while it does not bloat. In addition to that it is easy to implement and can have multiple outputs.

We use here a CGP implementation done by David Oranchak [1] as a contribution package to Sean Luke's ECJ [11].

3.2 Learning Data

We used here a large dataset, which was firstly created for the competition held at the Fifth International Conference on Knowledge Discovery and Data Mining KDD-99 and is about intrusion detection problem. The original training dataset contains 5 million lines conveying data about connections from the simulated traffic and their labels as normal connections and four attack classes. Another set, called corrected set is constructed with the same way and used as test dataset. The training process is run on the derived 10 % KDD-99 dataset to learn how to classify normal connections and attacks. The best individual of each run is then tested on the test set. Both datasets are presented in Table 1.

Table 1. KDD-99 datasets composition.

Class		Normal	Dos	Probe	R2L	U2R	Total patterns
Number of patterns	*10 % Training Set*	97278	391458	4107	1126	52	494021
	Test Set	60593	229853	4166	16347	70	311029

3.3 Implemented Sampling Methods

RSS and DSS were implemented identically to their author's description. The remaining methods have been altered as described hereafter.

BRSS. This method is Balanced RSS in which the subset generated by RSS reflects the original dataset classes' frequencies. The number of each class patterns is calculated with respect to the target size.

RSS-DSS. The implemented RSS-DSS here does not use the same fitness as proposed by authors. It uses the same fitness function as the other implemented methods. The effect of this transformation will be discussed in Sect. 4.3. Two configurations was tested as shown on the Table 4 changing target size, RSS and DSS iterations simultaneously. RSS-DSS2 denotes the second configuration.

IDI. Since we use a fixed subset grow increment, the full dataset size is not reached at the last generation.

TBS. We calculated the threshold applying a more simpler steps: we start by the lowest edge weight as the initial threshold to have the smallest possible subset and then TBS selection/exclusion steps are applied using the current threshold. When the subset size is to small, the new threshold value is the next edge weight and the previously excluded patterns are added to candidates.

BUSS. KDD-99 have a normal class and 4 attack classes. The subset selection uses the U2R attack (52 patterns) class as minority class and then randomly selects an equal number of patterns from the remaining 3 attacks to obtain 208 attack patterns. In a first experiment, 52 normal were added obtaining a subset of 260 patterns. In the second experiment (BUSS2), 208 normal patterns were chosen to get a final subset of 416 patterns. Subsets are renewed each generation.

3.4 Performance Metrics

By the end of each run, the best individual based on the fitness function is tested on the whole dataset and then the test dataset. Results are recorded in a confusion matrix from which Accuracy, True Positive Rate (TPR) and False Positive Rate (FPR) are calculated. The training time is the amount of elapsed seconds between the first and the last generation.

4 Experiments and Results

4.1 GP Parameters

The EA used for the experimental study is the Cartesian GP (CGP). With CGP, programs are coded with integer linear chromosome divided into groups. Each group corresponds to a position in a 2-D array.

Terminal and function sets. The terminal set includes input variables which are the 41 KDD-99 features set with 8 randomly generated constants. The function set has 17 functions that are basic arithmetic, comparison and logical operators (Table 2).

CGP parameters. A common approach in tuning GP is to undertake a series of trials to make parameter choices for the final GP runs. Since the main objective is to compare sampling methods in closely context, this tuning procedure is not fully investigated. The final design of CGP parameters used in this work is summarized in Table 3.

4.2 Specific Parameters for Sampling Methods

The values of additional parameters brought by each sampling method are assigned with respect to original method paper. Some of theses methods have been modified in a order to fit in a comparable context. The Table 4 below shows each method configuration.

Table 2. Terminal and function sets.

Function (node) set		Terminal set	
Arithmetic operators:	+, −, ∗, %	KDD-99	
Comparison operators:	<, >, <=, >=, =	Features	41
Logic operators:	AND, OR, NOT, NOR, NAND		
Other :	NEGATE, IF (IF THEN ELSE),	Random	
	IFLEZE(IF <=0 THEN ELSE)	Constants	8 in $[-2, 2[$

Table 3. CGP parameters.

Parameter	Value
Population size	512 for RSS, DSS, BUSS, BRSS and RSS-DSS
	128 for IDS, TBS and Full Subset
Subpopuations number	1
Number of generations	500 for RSS, DSS, BUSS and BRSS
	100 for IDS, TBS
	depending on the RSS iteration for RSS-DSS
CGP nodes	300
Inputs/Outputs	49/1
Tournament size	4
Crossover/Mutation probability	0.9/0.04
Fitness	Minimize classification errors

4.3 Experimental Results

Experiments are performed on an Intel i7-4810MQ (2.8 GHZ) workstation with 8 GB RAM running under Windows 8.1 64-bit Operating System. GP programs are trained to distinguish between normal and attack patterns. To compare the performance of the implemented sampling methods, six measures are recorded cross the 21 runs performed for each technique: the best individual and the mean values of the accuracy, TPR and FPR metrics for both training and test datasets. The mean and best individual measures are illustrated by Table 5. Otherwise, to compare the methods according to the computational cost, the spent time to complete each run and to find the best of run individual for all the trials are recorded. Table 6 lists the corresponding mean values for all implemented techniques. Experimental results show that introducing a sampling method to the GP system improves its performance when training from large data sets. All the implemented sampling methods have better results than the conventional GP trained on the whole dataset (FSS), and this is according to the three performance metrics (mean measures from Table 5).

From Table 5, we can make the following observations. Except for the basic version of RSS-DSS technique, all the implemented sampling techniques have given satisfactory results. As shown by best individual results, the accuracy and the TPR of the best solutions given by all the methods are quite similar.

Table 4. Specific methods' parameters.

Method	Parameter	Value
RSS	Target size	5000
BRSS	Target size	5000
	Balancing method	Full dataset distribution
DSS	Target size	5000
	Difficulty/Age exponent	1/3.5
RSS-DSS	Target size (level 2 block size)	2500 then 100
	Level 0 block size	5000
	RSS iterations	200 then 500
	Max DSS iterations	20 then 50
	Individuals evaluated per DSS iteration	100 (20 % of population size)
	Difficulty/Age exponent	1/3.5
	Difficulty/Age roulette	70 %/30 %
TBS	Target size	1000
	Loss rate	0.7
	Iterations	See paragraph below
IDI	Initial subset size	1000
	Increment	10
	Diversity factor	0.3
Basic Under (BUSS)	-	-

Table 5. Test set performance metrics

Method	Mean measures (%)			Best individual measures (%)		
	Accuracy	Recall	FPR	Accuracy	Recall	FPR
BRSS	82.4	78.56	1.72	91.18	90.58	1.23
BUSS	89.13	96.62	41.82	92.94	98.08	28.28
BUSS2	85.65	83.25	4.4	92.19	92.77	10.22
DSS	84.08	80.66	1.77	92.77	91.44	1.71
FSS	80.84	76.61	1.77	80.9	76.76	1.98
TBS	85.27	82.49	3.23	92.46	90.94	1.28
IDI	82.99	79.71	3.41	91.42	91.75	9.97
RSS	82.53	78.67	1.52	92.35	91.1	2.49
RSS-DSS	42.82	31.69	11.21	90.8	95.75	29.67
RSS-DSS2	74.07	74.49	27.63	92.02	92.98	11.94

Table 6. Time measures (in Seconds)

Method	BRSS	BUSS	BUSS2	DSS	FSS
Mean Run Time	1590.819	75.612	102.428	1996.892	7362.272
Mean 'Best of Run' Time	580.311	20.135	43.054	1358.257	2854.074
Method	**TBS**	**IDI**	**RSS**	**RSS-DSS**	**RSS-DSS2**
Mean Run Time	11174.507	11162.065	1493.788	1000.366	364.369
Mean 'Best of Run' Time	1494.977	2002.886	530.038	43.999	0.124

However, this finding is not available for the best FPR values. Indeed, according to the Accuracy and TPR metrics of the best solutions, BUSS method performed better than all the other sampling approach, but it has the 2^{nd} worst TPR value.

BUSS2, while using a larger subset (more normal cases than BUSS), has lost accuracy and TPR but realizes a much better FPR. With RSS-DSS2 settings, this method enhances its performance but FPR remains very high.

TBS, IDI and FSS have been tested with a reduced number of generations, population size, and target subset size due to the huge amount of time needed for each generation. Nevertheless, TBS and IDI reached a comparable performance level. All 3 methods have a very increased run time. Moreover, both IDI and TBS spent more time than using the full dataset assuming that the computing time needed for sampling is very high compared with the evaluation time.

From Table 6, we can see that RSS-DSS in both settings finds the best individual of run in the very first generations. BUSS makes the fastest runs because it uses the smallest subset size but this result depends on the classification problem itself and classes of the learning data.

5 Conclusion

The main purpose of this work is to study the ability of the most known active sampling techniques for GP learners to deal with very large data sets. These techniques were implemented on the same framework and applied to a classification problem on the KDD intrusion detection data set.

Three main conclusions can be deduced from the experimental results. First, any active sampling technique is able to induce better generalizing classifiers compared to the standard GP using the full data set. However, the applicability of some methods is limited because of their very high computation time (such as IDI and TBS) or because they need specific conditions to be efficient (such as RSS-DSS). This is the second conclusion. Third, simple sampling techniques, such as BUSS or RSS have very low computational cost and they are able to reach a competitive performance according to the advanced techniques.

To deal with run time problem of IDI and TBS, mixing them with other sampling methods in a hierarchical sampling is a promising solution. Fitness function affects the quality of GP results and this needs to be experimented on the sampling methods tested here.

References

1. CGP: Cartesian gp website, http://www.cartesiangp.co.uk
2. Cohn, D., Atlas, L., Ladner, R.: Improving generalization with active learning. Mach. Learn. **15**, 201–221 (1994)
3. Curry, R., Heywood, M.: Towards efficient training on large datasets for genetic programming. In: Tawfik, A.Y., Goodwin, S.D. (eds.) AI 2004. LNCS (LNAI), vol. 3060, pp. 161–174. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24840-8_12
4. Curry, R., Lichodzijewski, P., Heywood, M.I.: Scaling genetic programming to large datasets using hierarchical dynamic subset selection. IEEE Trans. Syst. Man Cybern. Part B **37**(4), 1065–1073 (2007)
5. Gathercole, C.: An Investigation of Supervised Learning in Genetic Programming. University of Edinburgh, Thesis (1998)
6. Gathercole, C., Ross, P.: Dynamic training subset selection for supervised learning in Genetic Programming. In: Davidor, Y., Schwefel, H.-P., Männer, R. (eds.) PPSN 1994. LNCS, vol. 866, pp. 312–321. Springer, Heidelberg (1994). doi:10.1007/3-540-58484-6_275
7. Hunt, R., Johnston, M., Browne, W., Zhang, M.: Sampling methods in genetic programming for classification with unbalanced data. In: Li, J. (ed.) AI 2010. LNCS (LNAI), vol. 6464, pp. 273–282. Springer, Heidelberg (2010). doi:10.1007/978-3-642-17432-2_28
8. Iba, H.: Bagging, boosting, and bloating in genetic programming. In: The 1st Annual Conference on Genetic and Evolutionary Computation, Proceedings of GECCO 1999, vol. 2, pp. 1053–1060. Morgan Kaufmann, San Francisco (1999)
9. Koza, J.R.: Genetic programming: on the programming of computers by means of natural selection. Stat. Comput. **4**(2), 87–112 (1994)
10. Lasarczyk, C., Dittrich, P., Banzhaf, W.: Dynamic subset selection based on a fitness case topology. Evol. Comput. **12**(2), 223–242 (2004)
11. Luke, S.: Ecj homepage. http://cs.gmu.edu/~eclab/projects/ecj/
12. Miller, J.F., Thomson, P.: Cartesian genetic programming. In: Poli, R., Banzhaf, W., Langdon, W.B., Miller, J., Nordin, P., Fogarty, T.C. (eds.) EuroGP 2000. LNCS, vol. 1802, pp. 121–132. Springer, Heidelberg (2000). doi:10.1007/978-3-540-46239-2_9
13. Nordin, P., Banzhaf, W.: An on-line method to evolve behavior and to control a miniature robot in real time with genetic programming. Adaptive Behav. **5**(2), 107–140 (1997)
14. Teller, A., David, A.: Automatically choosing the number of fitness cases: the rational allocation of trials. In: Genetic Programming 1997: Proceedings of the Second Annual Conference, pp. 321–328. Morgan Kaufmann (1997)
15. UCI: Kdd cup (1999). http://kdd.ics.uci.edu/databases/kddcup99/
16. Zhang, B.-T., Cho, D.-Y.: Genetic programming with active data selection. In: McKay, B., Yao, X., Newton, C.S., Kim, J.-H., Furuhashi, T. (eds.) SEAL 1998. LNCS (LNAI), vol. 1585, pp. 146–153. Springer, Heidelberg (1999). doi:10.1007/3-540-48873-1_20
17. Zhang, B.T., Joung, J.G.: Genetic programming with incremental data inheritance. In: The Genetic and Evolutionary Computation Conference, Proceedings, vol. 2, pp. 1217–1224. Morgan Kaufmann, Orlando (1999)

A Fast Deep Convolutional Neural Network for Face Detection in Big Visual Data

Danai Triantafyllidou and Anastasios Tefas(✉)

Artificial Intelligence and Information Analysis Lab, Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece
danaimar@csd.auth.gr, tefas@aiia.csd.auth.gr

Abstract. Deep learning methods are powerful approaches but often require expensive computations and lead to models of high complexity which need to be trained with large amounts of data. In this paper, we consider the problem of face detection and we propose a light-weight deep convolutional neural network that achieves a state-of-the-art recall rate of 90 % at the challenging FDDB dataset. Our model is designed with a view to minimize both training and run time and outperforms the convolutional network used in [2] for the same task. Our model consists of only 76.554 free parameters whereas the previously proposed CNN for face detection had 60 million parameters. Our model also requires 250 times fewer floating point operations than AlexNet. We propose a new training method that gradually increases the difficulty of both negative and positive examples and has proved to drastically improve training speed and accuracy. The proposed method is able to detect faces under severe occlusion and unconstrained pose variation and meets the difficulties and the large variations of real-world face detection..

1 Introduction

Face detection has been an active research area in the computer vision field for more than two decades mainly due to the numerous applications that require face detection as a first step (e.g., face verification [1]). The seminal work of Viola and Jones [3], made real-time face detection possible and later on inspired many cascade-based methods. Thereafter, research in face detection made noteworthy progress as a result of the availability of data in unconstrained capture conditions, the development of publicly available benchmarks and the fast growth in computational and processing power of modern computers.

The original Viola-Jones detector is fast to evaluate, yet fails in detecting faces from different angles. This issue was initially addressed either by using one classifier cascade for each specific facial view [7,8], or by using a decision tree for pose estimation and the corresponding cascade to verify the detection [6]. However, these approaches require pose/orientation annotations while complex cascade structures increase the computational cost. The main line of research in this direction was based on the combination of robust descriptors with classifiers [9,10]. Among the variants, a method named Headhunter [11] improved the

P. Angelov et al. (eds.), *Advances in Big Data*, Advances in Intelligent Systems and Computing 529, DOI 10.1007/978-3-319-47898-2_7

performance by deploying the integral channel features method along with 22 cascades. Finally, a joint cascade-based method proposed in [4] achieved state-of-the-art results by introducing an alignment step in the cascade structure.

Another common family of face detection algorithms learn and deploy a Deformable Parts-based Model (DPM) [12] to model the information between facial parts. While DPM detecors are more robust to occlusion than cascade based methods, they lack in computational efficiency and are prohibitive for real-time detection. In [12] a unified DPM framework for face detection, pose estimation, and landmark estimation was proposed. In [13], a simple DPM provides excellent performance and outperforms more complex DPM variants. Finaly, the detection accuracy was significantly improved by a face detector called Deep Pyramid DPM [14]. The last method, generates a deep feature pyramid and uses a linear SVM for classification.

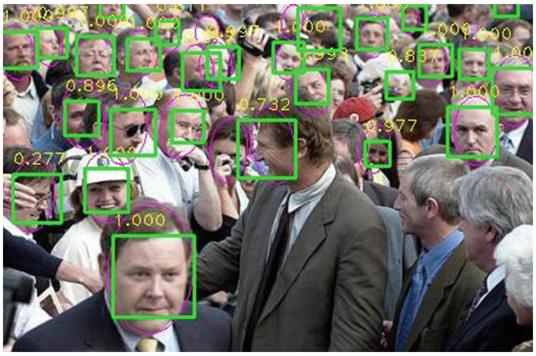


Fig. 1. An example of face detection in various poses and occlusions. The bounding boxes and scores show output of the trained CNN.

The recent resurgence of interest in deep neural networks owes a certain debt to the availability of powerful GPUs which routinely speed up common operations such as large matrix computations. Deep convolutional neural networks have wide applications in language processing, object classification and recommendation systems. A deep network named Alexnet [15], which was trained on ILSRC 2012, rekindled interest in convolutional neural networks and outperformed all other methods used for large scale image classification. The R-CNN method proposed in [16] generates category-independent region proposals and uses a CNN to extract a feature vector from each region. Then it applies a set of class-specific linear SVMs to recognize the object category. Recently, a face detector called DDFD [2], showed that a CNN can detect faces in a wide range of orientations using a single model.

In this study, a novel CNN for face detection is presented that extends the work in [2]. It is shown that a properly trained CNN can outperform more complex and computationally expensive architectures. This paper makes the following contributions:

1. We propose a novel light-weight model, consisting of only 76.554 free parameters, and we show that our method despite its minimum complexity can provide formidable results and is suitable for real-time detection with standard processing power as opposed to most neural network based detection techniques.
2. We present a new training methodology according to which the CNN is gradually supplied with training examples of scaling difficulty. We show that our method can drastically improve training speed and significantly reduce the number of false positives.
3. We propose adding a pooling layer to the output of the deep CNN to smoothen the produced heat map.

The proposed trained model is publicly available to the research Community[1]. Figure 1 shows examples of face detection.

2 Proposed Method

2.1 CNN Architecture

We trained a fully-convolutional CNN comprised of seven convolutional layers interspersed by dropout layers. The network was trained with RGB images of size 32×32. The architecture of the CNN is summarized in Fig. 2 and Table 1. The last column of Table 1 shows the required amount of floating-point operations (FLOPS) for computing the convolutions in each layer. The network requires around 6 million FLOPS per face frame. The output of the network is comprised of two different detection scores, each one corresponding to the two classes of the detection task (e.g. face, non-face). Our work follows the pipeline presented in [2], in a sense that our method does not require any extra module (e.g. SVM) for classification as the CNN's output is describing enough for the task of face detection. As the model is fully-convolutional it accepts images of arbitrary size and produces a heat map of the face detector. In addition, no pooling layer was used since there was no need or room to further decrease the size of data volume flowing through the network. As stated in [20], the use of the Parametric Rectified Linear Unit (PReLU) function had a positive impact regarding the detection accuracy of the CNN.

2.2 The Dataset

The CNN was trained with positive examples extracted from the AFLW [17] and the MTFL [21] datasets. The first consists of 21 K images with 24 K face annotations while the second consists of 12 K face annotations. Both datasets include real world images with expression, pose, gender, age and ethnicity variations. For AFLW we used the provided face rectangle annotations. For MTFL

[1] https://github.com/danaitri/Face-detection-cnn.

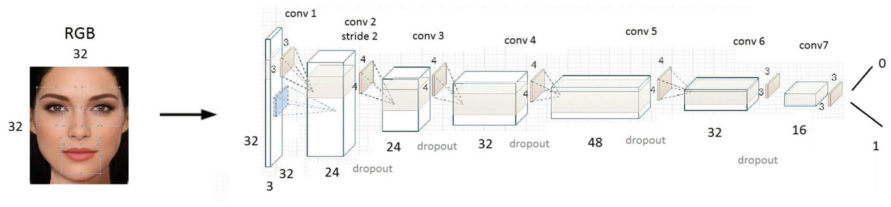


Fig. 2. The architecture of the trained CNN

Table 1. The trained CNN

Layer	Kernel	Filters	Input	Output	Weights	PReLu	Bias	FLOPS
Conv 1	3×3	24	$32 \times 32 \times 3$	$30 \times 30 \times 24$	648	24	24	510 K
Conv 2	4×4	24	$30 \times 30 \times 24$	$14 \times 14 \times 24$	9216	24	24	2 M
Conv 3	4×4	32	$14 \times 14 \times 24$	$11 \times 11 \times 32$	12288	32	32	1.5 M
Conv 4	4×4	48	$11 \times 11 \times 32$	$8 \times 8 \times 48$	24576	48	48	1.5 M
Conv 5	4×4	32	$8 \times 8 \times 48$	$5 \times 5 \times 32$	24576	32	32	614 K
Conv 6	3×3	16	$5 \times 5 \times 32$	$3 \times 3 \times 16$	4608	16	16	41 K
Conv 7	3×3	2	$3 \times 3 \times 16$	$1 \times 1 \times 2$	288		2	288
Total					76200	178	176	6 M
Free parameters					**76554**			

we used the given facial landmark annotations to produce face rectangles in a similar manner with AFLW examples regarding the positioning of faces.

The final training images of faces were resized to 32×32. This is a relatively small image size compared to image sizes typically used by AlexNet and other deep networks (e.g. in [2] AlexNet is trained with images 225×225). However, it has been proved that images of this size contain enough information to train the CNN. The relatively small image size allowed to reduce the image down to 1×1 (a face/no face result) without the use of pooling layers. We used only convolution and PReLU layers, each convolution producing a feature map smaller than its input layer.

In order to increase the number of positive and negative examples we used horizontal mirroring (flip) of the images with a probability 0.5. This has been proved very effective as it increased the number of training examples and it also led to a very large number of combinations of images to be entered in each training batch, thus allowing better generalization of the CNN. We also tried augmenting the face dataset by horizontal and vertical displacement of face images by 1 to 3 pixels along the horizontal and vertical axes. This technique proved to be inefficient in experiments and it did not give better results while training and testing.

2.3 Proposed Training Methodology

The CNN was trained successively in a set of five different data sets. Let $\mathcal{N}_T$ be a collection of images that will serve as a pool of negative examples. Let $\mathcal{D}_0$ be the original training set consisting of the original set of positive examples $\mathcal{P}_0$ and the set of negative examples $\mathcal{N}_0 \in \mathcal{P}$:

$$\mathcal{D}_0 = \mathcal{T}_0 \cup \mathcal{N}_0 \tag{1}$$

Once the training process is complete, we run the network to the set of images $\mathcal{N}_T$ and we recollect a new set of false positives $\mathcal{F}_1$ which is added to the original set of negative examples $\mathcal{N}_0$:

$$\mathcal{N}_1 = \mathcal{N}_0 \cup \mathcal{F}_1 \tag{2}$$

The set $\mathcal{F}_1$ is selected according to the network's score. During each training round, we sort the false positives according to their score and we select a predefined number of examples. In order to maintain the same ratio of positive to negative examples after each training round we increase the number of positive examples proportionally. A new set of images containing faces $\mathcal{T}_1$ is added to the original set of positive examples $\mathcal{P}_0$

$$\mathcal{T}_1 = \mathcal{T}_0 \cup \mathcal{P}_1 \tag{3}$$

The aforementioned process of training and increase of training examples is repeated after completion of training in the set D_i:

$$\mathcal{N}_{i+1} = \mathcal{N}_i \cup \mathcal{F}_{i+1} \tag{4}$$

$$\mathcal{D}_{i+1} = \mathcal{P} \cup \mathcal{N}_{i+1} \tag{5}$$

$$\mathcal{P}_{i+1} = \mathcal{P}_i \cup \mathcal{T}_{i+1} \tag{6}$$

Hence, the sets $\mathcal{N}_{i+1}$, $\mathcal{P}_{i+1}$ contain a larger number of negative examples than the sets $\mathcal{N}_i$, $\mathcal{P}_i$.

The increasing difficulty of the described process as well as the adaptation of the training set in the neural network's errors improved the network's performance in unknown data. The process of gradual training in i stages, as described previously, resolves a significantly important issue which was validated in practice. In the event of a training set being unequally distributed between the two classes a training batch may contain little to no actual examples of a class. As a result, the network may be deprived of the presence of examples of said class and the ability to identify between the two may be negatively impacted.

In all our experiments we trained the CNNs using Stohastic Gradient Descent (SGD). We start with a learning rate of 0.001 for the first 200.000 iterations and then we lower to 0.0001. The weights of the network were initialized according to the Xavier method [22]. Figure 3 shows the results of the described procedure for the FDDB dataset.

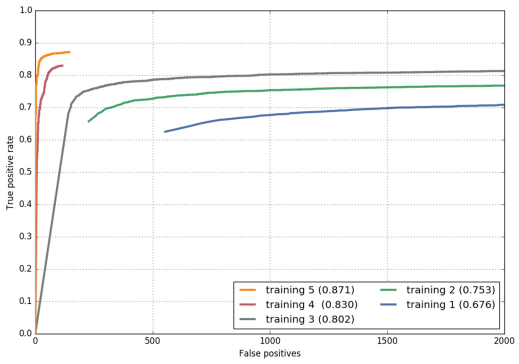


Fig. 3. The results of the proposed training methodology on the FDDB dataset for the face detection CNN.

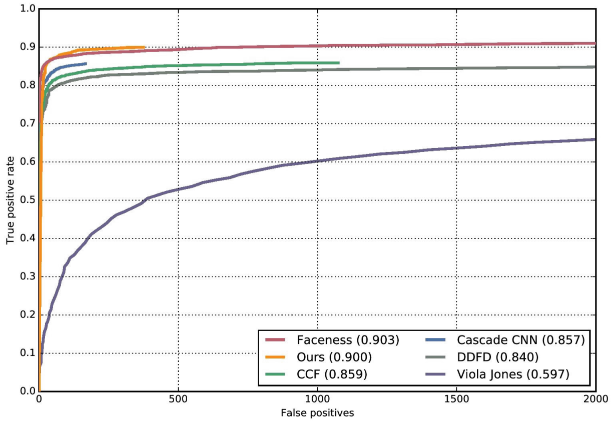
3 Experiments

3.1 System Analysis

We implemented the proposed face detector using the Caffe library [18]. The output of the network shows the scores of the CNN for every 32 × 32 window with a stride of 2 pixels in the original image. In order to detect faces smaller or larger than 32 × 32 we scale up or down the original image respectively. We apply the non maximum suppression strategy according to which all bounding-boxes with a possibility lower than the score of the maximum window multiplied by a constant factor are removed. The system was able to detect faces in the FDDB dataset that were not annotated. These detections were removed as they would count for false positives and lead to a deteriorated performance. During deployment of the CNN, we add an extra average pooling layer of 3×3 to the final output of the network. It has been verified that this layer reduces the number of false positives. The heatmap produced by the CNN is smoothened by this extra pooling layer.

3.2 Comparison with the State of the Art

We evaluate the proposed detector on the challenging dataset Face Detection Data Set and Benchmark (FFDB) [19]. Some of the recently published methods compared in this section include: DP2MFD [14], DDFD, Faceness [23], Head-hunter, JointCascade [4], SURF [9], ACF [5] and CCF [24]. For evaluation we use the toolbox provided by [11] which includes corrected annotations for the aforementioned benchmark. FDDB dataset is one of the most commonly used benchmark for face detection and consists of 2845 images with 5171 face annotations collected from journalistic articles. It is a really challenging dataset mainly due to the fact that it is rich in occluded and out-of-focus cases. Sample images with face detections using the proposed approach are shown in Figure 5. FDDB faces are annotated with elliptic regions. As stated in [11] changing the output format of detections to ellipses increases the overlap region between the detections and ground truth boxes. However, our detector achieves a high recall rate without this conversion. Figures 4(a)-4(b) show the final results on the FDDB dataset after circular training of the network in all the aforementioned datasets described in Sect. 2.3 Our method achieves a recall rate of 90 %, outperforming almost all recent published face detection methods.



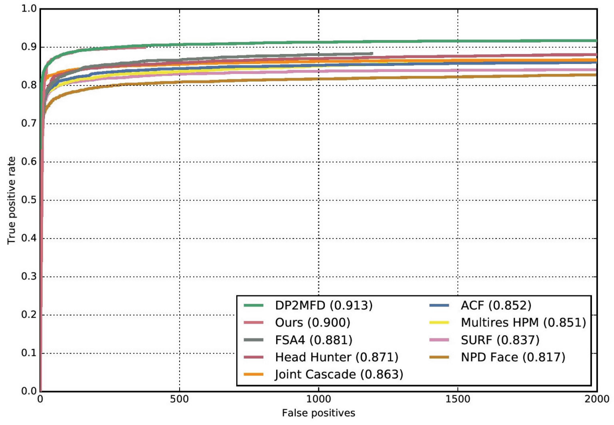


Fig. 4. Comparison of different face detectors on FDDB dataset. Against deep architectures (a) Against other state-of-the-art approaches (b)

The complexity of the competitive algorithms is very large compared to the proposed network. Indeed, the proposed deep CNN has 76.554 free parameters whereas the previously proposed deep CNN [2] had 60 million parameters. The number of the free parameters of the proposed method is considered as the total number of the network's weights including biases and PreLu parameters as shown in Table 1. The number of weights of each layer is calculated by multiplying the

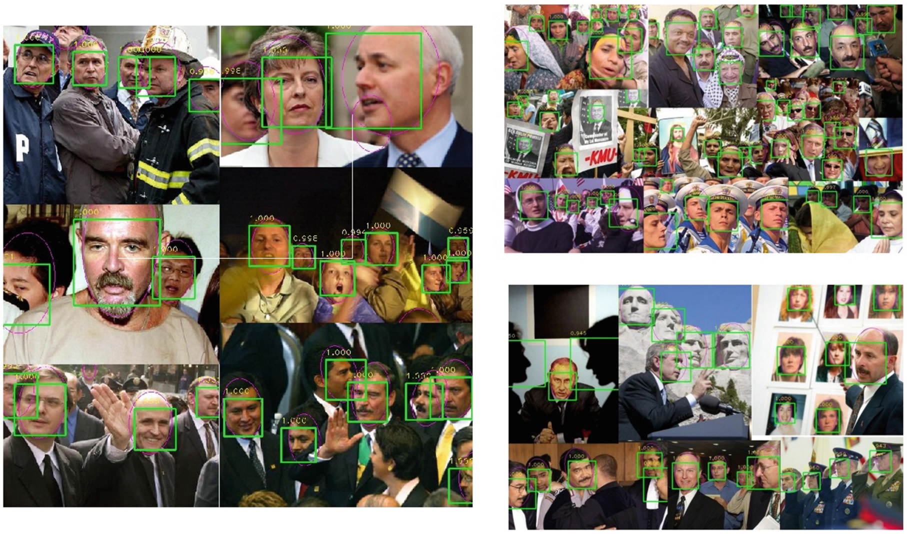


Fig. 5. Face detection examples in the FDDB databaset using the proposed CNN.

kernel size with the output and the input dimension of that layer. For example, as shown in 1 the first layer has $3 \times 3 \times 3 \times 24 = 648$ weights. The complexity of deep CNN can be estimated by the number of the required FLOPS for a single face frame. Table 2 summarizes this comparison. We compare the complexity of our network with AlexNet used in [2, 23, 25]. AlexNet requires a number of 1.5 billion FLOPS whereas our model requires only 6 million FLOPS. As a result, our model has 800 $\times$ fewer parameters and is 250 times faster than AlexNet.

This issue is very important during training but also during testing and deployment. The proposed lightweight model can be easily deployed to smart devices (e.g. smartphones, notepads, etc.) or robotic systems (e.g. drones) that do not have expensive and energy consuming multiple GPUs installed. Additionally, the proposed approach proves that when we have to deal with a specific task (i.e., face detection), even if it is very complex, we can design and train smaller and efficient architectures that outperform deeper and larger networks in performance and in execution time.

Table 2. Complexity comparison

	# parameters	# FLOPS
AlexNet	60 M	1.5 B
Ours	76 K	6 M

4 Conclusion

In this paper, we presented a novel deep convolutional neural network for the task of face detection. Our experiments on the publicly available benchmark FDDB show the success of our method. Our detector is able to detect faces in a wide range of orientations and expressions. Our detector does not require any extra modules usually used in deep learning methods such as SVM or bounding-box regression. Our work, extends the DDFD detector by using a light-weight model that improves run time and training speed. Our model outperforms the DDFD detector in the challenging FDDB dataset by a magnitude of 6 %. We show that a properly trained smaller model is efficient and outperforms a more complex and large network used for the same task. One of the future research directions is to exploit the proposed approach in order to train a lightweight deep model for face identification.

References

1. Kotropoulos, C., Tefas, A., Pitas, I.: Frontal face authentication using variants of dynamic link matching based on mathematical morphology. In: Proceedings of IEEE International Conference on Image Processing (ICIP 1998), Chicago, USA, vol. 1, pp. 122–126, 4–7 October 1998
2. Farfade, S.S., Saberian, M., Li, L.-J.: Multi-view face detection using deep convolutional neural networks. In: ICMR (2015)
3. Viola, P., Jones, M.J.: Robust real-time face detection. Int. J. Comput. Vis. **57**, 137–154 (2004)
4. Chen, D., Ren, S., Wei, Y., Cao, X., Sun, J.: Joint cascade face detection and alignment. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8694, pp. 109–122. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10599-4_8
5. Yang, B., Yan, J., Lei, Z., Li, S.: Aggregate channel features for multi-view face detection. In: IEEE International Joint Conference on Biometrics (2014)
6. Viola, M., Viola, P.: Fast multi-view face detection. In: Proceedings of CVPR (2003)
7. Wu, B., Ai, H., Huang, C., Lao, S.: Fast rotation invariant multi-view face detection based on real adaboost. In: Proceedings of IEEE Automatic Face and Gesture Recognition (2004)
8. Li, S.Z., Zhu, L., Zhang, Z.Q., Blake, A., Zhang, H.J., Shum, H.: Statistical learning of multi-view face detection. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2353, pp. 67–81. Springer, Heidelberg (2002). doi:10.1007/3-540-47979-1_5
9. Li, J., Zhang, Y.: Learning surf cascade for fast and accurate object detection. In: CVPR (2013)
10. Jun, B., Choi, I., Kim, D.: Local transform features and hybridization for accurate face and human detection. IEEE Trans. Pattern Anal. Mach. Intell. **35**, 1423–1436 (2013)
11. Mathias, M., Benenson, R., Pedersoli, M., Gool, L.: Face detection without bells and whistles. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8692, pp. 720–735. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10593-2_47

12. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: Proceedings of CVPR (2008)
13. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.: Cascade object detection with deformable part models. In: Computer Vision and Pattern Recognition (2010)
14. Ranjan, R., Patel, V.M., Chellappa, R.: A deep pyramid deformable part model for face detection. In: International Conference on Biometrics Theory, Applications and Systems (2015)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Proceedings of NIPS (2012)
16. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of CVPR (2014)
17. Martin Koestinger, P.M.R., Wohlhart, P., Bischof, H.: Annotated facial landmarks in the wild: a large-scale, real-world database for facial landmark localization. In: Proceedings of IEEE International Workshop on Benchmarking Facial Image Analysis Technologies (2011)
18. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)
19. Jain, V., Learned-Miller, E.: Fddb: A benchmark for face detection in unconstrained settings. Technical Report UM-CS-2010-009, University of Massachusetts, Amherst (2010)
20. He, S.R., Sun, K., Jian, X.Z.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: IEEE International Conference on Computer Vision (ICCV) (2015)
21. Zhang, Z., Luo, P., Loy, C.C., Tang, X.: Facial landmark detection by deep multitask learning. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8694, pp. 94–108. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10599-4_7
22. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: International Conference on Artificial Intelligence and Statistics (2010)
23. Yang, S., Luo, P., Loy, C.C., Tang, X.: From facial parts responses to face detection: a deep learning approach. In: IEEE International Conference on Computer Vision (2015)
24. Yang, B., Yan, J., Lei, Z., Li, S.Z.: Convolutional channel features. In: IEEE International Conference on Computer Vision (2015)
25. Ranjan, R., Patel, V.M., Chellappa, R.: HyperFace: A Deep Multi-task Learning Framework for Face Detection, Landmark Localization, Pose Estimation, and Gender Recognition arXiv:1603.01249 (2016)

Novel Automatic Filter-Class Feature Selection for Machine Learning Regression

Morten Gill Wollsen[1](✉), John Hallam[2], and Bo Nørregaard Jørgensen[1]

[1] Center for Energy Informatics, The Maersk Mc-Kinney Moller Institute, University of Southern Denmark, Odense, Denmark
{mgw,bnj}@mmmi.sdu.dk

[2] Center for BioRobotics, The Maersk Mc-Kinney Moller Institute, University of Southern Denmark, Odense, Denmark
john@mmmi.sdu.dk

Abstract. With the increased focus on application of Big Data in all sectors of society, the performance of machine learning becomes essential. Efficient machine learning depends on efficient feature selection algorithms. Filter feature selection algorithms are model-free and therefore very fast, but require a threshold to function. We have created a novel meta-filter automatic feature selection, Ranked Distinct Elitism Selection Filter (RDESF) which is fully automatic and is composed of five common filters and a distinct selection process.

To test the performance and speed of RDESF it will be benchmarked against 4 other common automatic feature selection algorithms: Backward selection, forward selection, NLPCA and PCA as well as using no algorithms at all. The benchmarking will be performed through two experiments with two different data sets that are both time-series regression-based problems. The prediction will be performed by a Multilayer Perceptron (MLP).

Our results show that RDESF is a strong competitor and allows for a fully automatic feature selection system using filters. RDESF was only outperformed by forward selection, which was expected as it is a wrapper which includes the prediction model in the feature selection process. PCA is often used in machine learning litterature and can be considered the default feature selection method. RDESF outperformed PCA in both experiments in both prediction error and computational speed. RDESF is a new step into filter-based automatic feature selection algorithms that can be used for many different applications.

1 Introduction

More data is available now than ever before, with cheaper sensors and the installation of sensors everywhere. The "Internet of Things" and "Big Data" are terms connected to the fact that the amount of data is increasing rapidly. Machine learning regression attempts to learn the relation between parameters of a system based on historical data. Implementing a successful machine learning algorithm requires choosing a representation of the solution, selecting relevant input features and setting parameters associated with the learning method [17].

Selecting the relevant input features, or *feature selection*, is the process of determining which subset of the combined available input data should be included to give the best performance. Feature selection is a critical task because excluding important input features means the learning algorithm will not be able to model the system. On the other hand, including unnecessary features complicates the learning. Any input that is added increases the search space by at least one dimension [17]. Feature selection can be performed by humans with the necessary domain expertise. In some cases the experts do not exist or the work itself can be expensive and time consuming [17]. A reduced feature set for the learning algorithm also reduces training time and over-generalization [5]. By automating the feature selection process, the time and expertise required is reduced and the practicality of a combined system with a learning algorithm is increased [17].

Feature selection is broadly split into three categories: *wrapper*, *embedded* and *filter* algorithms [11]. Common to all algorithms is that they only select a subset of the input features. Another strategy is to reduce the dimensions of the original feature set; such methods are named *dimension reduction* algorithms. Filtering algorithms are model-free which makes them very fast. Unfortunately they require a threshold that decides which features are selected. This presents a problem because the same threshold cannot be used for all algorithms and selecting a threshold requires a domain expert.

This paper proposes a novel filtering algorithm that automatically select features. The filter is called the Ranked Distinct Elitism Selection Filter (RDESF) and is composed of multiple common filtering algorithms. RDESF is benchmarked against other common feature selection algorithms such as forward search, backward search, PCA and NLPCA. The benchmark is performed on two time-series regression-based problems in both short-term (1 h ahead) and long-term (24 h ahead). The first problem is the prediction of indoor temperature from the SML2010 data set [18], available at the UCI Machine Learning Repository. The first of the two files in data set is used. More information is available at the UCI website [8]. The second problem is prediction of outdoor temperature. This data set is a design reference year from 2001-2010 created by The Danish Meteorological Institute [16]. Besides the weather parameters, we have added an input whether or not the sun is up, based on [9]. We have also added the earth's azimuth with reference to the sun, to have an input that differentiates the seasons. Both data sets are publicly available.

PCA will act as a baseline for the benchmark, but other commonly used filters are included to give a good indication of RDESF's capabilities. MATLAB and PCA is often used in machine learning litterature because MATLAB has implementations of Artificial Neural Networks and also includes PCA. In addition MATLAB is very easy and intuitive to use. We want to show that there are better and faster alternatives to PCA.

2 Method

The performance of RDESF will be benchmarked against commonly used feature selection algorithms described in the litterature. The benchmarking will be based on prediction error and computational speed.

2.1 Feature Selection Algorithms

The following feature selection algorithms are used:

Principal Component Analysis (PCA). PCA is a feature dimension reduction technique. The features are mapped into a smaller dimensional space to hopefully reveal structures in the underlying data [15]. The principal components are calculated using the singular value decomposition (SVD) method. Selecting a subset of the components is done by removing those components with a standard deviation close to zero with respect to machine precision.

Non-Linear Principal Component Analysis (NLPCA). Where PCA is a linear mapping between the original and the reduced dimension space, NLPCA offers a non-linear mapping, and thus any non-linear correlations between the features will be kept [7]. The non-linear mapping is performed with an artificial neural network (ANN) with three hidden layers. The middle hidden layer is a bottleneck layer and the other hidden layers are mapping layers. The bottleneck layer contains the number of nodes that the input set is reduced to. The number of nodes in the bottleneck layer of the ANN is determined by the Guttmann-Kaiser criterion, which picks components with eigenvalues above 1.0 [4]. The number of nodes in the mapping layers is set to the number of input features plus one, to avoid any bottlenecking in those layers. By training the network to approximate the input through this bottleneck layer, the bottleneck layer contains information for subsequent layers to reconstruct the input [7]. The network is trained using the backpropagation algorithm [12] until an error of 0.001 % has been achieved with a maximum of 500 iterations. After the training is complete, a new ANN is created from the first three layers. The first mapping layer is now the hidden layer and the previous bottleneck layer is now the output layer. The entire data set is then run through the new ANN, and the output of the new ANN is the reduced data set.

Forward and Backward Selection. The forward and backward selection are both wrapper-class feature selection algorithms. This means that the learning algorithm for which features are selected is included in the feature selection process. In forward selection the features are added one at a time. If the testing error decreases when a feature is added, the feature is kept. In backward selection the process is reversed where features are removed, and if the error increases, the removed features are included again [11]. As with the NLPCA, the data is randomly divided 50/50 into a training set and a test set. The error is the corrected Akaike information criterion (AICc) [1]. The formula for AICc is:

$$AICc = n \cdot \ln(RMSE) + 2(k+1) + \frac{2k(k+1)}{n-k-1} \tag{1}$$

with n being the sample size, k being the number of features and $RMSE$ being the root mean square error.

Ranked Distinct Elitism Selection Filter (RDESF). RDESF is our novel filter feature selection algorithm. It is a meta filter that combines commonly used filtering algorithms to combine their strengths and to create a broad-ranging generic filter that can be used in many application. The included filters return a ranked score of the input features based on their individual measurement. This means that a threshold is required to select the relevant features. To overcome this issue, an elitism selection is used inspired by Genetic Algorithms. The top 10 % highest ranked features are selected from every included filters. From the combined features a distinct selection based on set theory is performed to remove duplicate features. In our case the selection is a union. The process of RDESF is:

1. Let every included filter score the features based on their respective measurement
2. Rank the scores and select the best 10 % from each filter
3. Perform a union selection on the combined selected features from the filters

The included filters are Shannon entropy, Granger Causality, Mutual Information (MI), Pearson and Spearman:

Shannon entropy. The Shannon entropy is a measure of unpredictability, which means that features with unique probabilities will be ranked higher. The entropy of a feature, $H(X)$ is defined as:

$$H(X) = -\sum_i P(x_i) \log P(x_i) \tag{2}$$

with $P(x_i)$ being the probability density function, X is the feature and x_i are the samples of that feature [14].

Granger Casuality. A variable X "Granger-causes" Y if Y can be better predicted using the histories of both X and Y than it can using the history of Y alone [3]. The Granger causality score is calculated by creating two linear regressions; one that only contains the samples from Y and one that also includes the samples from X. F-tests are used to reject the null hypothesis that X does not Granger-cause Y. By using an F-distribution every input X_i can be scored as to how much it causes Y.

Mutual Information (M). The MI is a measure of the distance between the distributions of two variables and hence the dependence of the variables [2]. Choosing features with high dependence to the output could indicate that the feature is good for predicting the output. The MI of an input X to the output Y is defined as [2]:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \tag{3}$$

with $p(x,y)$ being the joint probability density function and $p(x)$ and $p(y)$ being marginal probability distribution functions of X and Y respectively.

Pearson. The Pearson correlation coefficient is a measure of the dependence between two variables [13]. The pearson correlation coefficient is defined as [13]:

$$\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} \tag{4}$$

with cov as the covariance and σ_X and σ_Y are the standard deviations of X and Y respectively.

Spearman. Spearman's rank correlation coefficient is simply the Pearson correlation coefficient applied to ranked data [13]. Ranking the data will be more resistant to outliers [13], and does not assume the data is numerical. The ranking used is the normal ordering of the input.

2.2 Artificial Neural Network

Artificial Neural Networks (ANNs) are universal approximators [6] and have been applied successfully in regression based problems for many years. Multilayer Perceptrons (MLPs) are one of the most used ANNs also known as feedforward networks. An MLP will be used to attempt to solve the regression problems. The ANN uses the tanh activiation function, 20 nodes in the hidden layer and the RPROP training algorithm [10]. A rule of thumb states to use the average of the number of input features and the number of outputs as the number of nodes in the hidden layer. However, we found that fewer nodes results in a better generalization, and we've settled on 20 hidden nodes. The network is trained until an error of 0.001 % is achieved with a maximum of 500 iterations. For further information on the technique the reader is refered to [19]. The focus on this paper is on the feature selection, which is why we have chosen MLP which may be the most basic, but very efficient, type of ANNs. Researchers choice of ANN comes down to personal preference, the type of problem, but also what is *hip* at the time. Modern types of ANNs such as Deep Neural Networks have feature selection functionality as well, but feature selection as a pre-processing step will always decrease the learning complexity, regardless of the ANN type.

10-fold cross-validation is performed, and in each fold the network is trained and tested 10 times to decrease the influence by the randomness in the ANN. The average of those 10 repetitions is used for the comparison.

2.3 Statistics

The error of the prediction is calculated using the root mean square error (RMSE) measurement. The RMSE is defined as [13, p. 497]:

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n}(\hat{y}_t - y_t)^2}{n}} \tag{5}$$

where $\hat{y}_t$ is the predicted value and y_t is the actual value. The number of predictions in the series is denoted n. RMSE punishes negative and positive errors equally.

The feature selection methods will be compared against each other using a Wilcoxon signed-rank test. Because the output from all methods are used in the equally configured ANN, we assume the samples are dependent, and hence we must use a paired test. Because we have a small sample size, we cannot make any safe assumptions about the underlying distribution. For that reason a rank test is necessary. The Wilcoxon signed-rank test works any measurement type and returns a p-value on the null hypothesis that the two sample populations are identical. This also means that any specific error measurements or time measurements will not be presented. The Wilcoxon signed-rank test uses the following test statistic, W:

$$W = \sum_{i=1}^{N_r} [\text{sgn}(x_{2,i} - x_{1,i}) \cdot R_i] \tag{6}$$

where N_r is the number of pairs with equalities removed, R_i is the rank of pair i and x_1 and x_2 are the pair samples.

2.4 Experiments

The feature selection algorithms will be tested against two problems. Both data sets are publicly available and will function as benchmarks for future comparison and experimentation. The problems are time-series problems and we assume the parameters change naturally over time. For this reason it is necessary to add the delayed input and output to the available input features. We did a grid search on a reduced version of one of the data sets, and found that a delay of 12 h gives the best performance. We assume this delay will perform equally well for the full data set as well as the other data set. With the delay the first problem will have a total of 285 input features and the second problem will have a total of 142 features. To perform the long term prediction the output is shifted accordingly.

Besides the prediction error the computational speed will also be measured. This gives an indication of the implications of using the feature selection algorithms. The time will also be measured 10 times in each cross fold to even out any randomness. The timer is started before the feature selection and stopped after the output from the ANN has been denormalized. There is no significant time difference between short-term and long-term predictions, so only the measured time from the short-term prediction will be used for comparison.

The data is first delayed and then divided into the crossfold bins. In each crossfold bin the feature selection is performed on the training part followed by a normalization of the entire data in the bin to prepare it for the ANN. After the ANN training the same features are selected from the test part and the ANN is run using this data.

3 Results and Discussion

3.1 Experiment 1 - SML2010 Data Set

The results of the performance of predicting the indoor temperature and the computational speed can be seen in Table 1. With 5 % significance the prediction with RDESF outperforms backward selection, NLPCA and PCA on both short-term and long-term. RDESF also outperforms using all available features on short-term and with 10 % significance on long-term. Only forward selection is able to get a better prediction than RDESF and only on short-term. On long-term there is not enough statistical significance to make any statements. It was expected that forward selection performs best because it includes the model in the feature selection process. Interestingly, the backward selection also includes the model in the selection process but does not have the same performance.

Table 1. p-values from the Wilcoxon signed-rank test for methods compared to RDESF for the SML2010 data set

	Backward	Forward	NLPCA	PCA	All input features
Short-term error	0.019	1.0	0.001	0.003	0.014
Long-term error	0.018	0.862	0.001	0.002	0.053
Computational speed	0.001	0.001	0.001	0.001	1.0

Looking at the computational speed in Table 1, RDESF outperforms all other algorithms with 1 % significance. Only using no algorithms at all is faster and therefore best in terms of computational speed. Using all input features equals no pre-calculations before the prediction and the fact that the MLP can be computed in parallel makes this the fastest option. This result will not reproduce for other types of neural networks because they are not suited for a large number of input parameters, for example Support Vector Regression (SVR). A large increase in the number of features will also affect the speed for MLP, however this has not been encountered yet.

3.2 Experiment 2 - Temperature Forecasting

Results from the temperature prediction from the design reference year data set can be seen in Table 2. The results do not change much between short term

Table 2. *p*-values from the Wilcoxon signed-rank test for methods compared to RDESF for the reference year data set

	Backward	Forward	NLPCA	PCA	All input features
Short-term error	0.652	1.0	0.001	0.003	0.52
Long-term error	0.313	1.0	0.001	0.002	0.423
Computational speed	0.001	0.001	0.001	0.001	1.0

and long term prediction as seen in experiment 1. Our algorithm RDESF clearly outperforms both NLPCA and PCA (with 1 % significance) when it comes to the prediction error. There is not statistical significance for the performance of RDESF against using all available input features nor backward selection. Again RDESF is only outperformed by the forward selection.

Just like in experiment 1, we expected that forward selection would outperform RDESF, because the model is included in the feature selection process. In experiment 1 we see that backward selection does not have the same superior performance as forward selection. Forward and backward selection both have advantages and disadvantages and it might be that the disadvantages of backward selection is influencing the results. One could overcome the disadvantages of both selection algorithms by using stepwise regression that combines forward and backward selection, but that will not be further investigated in this paper.

The computational speed of RDESF outperforms all other algorithms with 1 % significance just as in experiment 1. As expected using all input features is also faster in this experiment. However, keep in mind that using all input features will increase the computational time heavily for other types of neural networks.

3.3 Discussion of RDESF

The filters we chose to include in RDESF are by no means final. Mutual information (MI), Pearson and Spearman ranking are all measures of dependency. Their effectiveness is based on the assumption that a dependence between an input feature and the output will equal a better prediction performance. The Shannon entropy ranks features higher that are unique with respect to their probability density function. Selecting features with a high Shannon entropy score will include features that are different from each other and thereby decreasing the amount of similar information given to the prediction algorithm. The last included filter was the Granger causality which investigates if histories of the input feature and the output are better together. We believe that this mix of different types of filter makes RDESF strong and a generic solution to automatic feature selection of the filter-class. We will continue to investigate the performance of RDESF with other filters.

The numbers of filters included in RDESF has a big influence. Too many filters will decrease the influence of the individual filter. Because of the filter-wise selection, all possible features can be selected if too many filters are included. On the other hand, too few filters will mean the individual filters are too influential. If the filters have measurements, it means that the 10 % from every filter will be almost identical. The meta approach implies that a variety of included filters with different types of measurements will result in a better performance.

The union selection used as the distinct selection in RDESF was the obvious choice for us. Other set theory selections should be investigated such as intersection or even cartesian product. Other elaborate possibilities such as heuristics or voting systems should also be further investigated in future work.

Our initial goal was to beat the prediction performance by using PCA. PCA is included in MATLAB which is often used in machine learning. PCA is often the default feature selection method used and it has clear advantages such as reducing the dimension space and reversibility. It is a very positive result that RDESF outperforms PCA. That RDESF outperforms or evens with using all input features is a good indication that RDESF will perform well for computationally heavy types of ANNs. Support Vector Regression is one of those types of network, and preliminary results show that RDESF performs equally well for SVRs and RBFs. Testing the RDESF with other types of ANNs and other application areas are planned for further research.

Through careful implementation of the included filters, RDESF is very fast, especially compared to the computational intensive NLPCA and forward and backward selection. We've implemented RDESF through a mix of the Apache Commons library and an optimized use of data structures in Java. The speed can be further improved by the use of multi-threading, with every filter scoring the features simultaneously.

4 Conclusion

A novel filter-class algorithm for automated selection of features has been proposed. Our algorithm called Ranked Distinct Elitism Selection Filter (RDESF) was tested in two experiments. The filter-class of feature selection are model-free and thereby fast. A big drawback of filters is the requirement of choosing a threshold to select the features. This problem was overcome by creating a meta-filter that selects the top 10 % features for each included filter. To avoid duplicate features a distinct selection was performed, in this case a union selection.

The experiments in which RDESF was tested were time-series regression based problems of prediction a variable. RDESF was only outperformed by forward selection which was expected, because the prediction model is included in the forward selection process. All the other feature selection algorithms which were: Backward selection, NLPCA and PCA were all outperformed by RDESF. In the first experiment RDESF even outperformed using all input features, that indicates a good performance in computationally heavy types of ANNs. The computational speed of RDESF was only outperformed by using all input features which was also expected, since no pre-calculations are required.

RDESF clearly outperformed PCA in both prediction error and computational speed, which was our benchmark baseline. Unlike PCA, RDESF allows for feature analysis in fault detection scenarios because the features are not transformed. This means that RDESF is a strong competitor in the feature selection field, and applicable to a variety of application areas. The meta-filter approach does not require the user to select a threshold for the filter which allows the user to include filters into an automatic feature selection process or system.

References

1. Cavanaugh, J.E.: Unifying the derivations for the akaike and corrected akaike information criteria. Stat. Probab. Lett. **33**(2), 201–208 (1997)
2. Cover, T.M., Thomas, J.A.: Elements of Information Theory. Wiley, New York (1991)
3. Granger, C.: Some recent development in a concept of causality. J. Econometrics **39**, 199–211 (1988)
4. Guttman, L.: Some necessary conditions for common-factor analysis. Psychometrika **19**(2), 149–161 (1954)
5. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3**, 1157–1182 (2003)
6. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5), 359–366 (1989)
7. Kramer, M.A.: Nonlinear principal component analysis using autoassociative neural networks. AIChE J. **37**(2), 233–243 (1991)
8. Lichman, M.: UCI machine learning repository (2013). http://archive.ics.uci.edu/ml
9. Nautical Almanac Office: Almanac for Computers 1990. U.S Government Printing Office (1989)
10. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In: IEEE International Conference on Neural Networks, pp. 586–591. IEEE (1993)
11. May, R., Dandy, G., Maier, H.: Review of Input Variable Selection Methods for Artificial Neural Networks. InTech, April 2011
12. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by backpropagating errors. Cogn. Model. **5**, 1 (1988)
13. Shanmugan, K.S., Breipohl, A.M.: Random Signals: Detection, Estimation, and Data Analysis. Wiley, New York (1988)
14. Shannon, C.E.: A mathematical theory of communication. SIGMOBILE Mob. Comput. Commun. Rev. **5**(1), 3–55 (2001)
15. Shlens, J.: A tutorial on principal component analysis. arXiv preprint arXiv:1404.1100 (2014)
16. Wang, P.G., Mikael, S., Nielsen, K.P., Wittchen, K.B., Kern-Hansen, C.: Reference climate dataset for technical dimensioning in building, construction and other sectors. DMI Technical reports (2013)
17. Whiteson, S., Stone, P., Stanley, K.O., Miikkulainen, R., Kohl, N.: Automatic feature selection in neuroevolution. In: GECCO (2005)
18. Zamora-Martínez, F., Romeu, P., Botella-Rocamora, P., Pardo, J.: On-line learning of indoor temperature forecasting models towards energy efficiency. Energy Build. **83**, 162–172 (2014)
19. Zhang, G., Patuwo, B.E., Hu, M.Y.: Forecasting with artificial neural networks: the state of the art. Int. J. Forecast. **14**(1), 35–62 (1998)

Learning Using Multiple-Type Privileged Information and SVM+ThinkTank

Ming Jiang[1] and Li Zhang[2(✉)]

[1] Faculty of Applied Sciences, Sunderland University, Sunderland, UK
Ming.Jiang@sunderland.ac.uk
[2] Faculty of Engineering and Environment, Northumbria University, Newcastle, UK
Li.Zhang@northumbria.ac.uk

Abstract. In this paper, based on the extension to the standard Support Vector Machines Plus (SVM+) model for the Learning Using Privileged Information (LUPI) paradigm, a new SVM+ThinkTank (SVM+TT) model, is proposed for Learning Using Multiple-Type Privileged Information (LUMTPI). In cases that Multiple-Type Privileged Information (MTPI) from different perspectives is available for interpreting those training samples, such as in Big Data Analytics, it could be beneficial to leverage all these different types of Privileged Information collectively to construct multiple correcting spaces simultaneously for training the maximum margin based separating hyperplane of SVM model. In fact, from the practical point of view, organising Privileged Information from different perspectives might be a relatively easier task than finding a single type perfect Privileged Information for those hardest training samples. The MTPI collectively plays the role of a think tank as the single type Privileged Information plays the role of a single perfect master class teacher. The preliminary experimental results presented and analysed in this paper demonstrate that SVM+TT, as a new learning instrument for the proposed LUMTPI paradigm, is capable of improving the generalisation ability of the standard SVM+ in learning from a small amount of training samples but incorporating with the MTPI interpretations to these samples for improved learning performance.

Keywords: Support Vector Machines · Multiple-Type Privileged Information · Statistical Machine Learning · Classification

1 Introduction

As a new training strategy by leveraging extra interpretation information to improve the generalisation ability of machine learning in the cases of learning from small amount of data samples, Learning Using Privileged Information (LUPI) paradigm [1, 2] has gained increasing popularity in recent years [3–7]. At the expense of using extra information, the LUPI paradigm uses the Privileged Information to guide the training of a classifier to make better distinctions between those hard and easy training samples during the training and hence improve the prediction accuracy of the classifier during the testing.

In this paper, we investigate how Learning using Multiple-Type Privileged Information (LUMTPI) could improve the prediction accuracy of SVM+ for a binary classification problem. In order to incorporate Multiple-Type Privileged Information (MTPI) simultaneously into a learning process, we first extend the standard SVM+ model for single type Privileged Information to support the adjustment of hyperplane based on a collective optimisation both in the common decision space with technical data and in the multiple correcting spaces with MTPI simultaneously. The extended model is named SVM+ThinkTank (SVM+TT) as an analogy of the collective contribution of a think tank to the decision making process of a complex problem. In this paper, as a case study, the SVM+TT model is implemented with the support of two types Privileged Information and evaluated with real-world binary classification problems. The key contributions of the paper are as follows:

- *A new paradigm of Learning using Multiple-Type Privileged Information (LUMTPI)* is proposed to improve the generalisation ability of supervised learning model for cases of learning with a small amount of training samples but with rich interpretation information to these samples;
- *A new learning instrument of SVM+TT* is specified theoretically and implemented with numerical optimisation programming to implement the LUMTPI paradigm.
- *A case study* of the applying the SVM+TT instrument and the LUMTPI paradigm into a binary classification in breast cancer patients is used to evaluate the performance of the new instrument and paradigm. Experimental results demonstrate the effectiveness and efficiency of the new approach.

2 Multiple-Type Privileged Information and SVM+ ThinkTank

In this section, the basic models of SVM [8] and SVM+ [1] for classification problems are extended to devise a Multiple-Type Privileged Information (MTPI) based SVM +ThinkTank for the case of a classical binary classification problem.

2.1 Binary Classification Problem Using SVM

In the classical setup of a binary classification problem by using a standard SVM, there are two finite subsets of vector x from the training set:

$$(x_1, y_1), \ldots, (x_l, y_l), \qquad x \in R^n, \qquad y \in \{+1, -1\} \tag{1}$$

where there is one subset with $y = +1$ and another subset with $y = -1$. In order to find the decision rule $y = f(x)$ to separate the two subsets, SVM first maps the vector x_i of space X into vector z_i of space Z where it constructs the optimal separating hyperplane by minimising the functional:

$$R(w,b,\xi) = \frac{1}{2}(w,w) + c\sum_{i=1}^{l}\xi_i \tag{2}$$

subject to the constraints:

$$\begin{gathered} y_i[(w,z_i)+b] \geq 1 - \xi_i,\ i = 1,\ldots,l \text{ and} \\ \xi_i > 0, \end{gathered} \tag{3}$$

where (w, w) is the margin size of the separating hyperplane, ξ_i represents the deviation of z_i from the margin border, and C is the user defined model regularisation parameter, which balances the size of margin and number of acceptable non-separable training sample vectors. The generalisation ability of the decision rule $y = f(x)$, in terms of prediction accuracy, can be tested with testing samples.

2.2 Binary Classification Problem Using SVM+ThinkTank

In the case of Multiple-Type Privileged Information (MTPI), there are two finite subsets of vector x from the training set:

$$\begin{gathered} (\mathrm{x}_1, \mathrm{x}_1^1, \ldots, \mathrm{x}_1^{\mathrm{t}}, \mathrm{y}_1), \ldots, (\mathrm{x}_1, \mathrm{x}_1^1, \ldots, \mathrm{x}_1^{\mathrm{t}}, \mathrm{y}_1) \\ \mathrm{x} \in \mathrm{R}^{\mathrm{n}}, \quad \mathrm{y} \in \{+1, -1\} \end{gathered} \tag{4}$$

here $x_1^1, \ldots, x_1^t$ are the different types of privileged information which are only available during the training process, and there is one subset with $y = +1$ and another subset with $y = -1$.

In order to leverage the MTPI for training the classifier to control the slack variables of all sample data in its decision space, multiple correcting spaces should be constructed and used to assist the classifier to make a better distinction between a hard example and easy one in that decision space. Therefore, a new SVM+ThinkTank (SVM +TT) model is proposed to incorporate one common decision space and multiple correcting spaces together to train the classifier. By extending the objective function (2), a generalised objective function will be:

$$R(w, w_1, \ldots, w_t) = \frac{1}{2}(w,w) + \frac{1}{2\gamma}\sum\nolimits_{\mathrm{r}=1}^{\mathrm{t}}(w_r, w_r) + \frac{c_r}{\mathrm{t}}\sum\nolimits_{r=1}^{t}\sum\nolimits_{i=1}^{l}\xi_i^r, \tag{5}$$

where

$$\xi_i^r = (w_r, z_i^r) + b_r^i \tag{6}$$

subject to constrains:

$$\begin{gathered} y_i[(w,z_i)+b] \geq 1 - \xi_i^r, \\ i = 1,\ldots,l, r = 1,\ldots,t, \xi_i^r > 0 \end{gathered} \tag{7}$$

Without loss of generality, let us consider two types (i.e., t = 2) Privileged Information x^* and x^{**} in the problem setup of binary classification on two finite subsets of vector x from the training set:

$$(x_1, x_1^*, x_1^{**}, y_1), \ldots, (x_l, x_l^*, x_l^{**}, y_l),\ x \in R^n,\ y \in \{+1, -1\} \tag{8}$$

where the Privileged Information is only available during the training process and there is one subset with $y = +1$ and another subset with $y = -1$.

The proposed SVM+TT first maps the vector x_i of space X into vector z_i of space Z, x_i^* into vector z_i^* in Z^* space, and x_i^{**} into vector z_i^{**} in Z^{**} space where it constructs the optimal separating hyperplane by minimising the functional:

$$R(w, w^*, w^{**}, b, b^*, b^{**}) = \frac{1}{2}[(w, w) + \gamma((\mathrm{w}^*, \mathrm{w}^*) + (\mathrm{w}^*, w^{**}))] + \frac{C_1}{2}\sum_{i=1}^{l}\xi_i + \frac{C_2}{2}\sum_{i=1}^{l}\xi_i^* \tag{9}$$

$$\xi_i = (w^*, z_i^*) + b^* \tag{10}$$

$$\xi_i^* = (w^{**}, z_i^{**}) + b^{**} \tag{11}$$

subject to the constraints:

$$\begin{gathered} y_i[(w, z_i) + b] \geq 1 - \xi_i, \\ y_i[(w, z_i) + b] \geq 1 - \xi_i^*, \\ i = 1, \ldots, l, \xi_i > 0, \xi_i^* > 0 \end{gathered} \tag{12}$$

where γ is an additional model regularisation parameter. Similarly, as being transformed in SVM+, by constructing the Lagrangian of (9) and replacing the inter products of (z_i, z_j), (z_i^*, z_j^*) and (z_i^{**}, z_j^{**}) in the common decision space and two correcting spaces with the corresponding kernel functions respectively, the minimisation of functional (9) is transformed to maximise functional:

$$\begin{aligned} R(\alpha, \beta, \pi) = \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{l}\alpha_i\alpha_j y_i y_j K(x_i, x_j) \\ - \frac{1}{2\gamma}[\sum_{i,j=1}^{l}(\alpha_i + \beta_i - C_1)(\alpha_j + \beta_j \\ - C_1)K^*(x_i^*, x_j^*) \\ + \sum_{i,j=1}^{l}(\alpha_i + \pi_i - C_2)(\alpha_j + \pi_j \\ - C_2)K^{**}(x_i^{**}, x_j^{**})] \end{aligned} \tag{13}$$

in which there are two model regularisation parameters C_1 and C_2 applied to the two correcting spaces respectively and the maximisation is subject to constraints:

$$\sum_{i=1}^{l} y_i \alpha_i = 0, \quad \sum_{i=1}^{l} (\alpha_i + \beta_i - C_1) = 0, \quad \sum_{i=1}^{l} (\alpha_i + \pi_i - C_2) = 0, \quad i = 1, \ldots, l \text{ and } \alpha_i \geq 0,\ \beta_i \geq 0,\ \pi_i \geq 0 \tag{14}$$

and the decision function is

$$f(x) = \sum_{i=1}^{l} y_i \alpha_i K(x_i, x) + b \tag{15}$$

3 Experiments

3.1 Experiment Dataset

The proposed SVM+TT is implemented based on the Ph.D. work in [9] and evaluated with the UCI breast cancer dataset which consists of 286 instances, each with 9 attributes*: men, inv, nod, deg, bre, qua, irr, size, age* [10]. There are two Classes: *No-recurrence-events* and *Recurrence-events*. Since, among the 286 instances, there are 9 instances with missing attribute values, in our experiments, we use the remaining 277 instances with the complete attribute values.

3.2 Experiment Setups

We use a linear kernel to map the vectors of non-privileged features into the common decision space, and a RBF kernel and a polynomial kernel to map the two types of vectors of different privileged features into the different correcting spaces respectively. For the purpose of simulating the effect of using Privileged Information, two particular original features are selected as the Privileged Information: *inv-nodes* is selected as the type I privileged feature and *age* is selected as the type II privileged feature. The parameters for model/kernel selection are the following:

- Model regularisation parameters: C_1 and C_2 = [0.1, 1, 1.5]
- Correcting space capacity control parameter: *gamma* = [1, 10]
- RBF Kernel parameter: *sigma* = [0.25, 0.5]
- Polynomial Kernel parameter: the order of polynomial, *d* = [2, 5]

The prediction accuracies of standard SVM, standard SVM+ and SVM+TT are evaluated and compared. The experiments for the three different scenarios are set up as follows:

SVM: An experiment without any privileged information, i.e., both training and testing with the non-privileged features only.

SVM+: Experiments with either type I or type II privileged feature, i.e., training with non-privileged features and with either type I or type II privileged feature together, but testing with non-privileged features only. In this scenario, regarding the combinations of the use of privileged features and use of kernels in correcting spaces, there are further four different cases as combined with two different kernels of RBF or Polynomial.

SVM+TT: Experiments with both type I and type II privileged information together, i.e., training with the non-privileged features and type I and type II privileged features together, but testing with non-privileged features only. In this scenario, regarding the combinations of the use of privileged features and use of kernels in correcting spaces, there are four different cases as combined with two different kernels of RBF or Polynomial.

The model selection uses a 5-fold cross-validation and the final prediction accuracy of the selected model is calculated based on the average accuracies of 5 runs.

3.3 Experimental Results and Analysis

Based on the experiment setups in Sect. 3.2, the final comparison results show an average of prediction accuracy rates and its standard deviation of each test case configuration.

The result highlighted in bold in the Table 1 demonstrates that (1) by selecting the appropriate correcting space kernels for different privileged information, the generalisation ability of SVM+ is better than the standard SVM and (2) by further tuning the appropriate model regularisation parameters *C₁* and *C₂* with different combinations of privileged information and correcting space kernels, the generalisation ability of SVM +TT could be even better than SVM+. The reason of the further generalisation ability improvement of SVM+TT could be that the Multiple-Type Privileged Information is able to complement each other on correcting the classifier collectively. This promising results confirm that SVM+TT, as a new learning instrument for the proposed LUMTPI paradigm, is capable of improving the generalisation ability of SVM+ in learning from a small amount of training samples but with multiple-type interpretation information to these samples. Since both of the non-privileged features and privileged ones are the subset of the same feature set that represents the original training data, the privileged

Table 1. SVM, SVM+ and SVM+TT prediction accuracy comparisons

SVM type	Pri. Inf. and kernel	Predication accuracy	
SVM	Non-privileged	**0.696558 ± 0.048373**	
SVM+	Case1: **Attri2 + RBF**	**0.740065 ± 0.041396**	
	Case2: Attri2 + POLY2	0.613377 ± 0.162487	
	Case3: **Attri9 + RBF**	**0.747078 ± 0.035489**	
	Case4: Attri9 + POLY2	0.670649 ± 0.143768	
SVM+TT	Case1 Attri2 + POLY2 & Attri9 + RBF	0.699870 ± 0.107690 (C1 = 0.1, C2 = 1)	0.639675 ± 0.206775 (C1 = 0.1, C2 = 1.5)
	Case2 Attri2 + RBF & Attri9 + POLY2	0.743701 ± 0.054642 (C1 = 0.1, C2 = 1)	0.681818 ± 0.122263 (C1 = 0.1, C2 = 1.5)
	Case3 **Attri2 + RBF & Attri9 + RBF**	**0.754221 ± 0.059289** (C1 = 0.1, C2 = 1)	**0.754416 ± 0.052648** (C1 = 0.1, C2 = 1.5)
	Case4 Attri2 + POLY2 & Attri9 + POLY2	0.668636 ± 0.167369 (C1 = 0.1, C2 = 1)	0.671883 ± 0.129460 (C1 = 0.1, C2 = 1.5)

and original feature spaces belong to the same data modality. In future, information from different data modalities such as the narratives extracted and featured from diagnosis reports, could be explored and as the Privileged Information to validate and demonstrate the possible further generalisation performance improvements.

4 Related Work

Regarding the use of multiple correcting spaces/kernels in parallel of a single decision making space in SVM and SVM+, there are two most related works, which are quite close to but different from our SVM+TT model. In this section, these works are introduced and discussed briefly. In [1], Multiple Space Privileged Information (MSPI) are provided for the different part of the training samples, i.e., the different subsets of training samples are equipped with different privileged information which is drawn from multiple correcting space/kernels. The difference between this MSPI approach and our Multiple Type Privileged Information (MTPI) approach is that we apply each type of Privileged Information on the entire set of training samples rather than on the individual subsets, i.e., the entire set of the training samples are equipped with different Privileged Information from different perspectives collectively in multiple times. Of course, our MTPI approach is based on the assumption that Multiple Type Privileged Information is available for all the training samples in the first place.

Another closely related to SVM+ and SVM+TT but quite different work is the use of SVM for Multi-Task Learning Problem (SVM+MTL) [11], in which the training samples are divided into groups and the group labels are used as the extra information and mapped into different correcting spaces. The difference between the extra group labels information and Privileged Information is that the group labels are available on the future test samples, i.e., group labels are not privileged for training samples only. It is demonstrated that in case that the number of training samples per group is sufficiently large, the generalisation ability of SVM+MTL is better than SVM and SVM+ [11]. However, one of the purposes and basic assumptions of SVM+ and SVM+TT is that only a small amount of training samples are available and hence extra Privileged Information is used to speed up training convergence and improve the generalisation ability of a classifier. It is also worthwhile to note that the SVM+MTL approach has a similarity to the regularized multi-task learning technique (rMTL) [12], however, for the rMTL approach, the decision space and correcting space are the same.

Finally, there are works of applying SVM+ with Privileged Information onto multiple classification problems [13] and studying the relation of SVM+ to weighted SVM [14]. However, these work only focus on using a single type of Privileged Information for the training.

5 Conclusions

This paper proposes an innovative SVM+ThinkTank (i.e. SVM+TT), as a new learning instrument, to realise the new Learning Using Multiple-Type Privileged Information paradigm. The preliminary experimental results promisingly demonstrate that

Multiple-Type Privileged Information, used with appropriate model and regularisation parameter selection, is able to further improve the generalisation ability of a classifier collectively. In future work, we aim to focus on evaluating the synergy strategies [15] of Privileged Information from different data modalities, investigating the principles and guidelines of model and regularisation parameter selection [16] for large scale machine learning applications and speeding and scaling up the training of the SVM+TT with parallel computing and scale machine learning technologies.

References

1. Vapnik, V., Izmailov, R.: Learning using privileged information: similarity control and knowledge transfer. J. Mach. Learn. Res. **16**, 2023–2049 (2015)
2. Vapnik, V., Izmailov, R.: Learning with intelligent teacher. In: Gammerman, A., Luo, Z., Vega, J., Vovk, V. (eds.) COPA 2016. LNCS, vol. 9653, pp. 3–19. Springer, Heidelberg (2016). doi:10.1007/978-3-319-33395-3_1
3. Wang, S., Yang, J.: Learning to transfer privileged ranking attribute for object classification. J. Inf. Comput. Sci. **12**(1), 367–380 (2015)
4. Liu, J., Zhu, W., Zhong, P.: A new multi-class support vector algorithm based on privileged information. J. Inf. Comput. Sci. **10**(2), 443–450 (2013)
5. Sharmanska, V., Quadrianto, N., Lampert, C.: Learning to Transfer Privileged Information, CoRR, October 2014
6. Sharmanska, V., Quadrianto, N., Lampert, C.: Learning to rank using privileged information. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV 2013), 1–8 December 2013
7. Serra-Toro, C., Traver, V., Pla, F.: Exploring some practical issues of SVM+: is really privileged information that helps? Pattern Recogn. Lett. **42**, 40–46 (2014)
8. Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)
9. Cai, F.: Advanced Learning Approaches Based on SVM+ Methodology. Ph.D. thesis, the University of Minnesota, July 2011
10. https://archive.ics.uci.edu/ml/datasets/Breast+Cancer
11. Liang, L., Cherkassky, V.: Connection between SVM+ and multi-task learning. In: Proceedings of the IJCNN (2008)
12. Evgeniou, T., Pontil, M.: Regularized multi-task learning. In: Proceedings of the 17th SIGKDD Conference on Knowledge Discovery and Data Mining (2004)
13. Ji, Y., Sun, S., Lu, Y.: Multitask multiclass privileged information support vector machines. In: Proceedings of the ICPR, pp. 2323–2326. IEEE (2012)
14. Lapin, M., Hein, M., Schiele, B.: Learning using privileged information: SVM+ and weighted SVM. Neural Netw. (2014)
15. Zhang, Y., Zhang, L., Hossain, A.: Adaptive 3D facial action intensity estimation and emotion recognition. Expert Syst. Appl. **42**(3), 1446–1464 (2015)
16. Zhang, Y., Zhang, L., Neoh, S.C., Mistry, K., Hossain, A.: Intelligent affect regression for bodily expressions using hybrid particle swarm optimization and adaptive ensembles. Expert Syst. Appl. **42**(22), 8678–8697 (2015)

Learning Symbols by Neural Network

Yoshitsugu Kakemoto[1(✉)] and Shinichi Nakasuka[2]

[1] The JSOL, Ltd., Harumi Center Building, 2-5-24 Harumi, Chuo-ku, Tokyo, Japan
kakemoto@yf6.so-net.ne.jp

[2] The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan

Abstract. VSF−Network is a neural network model that learns dynamical patterns. It is hybrid neural network combining a chaos neural network and a hierarchical neural network. The hierarchical neural network learns patterns and the chaos neural network monitors behavior of neurons in the hierarchical neural network. In this paper, two theoretical backgrounds of VSF−Network are introduced. An incremental learning framework using chaos neural networks is introduced. The monitoring by chaos neural network is based on clusters generated by synchronous vibration. Using the monitoring results, redundant neurons in the hierarchical neural network are found and they are used for learning of new patters. The second background is about the pattern recognition by combining learned patterns. This is explained by code words expression used in multi-level discrimination. Through an experiment, both the incremental learning capability and the pattern recognition are shown.

1 Introduction

When robots autonomously act under a certain environment, they need an intelligent information process mechanism such as learning patterns from surrounded environments. Reasoning based on symbol are a basic component of intelligent information processing and symbol can be used in subsequent intelligent information processing. It is a kind of pattern generated by abstracting signals from outside world but it different in that symbol can be individually used or used in combining with other symbols based on situation.

Since the early 2000's, symbol learning for robots studied. Inamura [1] proposed a model of stochastic behavior recognition and symbol learning. Kadone [2] proposed symbolic representation learning model for a humanoid robot. On these studies, symbol and relations between symbols are learned based on cooccurrences of actions. In semiotics, many theories about symbol have been also proposed. Chandler [3] proposed the model about the double articulation of semiotic codes. Semiotic codes belong to any class, that is, single articulation, double articulation or no articulation. The double articulation enables a semiotic code to form an infinite number of meaningful combinations using a small number of low-level elements.

In any of these position, the common matter is that symbol is not only abstracted pattern, but also symbol can have a mutual relationship. We proposed

VSF-network, Vibration and Synchronize Function network, as a neural network model for abstracting input data and learning patterns used as symbol [4]. In this paper, theoretical backgrounds of VSF-network are introduced. Especially, we focus on its ability for incremental learning and combinational utilization of learned symbols as important properties of symbol.

2 Incremental Learning by Chaos Neural Network

2.1 Incremental Learning

For neural networks, the learning of symbol is an instance of incremental learning [5] because neural networks incrementally learn unlearned patterns while remembering learned patterns. On the incremental learning, correlations between patterns take an important role. Lin and Yao [6] proposed Negative Correlation Leaning as an incremental learning model for neural networks. In the proposed method, the neural network incrementally learns with increasing neurons depending on correlations between an unlearned pattern and learned patterns. If we can estimate an appropriate number of neurons before learning, it is reasonable to increase the number of neurons according to progress of learning. Even if a natural network completed a learning, there are redundant neurons in a natural network and the over-fitting problems occur. This problem worsens with an increase of the number of neurons. A technique to the problem is reusing neurons for incrementally learning unlearned patterns, because neural networks frequently have redundant neurons that do not participate pattern recognition by a neural network. The unlearned pattern has low correlation to learned patterns. VSF−Network learns unlearned patterns with reusing redundant neurons.

2.2 Pattern Recognition by Chaos Neural Network

Hopfield [7] showed that associative memory has attractors and several attractors reach equilibrium points, if its weight matrix is symmetric and its activation function is a monotonic increasing function. Kanode [2] theoretically analyzed relations between patterns and attractors on Hopfield type associative memory. He concluded that some learned patterns get attractors, if a correlation between a presented pattern and a learned pattern is high. On the other hand, the associative memory can not get any attractor and it equilibrates at an averaged pattern of learned patterns, if the correlation between an unlearned pattern and learned patterns is low. This result shows that retrievals reach different statuses, depending on correlations between patterns. Therefore, we can discriminate unlearned parts in an input from learned patterns for a neural network by monitoring retrieval process of an associative memory. We analyze each equilibrium point of neuron in an associative memory to find unused neurons with analyzing chaotic oscillation on an associative memory.

Aihara [8] proposed Chaos Neural Network, CNN, to show chaotic retrieval dynamics in addition to normal dynamics of associative memory. The chaotic dynamics can be found, if an input pattern do not match a part of pattern to learned patterns. An output x_i of neuron i in CNN that has M input neurons and N chaotic neurons is defined by (1).

$$x_i(t+1) = f\left[\xi_i(t+1) + \eta_i(t+1) + \zeta_i(t+1)\right] \tag{1}$$

In (1), ξ_i is an input term, η_i is an interaction term and ζ_i is an inhibitory term for a neuron i. They are defined as follows,

$$\xi_i(t+1) = k_s \xi_i(t) + \sum_{j=1}^{N} v_{ij} A_j(t), \tag{2}$$

$$\eta_i(t+1) = k_n \eta_i(t) + \sum_{j=1}^{N} w_{ij} x_j(t), \tag{3}$$

$$\zeta_i(t+1) = k_r \zeta_i(t) - \alpha x_i(t) - \theta(1-k_r). \tag{4}$$

In (2), $A(t)$ is input at a time t, and v_{jk} is the connection weight between the k the element of the input and the neuron j. In (3), w_{ij} is the connection weight between chaos neuron i and chaos neuron j, α is an inhibitory strength parameter and θ_i is a threshold for the inhibitory term in (4). k_s, k_n, and k_r are a parameter for each term. f is Sigmoid function (5)

$$f(x,a) = \frac{1}{1+e^{-ax}}. \tag{5}$$

2.3 Correlation on Chaos Neural Network

An internal status u_i of CNN neuron i changes periodically. Some neurons show periodical oscillations with other neurons. This is called a synchronized oscillation. CNN (1) is thought at an instance of GCM, Globally Coupled Mapping [9], because it is a system that statuses of neuron i diffuses to other neurons through η term. Komuro [10] studied dynamics on GCM and analyzed behavior of elements in GCM. GCM is defined by (6) on $\mathbb{R}^N$ $(N \geq 1)$. In (6), F is a diagonal matrix whose diagonal elements are a logistic function $f_{a,\varepsilon}(x_i)$ parameterized by a, ε.

$$\begin{aligned} F_{a,\varepsilon} &: \mathbb{R}^N \to \mathbb{R}^N, \\ x = (x_1, \cdots, x_N)^T &\mapsto y = (y_1, \cdots, y_N))^T, \\ y_i = (1-\varepsilon) f_a(x_i) &+ \frac{\varepsilon}{N} \sum_{j=1, j\neq i}^{N} f_a(x_j),\ (1 \leq i \leq N) \end{aligned} \tag{6}$$

On GCM, elements show synchronized oscillation with other elements by retracting or phase locking [10]. The synchronized element is called a cluster or a synchronized oscillating group.

An invariant manifold $U_\sigma = \left\{x \in \mathbb{R}^N; x_i = x_{\sigma(i)}, 1 \leq i \leq N\right\}$ is a sub manifold of a manifold U^N that a coordinate permutation P_σ is invariant. σ

is a cyclic permutation $\sigma = \left(i_{11}, \cdots, i_{1m_1}\right) \cdots \left(i_{k1}, \cdots, i_{km_k}\right)$. σ_k is cluster $k(k = 1, \cdots, n)$ and m_k is an index of elements in a cluster k. For any $\sigma \in U_\sigma$, an orthogonal complement $U_\sigma^\perp$ is invariant to $F_{a,\varepsilon}{}'(x)$. A dynamics of each cluster is measured by traversal Lyapunov exponent. Traversal Lyapunov exponent is Lyapunov exponent from U_σ to $U_\sigma^\perp$. Traversal Lyapunov exponents depend on the eigenvalue value of $F_{a,\varepsilon}(x)$ on $U_\sigma^\perp$ and that is $(1-\varepsilon)\, f_a{}' \left(x_{i_{j1}}\right), \quad (j = 1, \cdots, k)$. Let u_μ be a status of neuron i for a learned pattern μ. u_ν is a status of neuron i for an unlearned pattern ν. Further, $u_{\nu,a}$ is a part of u_ν that has high correlation with u_μ and $u_{\nu,c}$ is a part of u_ν that has low correlation with u_μ. For $u_{\nu,a}$ and $u_{\nu,c}$, we respectively define the cluster σ_a and σ_c. The eigenvalue of U_{σ_a} is $F\left(x_i\right)' u_a = (1-\varepsilon)\, f_a{}' \left(x_i\right) u_a$, and the eigenvalue of U_{σ_c} is $F_{a,\varepsilon}\left(x_i\right)' u_c = (1-\varepsilon)\, f_a{}' \left(x_i\right) u_c$. From the definition $u_{\nu,a}$ and $u_{\nu,c}$, they get a stable equilibrium points on U_{σ_a} but that is not always the case for u_{σ_c} Therefore, the eigenvalue of U_{σ_a} shows a stable status but the eigenvalue of U_{σ_c} dose not show a stable status.

In retrieval process of CNN, high correlated neurons retract to other neurons. On the other hand, low correlated neurons do not retract. As the result, the low correlated elements show isolate behavior. Therefore, role of a neuron to recognize patterns is determined through the status of each neurons based on the retrieval process of CNN.

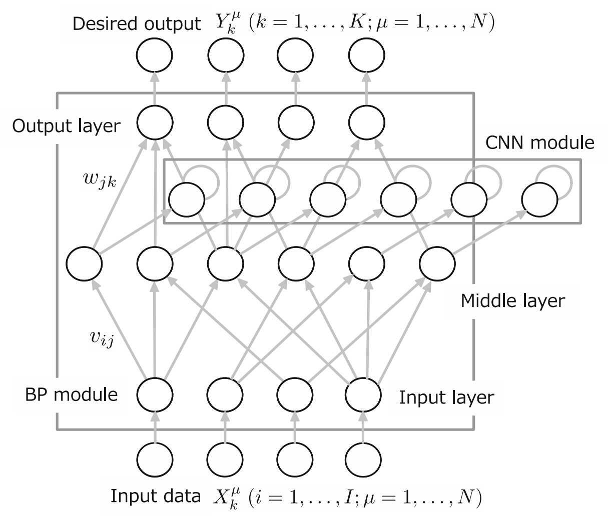
2.4 Incremental Learning by VSF−Network

An overview of VSF-Network is shown in figure Fig. 1. VSF−Network is composed of BP−module, Back-Propagation module and CNN−module, Chaos Neural Network module. BP−module is a hierarchical neural network and the number of the layers is three. It learns relations between input and output to close to an expected output. CNN−module monitors statuses of BP−module. It finds out redundant neurons in the middle layer of BP−module and unlearned elements of an input data. The results of the monitoring are used for updating the weights between layers of BP−module. When CNN−module retrieves a pattern from a status of middle layer in BP−module, neurons that exist on a cluster is regarded as used neuron and the connection weights associated with the neuron are not updated. Neurons that do not exist on any cluster are regarded as redundant neuron and the connection weights associated with the neuron are updated. By this selective weight updating, BP−module is divided into sub-networks at the middle layer level and each sub-network correspond to each learned patter.

3 Recognition of Combinational Patterns

3.1 Oscillated Firing and Pattern Combination

When VSF−Network uses its learning results, all learned patterns are not required for a recognition of a pattern. For example, if VSF−Network learned patterns μ and ν, then it should return an output for a pattern μ for an input



Desired output Y_k^μ $(k = 1, \ldots, K; \mu = 1, \ldots, N)$
Output layer
CNN module
w_{jk}
Middle layer
v_{ij}
BP module
Input layer
Input data X_k^μ $(i = 1, \ldots, I; \mu = 1, \ldots, N)$

Fig. 1. An overview of VSF−Network

that the pattern μ is presented. In the same way, it returns an output for a pattern ν, if the pattern ν is presented. It should return outputs the pattern that patters μ and ν are combined, if the patterns μ and ν are presented in an input. Use of suitable patterns depending on situations is a basic requirement of the pattern as symbol. The convolution neural network [11] learns partial patterns at the middle layers and recognizes more complex pattern by combining these partial patterns. For the developing VSF−Network, we take this technique to recognize patterns by a combination of learned patterns.

3.2 Expression of Patterns Combination

After incremental learning, we can find several sub-networks on BP-module. Every sub-network is specialized for recognizing specific patterns or pattern sets. Therefore, encoded patters on the middle layer of every sub-network can be considered as code words specialized for each pattern.

We can explain the recognition by patterns combination on VSF-Network based on the code words. For the developing NETtalk [12], Sejnowski proposed an expression of pattern in neural network by code words. For example, statuses of the middle layer of BP−module which learned patterns μ and ν can be expressed as show in Table 1. The cord words shown in Table 1 are an extended expression of ECOC [13]. The code word u_x $(x = \mu, \nu, \xi, \cdots)$ shows that sub-network f_i $(i = 1, \cdots, n)$ learns a pattern in a class $c_x (x = \mu, \nu, \xi, \cdots)$.

As shown in Table 1, the number of classes and combinations of patterns increase in accordance with increase the number of learning patterns. On the other hand, the number of classes is equal to the number of sub-networks, if each pattern is independently recognized by each sub-network. Let U be a

Table 1. An example of code words table

Class	Sub-network			
	f_1	f_2	$\cdots$	f_n
c_μ	$u_{\mu\wedge\nu}$	u_*	$\cdots$	u_*
c_ν	u_*	u_ν	$\cdots$	u_*
$c_{\mu\wedge\nu}$	u_μ	u_ν	$\cdots$	u_*
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

space spanned by a hierarchical neural network and the forward pass between the input layer and the middle layer is a mapping $f : U \to U$. For a learned hierarchical neural network, we can define an invariant space Λ for f. For Λ and its open neighborhood T, there is attractors and trapping areas that satisfy the conditions $f(\bar{T}) \in T$ and $\bigcap_{i=0}^{\infty} f^i(\bar{T}) = \Lambda$. Futhermore, there are basins $B(T) = \{x \in U : f^M(x) \in T\}$ of the attractors for large enough number $M > 0$. The basins exit on each sub-space $U_i \ (i = 1, \ldots, p)$ spanned by learned sub-network $f_i \ (i = 1, \ldots, p)$. Let $B_i \ (i = 1, \ldots, p)$ be a basin on U_i. If each sub-network f_i is an independent hierarchical neural network, each sub-space U_i is orthogonal to others and

$$U = U_1 \oplus \cdots \oplus U_i, i = 1, \ldots, p.$$

In this case, when patters corresponding to each sub-network f_μ and $f_\nu (\mu \neq \nu)$, each basin $B_{\mu\wedge\nu}$ exists on the sub-space $U_{\mu\wedge\nu} = U_\mu \oplus U_\nu$ and $B_{\mu\wedge\nu} = B_\mu \cap B_\nu$. As the result, the pattern μ and ν are simultaneously recognized.

In Table 1, u_* indicates statuses that neurons in the middle layer are not used or learned patters are not found in current input. We should distinguish between the former case and the later case. These two statuses should be distinguished. In recognizing phase, neurons in sub-networks output constant values if learned patters are not found in the current input. These values should not have any effect on pattern recognition for the current input, namely, they should be identity for the current output. The most suitable identity value is 0. Because of relationship between the nature of the activation function and input data, values to be $\simeq 0$ are obtained by adjusting threshold value or inclination coefficient in the case of Sigmoid function (5). If neurons in the middle layer are not used in the learning phase, they do not belong any cluster in the processing at CNN−module. In the recognizing phase, these neuron output averaged value for learned pattern and they are not trapped any basins. As the result, they can not retrieve any code word, and they do not affect current output.

4 Selective Weight Updating

4.1 Learning Procedure

VSF−Network is developed for incremental learning, therefore we assume that their connection weights are trained using other learning tasks. The initial connection weight of CNN−module are calculated from the connection weights between the input layer and the middle layer of BP−module at each learning. The learning procedure of VSF−Network is summarized as follows.

1. Learning data for a pattern μ is input to the input layer of BP−module.
2. The outputs from the middle layer of BP−module are input to CNN−module as a key pattern for retrieval process in CNN−module.
3. CNN−module retrieve the pattern corresponding to every input. This process is based on (1) for times $t((=1,\cdots,T))$.
4. The forwarding values between the middle layer and the output layer of BP−module is calculated. At this point, the results of CNN−module are not used.
5. Error $E_{k,\mu}$ between the output $\hat{Y^{\mu}}$ from BP-module and the expected output Y^{μ} for the pattern μ is calculated.
6. The connection weights of BP−module are updated by the weight update rule (7) and (8).

4.2 Selecting Weights Updating

An input of CNN−module are outputs from the middle layer of BP−module. As the result, VSF−Network finds redundant neurons at the middle layer that do not have any role in pattern recognition. CNN−module finds redundant neurons that show asynchronous oscillation. Update amount for connection weights for neurons that show asynchronous oscillation is larger than amount for synchronous oscillated neurons. Therefore, VSF−Network can learn unlearned patterns without forgetting learned patterns.

The update value $\mathit{\Delta} W_{ij}$ for the weight between the i-th neuron in the input layer and the j-th neuron in the middle layer in BP−module is defined as,

$$\begin{aligned} \mathit{\Delta} W_{ij} &= \begin{cases} \eta \frac{\partial E_{ij}}{\partial W_{ij}} & (\lambda_i \le P) \\ 0 & (\lambda_i > P) \end{cases} \\ \frac{\partial E_{ij}}{\partial W_{ij}} &= (1.0 - |cor_{i,j}|) \sum_{j=1}^{n} \frac{\partial E_{jk}}{\partial W_{jk}} f'\left(H_i^{\mu}\right). \end{aligned} \tag{7}$$

In (7), η is a coefficient for update, H_i^{μ} is an output from jth neuron in the middle layer. λ_i is a synchronous rate calculated with (9) and P is a threshold for λ_i. cor_{k_i,k_j} is a correlation between neuron i and j in the middle layer. Our weight updating scheme is based on the correlations between patterns, therefore we apply the correlations of weights between the input layer and the middle layer. These have an effect to amplify the results from CNN-module.

The update value $\varDelta W_{jk}$ for the weight between the j-th neuron in the middle layer and the k-th neuron in the output layer is defined as

$$\begin{aligned} \varDelta W_{jk} &= \begin{cases} \eta \frac{\partial E_{jk}}{\partial W_{jk}} & (\lambda_i \leq P) \\ 0 & (\lambda_i > P) \end{cases} \\ \frac{\partial E_{jk}}{\partial W_{jk}} &= \sum_{j=1}^{n} E^{\mu} f' \left(Y_k^{\mu}\right) . \end{aligned} \tag{8}$$

In (8), E^{μ} is an error between a current output and $\hat{Y^{\mu}}$ and an expected output Y^{μ} for it.

The synchronous rate λ_i is calculated by correlation integration (9) based on Heaviside function H. In (9), r is a threshold for a difference $x_i - x_j$.

$$\lambda_i = \frac{1}{T^2} \sum_{j=1, i \neq j}^{N} H(r - |x_i - x_j|) \tag{9}$$

5 Experiment and Result

In this section, we show capabilities of VSF−Network through an experiment. The task for the experiment is a learning of avoiding behavior for obstacles by a rover. The rover learns whether it can avoid obstacles or not when it begins turning to the left at a current place. If the rover can avoid the obstacle, the output is = 1 otherwise the output is = 0. We prepare three conditions for the obstacle configurations. The condition 1 is a T-Junction obstacle condition, the condition 2 is a simple obstacle condition and the condition 3 is a combined obstacle condition. An overview of these conditions is shown in Fig. 2.

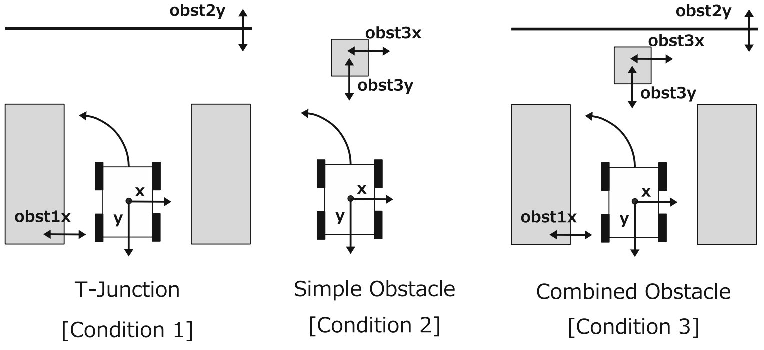


Fig. 2. Configurations of Rover and Obstacles

Before our main experiment, we determined a value of each parameter of CNN−module trough a numerical experiment. In Table 2, parameters and their

Table 2. Parameters and values for CNN−module

Parameter	Description	Value
k_s	Coefficient for the input term (2)	0.95
k_n	Coefficient for the interaction term (3)	0.1
k_r	Coefficient for the inhibitory term (4)	0.95
α	Inhibitory coefficient for (4)	2.0
θ	Threshold for (4)	0.2
τ_{hst}	Time span of periodic phase	100
P	Threshold about λ_i for applying weight update rule	0.01
Θ	Threshold for Hviside function in (9)	0.01

Table 3. Parameters for BP−module

Parameter	Description	Value
ε	Coefficient of activation function	1.0
η	Weight updating rate	0.1

values are shown. Each value is determined for CNN−module to show a weak chaotic behavior. Other parameters and values for BP−module at the experiment are shown in Table 3.

The initial weights for BP−Module is obtained from other trained hierarchical neural network. The hierarchical neural network learned 1500 times of epochs and its topology is same as BP−modele. Each epoch consists of 2000 times of training and the data are randomly selected from the data set of condition 1. At the incremental learning step, the number of epochs and training are same as the first step. The input data for this step are randomly selected from data set of the condition 2. To examine an ability of VSF−Network for recognizing combined type pattern, we provide data randomly selected from data set of the condition 3. Data of the condition1 and the condition 2 are provided at this step to examine other ability of VSF−Network for incremental learning. On this step, we compare MSE, Mean Squired Error, at each epoch for each data set.

In Fig. 3, we show the changes of MSE according to the progress of the incremental learning. For incremental learning, the effect of VSF−network is obvious. VSF-Network incrementally learns unlearned patterns and the weights for the condition 1 are not destroyed. The combined patterns can be recognize without learning with the progress of incremental learning. After certain number of learning, MSE for each condition reaches at an equilibrium status and the learning stops when there is no redundant neuron because VSF−Network incrementally learns by reusing redundant neurons.

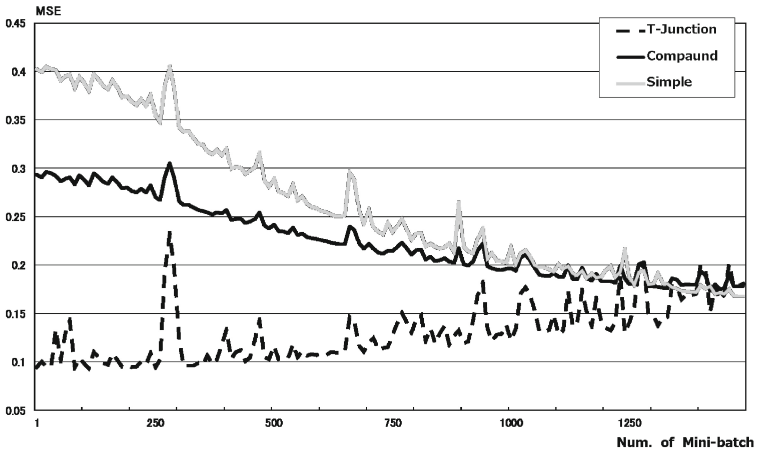


Fig. 3. Learning progress of VSF−Network

6 Conclusion

In this paper, we show the theoretical backgrounds of VSF−Network. Through the experiment, VSF−Network show good performance to incremental learning. The combined patterns are also recognized well without any learning of combined pattern.

Now, we have three research tasks for VSF−Network. The first task is to expand kinds of experiment task to examine additional abilities of VSF-Network. The second task is application of the deep learning scheme to VSF−network. The current BP−module has 3 layers. We expects an improvement of its ability by introducing a deep learning scheme. Other task is sophistication of relation between learned patterns by VSF−Network. Several related studies about symbol learning showed learning of high level relation between symbols such as hierarchy of symbols. For VSF−Network, such a high level relations can be introduced by a more detailed analysis about dynamicses on CNN−module.

References

1. Inamura, T., Tanie, H., Nakamura, Y.: Proto-symbol development and manipulation in the geometry of stochastic model for motion generation and recognition. Technical Report NC2003-65, IEICE (2003)
2. Kadone, H., Nakamura, Y.: Symbolic memory of motion patterns using hierarchical bifurcations of attanctors in an associative memory model. J. Robot Soc. Jpn. **25**, 249–258 (2007)
3. Chandler, D.: Semiotics for Beginners. Routledge, London (1995)
4. Kakemoto, Y., Nakasuka, S.: Neural assembly generation by selective connection weight updating. In: Proceedings of IjCNN 2010 (2010)

5. Giraud-Carrier, C.: A note on the utility of incremental learning. AI Commun. **13**, 215–223 (2000)
6. Lin, M., Tang, K., Yao, X.: Incremental learning by negative correlation leaning. In: Proceedings of IJCNN 2008 (2008)
7. Hopfield, J.: Neurons with graded response have collective computational properties like those of two-stage neurons. Proc. Nat. Acad. Sci. U.S.A. **81**, 13088–3092 (1984)
8. Aihara, T., Tanabe, T., Toyoda, M.: Chaotic neural networks. Phys. Lett. **144A**, 333–340 (1990)
9. Kaneko, K.: Chaotic but regular posi-nega switch among coded attractor by cluster size variation. Phys. Rev. Lett. **63**, 219 (1989)
10. Komuro, M.: A mechanism of chaotic itinerancy in globally coupled maps. In: Dynamical Systems (NDDS 2002) (2002)
11. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. Neural Comput. **1**, 541–551 (1989)
12. Sejnowski, T.J., Rosenberg, C.R.: Nettalk: a parallel network that learns to read aloud. In: Anderson, J.A., Rosenfeld, E. (eds.) Neurocomputing: Foundations of Research, pp. 661–672. MIT Press, Cambridge (1988)
13. Dietterich, T.G., Bakiri, G.: Solving multiclass learning problems via error-correcting output codes. J. Artif. Intell. Res. **2**, 263–286 (1995)

A CPM-Based Change Detection Test for Big Data

Giada Tacconelli and Manuel Roveri(✉)

Politecnico di Milano, Piazza L. da Vinci 32, 20133 Milano, Italy
giada.tacconelli@mail.polimi.it, manuel.roveri@polimi.it

Abstract. Big data analytics nowadays represent one of the most relevant and promising research activities in the field of Big Data. Tools and solutions designed for such purpose are meant to analyse very large sets ot data to extract relevant/valuable information. In this path, this paper addresses the problem of sequentially analysing big streams of data inspecting for changes. This problem that has been extensively studied for scalar or multivariate datastreams, has been mostly left unattended in the Big Data scenario. More specifically, the aim of this paper is to introduce a change detection test able to detect changes in datastreams characterized by very-large dimensions (up to 1000). The proposed test, based on a change-point method, is non parameteric (in the sense that it does not require any apriori information about the system under inspection or the possible changes) and is designed to detect changes in the mean vector of the datastreams. The effectiveness and the efficiency of the proposed change detection test has been tested on both synthetic and real datasets.

1 Introduction

In recent years the pervasive dissemination of sensors and devices (e.g., cyber-physical systems, Internet-of-things technology and mobile devices) led to the availability of a tremendous and ever-growing amount of collected data [13]. Such a *Big Data* revolution made commonly-used techniques for data acquisition, managing and processing inadequate and led to the design of novel tools and solutions able to deal with such an enormous amount of data.

Among the wide range of technological and scientific issues related to Big Data, the ability to make sense out of such a huge amount of data and exploit its value is increasingly gaining importance. This is where Big Data Analytics come into play, where novel techniques and mechanisms are designed to analyse (possibly very) large amounts of data to extract relevant information [5,7,15]. Often such large amounts of data arrive in a stream manner (e.g., collected by large-scale IoT systems, sensor networks or acquired through on-line systems/services), hence requiring a sequential analysis to inspect for events/information of interest.

The aim of this paper is to address the problem of detecting changes in Big Data scenarios, where massive amounts of data steadily arrive over time. The detection of changes affecting streams of Big Data is extremely important

© Springer International Publishing AG 2017
P. Angelov et al. (eds.), *Advances in Big Data*, Advances in Intelligent Systems and Computing 529, DOI 10.1007/978-3-319-47898-2_11

since it could reveal critical situations, such as ageing effects and faults affecting sensing/control apparatus, anomalous events/behaviours, or an unforeseen evolution of system under inspection. Hence, the prompt detection of such changes is essential for undertaking suitable countermeasures like repairing/replacing a sensor, raising an alarm, or activating adaptation mechanisms. While the problem of detecting changes in scalar datastreams has been extensively explored (e.g., see [3]), very few works have been proposed for multivariate datastreams (e.g., see [11]) and, to the best of our knowledge, the problem of sequentially detecting changes in Big Data has never been addressed in the literature. A review of related literature is presented in Sect. 2.

The aim of this paper is to propose a Change Detection Test (CDT) able to detect changes in Big Data. The proposed test is non-parametric, hence not requiring any a-priori information about the data-generating process or the change to operate, and able to manage datastreams characterized by very large dimensions (up to 1000). More specifically, the proposed CDT extends a multivariate Change-Point Method (CPM) [14], i.e., a theoretically-grounded statistical hypothesis test able to verify and locate the presence of a change-point in a fixed-length sequence of data, by considering sliding windows of data and a modification of the Hotellings T^2 statistic to detect changes in the mean vector of the datastream.

The proposed CDT for Big Data has been validated on both synthetic and real datasets characterized by large dimensions and made available to the scientific community as a MATLAB Demo Toolbox[1]. In addition, the computational times of the proposed CDT has been evaluated for different dimensions of the datastreams.

The paper is organized as follows. Section 2 critically reviews the related literature, while Sect. 3 formulates the change-detection problem. The proposed CDT is detailed in Sect. 4. The experimental analysis is shown in Sect. 5, while conclusions are finally drawn in Sect. 6.

2 Related Literature

CDTs are statistical techniques aiming at assessing the stationary of a data generating process over time. CDTs typically operate by inspecting independent and identically distributed (i.i.d.) features extracted from the considered data stream (e.g., the empirical error of the model, the sample mean or variance), building an initial hypothesis and checking the validity of such an assumption over time. Differently from statistical hypothesis tests or CPMs that require a fixed dataset to operate, CDTs are meant to sequentially analyse streams of data. While the majority of existing CDTs have been designed for univariate distributions [3], several multivariate solutions have been presented in the literature. These multivariate solutions can be divided into three main families according to

[1] The Matlab Demo Toolbox of the proposed CDT can be found at the following url: http://roveri.faculty.polimi.it/software-and-datasets/.

the amount of information they require to operate: parametric, semi-parametric and non-parametric.

Parametric CDTs require the a-priori knowledge of the system before and after the change. Examples of CDTs belonging to this family are the Hotellings T^2, multivariate CUSUM and multivariate exponentially weighted moving average control charts (MEWMA) [11]. *Semi-parametric CDTs* require partial information about the system or the change to operate (e.g., the knowledge of the probability density function of the system before the change). Examples of semi-parametric CDTs include the hierarchical Bayesian approach proposed by [1]; the log-likelihood criterion (SPLL) introduced in [9] that has been tested with up to 60 dimensions and two procedures based on the likelihood ratio test (LRT) statistic and on a cumulative sums (cusum) statistic developed by [6]. In addition, a system of semi-parametric online learners based on random walks and a semi-parametricneural-fuzzy model for detecting mean shifts have been recently introduced in [8,12], respectively.

The availability of (at least partial) a-priori information about the system or the change is an assumption that rarely holds in real-world scenarios, making parametric and non-parametric solutions not a viable solution. To overcome this limitation, *non-parametric CDTs* have been introduced, hence not requiring any a-priori information about the system or the change. Unfortunately, very few non-parametric CDTs have been proposed for multivariate datastreams [2,10]. The *CI-CUSUM* [2] extends the traditional CUmulative-SUM test by extracting a set of features that are approximately Gaussian distributed and by autonomously learning the nominal and the possible changes. Differently, the Rank-Based multivariate CUSUM procedure [10] is based on the analysis of the cross-sectional antiranks of the observations. To the best of our knowledge, non-parametric CDTs meant to operate in Big Data scenarios have never been proposed in the literature.

3 Problem Formulation

Let $X = \{x(t), t = 1, 2, ...\}$, $x(t) \in \mathbb{R}^p$ be a sequence of i.i.d. p-variate observations coming from a data generating process with unknown probability density function (pdf) Φ_0. A change in X can be modelled as a transition from Φ_0 to a different and a-priori unknown pdf Φ_1 at an unknown time instant t_0 as follows:

$$x(t) \sim \begin{cases} \Phi_0, \, if \ t < t_0 \\ \Phi_1, \, if \ t \geq t_0 \end{cases} \tag{1}$$

This is a quite general and widely assumed data/change model in change detection [3]. When the i.i.d. assumption on acquired data does not hold, suitably-defined pre-processing techniques (e.g., fitting linear/nonlinear regression models and computing the residuals) could be used to extract i.i.d. features from not-i.i.d. datastreams. The goal of the proposed CDT is to promptly detect the occurrence of changes in X, while not introducing false positive (i.e., detections before t_0) or false negative detection (i.e., changes that are not detected by CDT).

4 The Proposed CPM-Based Change Detection Test

The proposed CDT, which is summarized in Fig. 1, operates as follows. It processes $x(t) \in \mathbb{R}^p$ with $t = 1, 2, \ldots$ that are assumed to follow a multinormal p-variate distribution. When needed, data $y(t)$s acquired by the data-generating process are transformed into approximately normal-distributed random variables $x(t)$s through a suitable preprocessing phase (e.g., based on the Multivariate Central Limit theorem). The CDT sequentially analyses $x(t)$ over time inspecting for changes its mean vector. Remarkably, the CDT does not require any a-priori information about the mean vector before or after the change, nor about the covariance matrix of $x(t)$s. In order to operate the CDT requires a threshold that depends on p and the expected Average Run Length (ARL), i.e., the average number of samples to be processed in stationary conditions (i.e., when Φ_0 holds) before a false positive detection occurs [3]. Hence, during an initial configuration phase, the user selects the expected ARL (together with p) and the corresponding threshold is provided to the CDT as parameter.

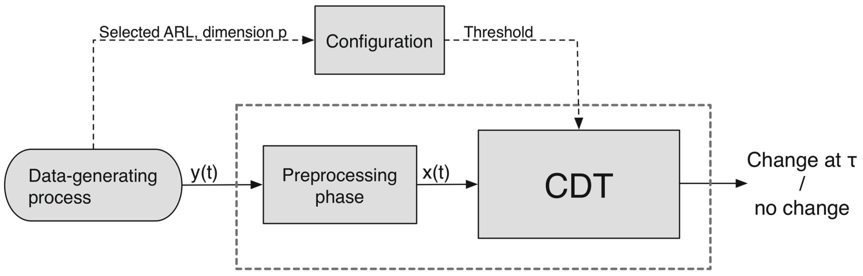


Fig. 1. The workflow of the proposed change detection test.

The algorithm of the proposed CDT is detailed in Algorithm 1. In more detail, the proposed CDT is inspired by a multivariate CPM [14], whose goal is to find the presence of a change-point in the mean vector of a sequence of data characterized by a fixed length (hence not operating in a streaming fashion). We extended that CPM to operate sequentially on $x(t)$s by considering two windows of acquired data W_1 and W_2^t. The first window refers to an initial training sequence for the CDT, which is assumed to be change-free, while the second one represents a sliding window on the recently-acquired values of $x(t)$. Without loss of generality we set the length of W_1 and W_2^t to be proportional to p, i.e., $|W_1| = |W_2^t| = \lceil \gamma p \rceil$, where $|\bullet|$ is the length operator, $\lceil \bullet \rceil$ is the ceiling operator and $\gamma > 1$ is a user-defined parameter. Summing up, $W_1 = \{x(1), \ldots, x(\gamma p)\}$ and $W_2^t = \{x(t - \gamma p + 1), \ldots, x(t)\}$.

Algorithm 1. The proposed Change Detection Test for Big Data

```
1: input: $\left[\{x(1),...,x(\gamma p)\}, h_{\gamma,\alpha,p}\right]$;
2: $W_1 = \{x(1),\dots,x(\gamma p)\}$;
3: Compute $S$ on $W_1$;
4: while (1) do
5:     Acquire $x(t)$;
6:     $W_2^t = \{x(t-\gamma p+1),\dots,x(t)\}$;
7:     $W = [W_1, W_2^t]$;
8:     for $\tau := \gamma p+1 \rightarrow 2\gamma p$ do
9:         $A_\tau = \{W(i), i=1,\dots,\tau\}$;
10:        $B_\tau = \{W(i), i=\tau+1,\dots,2\gamma p\}$;
11:        Compute $\bar{X}_{1,\tau}$ and $\bar{X}_{\tau+1,2\gamma p}$;
12:        Compute $Y_\tau$ as in (3);
13:        Compute $T_\tau^2$ as in (2);
14:        if $T_\tau^2 \geq h_{\gamma,\alpha,p}$ then
15:            Detection at time $t$;
16:            break;
17:        else“no change found”;
18:        end if
19:    end for
20: end while
```

At each time instant t, we define $W = [W_1, W_2^t]$ by juxtaposing W_1 and W_2^t. Then, the proposed CDT operates as follows: it divides W into subsequences

$$A_\tau = \{W(i), i=1,\dots,\tau\},$$
$$B_\tau = \{W(i), i=\tau+1,\dots,2\gamma p\},$$

with $\gamma p+1 \leq \tau \leq 2\gamma p$ and compute the modified Hotelling statistic $\overline{T}_\tau^2$ for A_τ,B_τ as follows

$$\overline{T}_\tau^2 = (Y_\tau)'(S^{-1})(Y_\tau), \tag{2}$$

where

$$Y_\tau = [\tau(2\gamma p-\tau)/2\gamma p]^{1/2}(\bar{X}_{1,\tau} - \bar{X}_{\tau+1,2\gamma p}) \tag{3}$$

and $\bar{X}_{1,\tau}$, $\bar{X}_{\tau+1,2\gamma p}$, S^{-1} are the sample mean vector of A_τ, the sample mean vector of B_τ and the sample covariance matrix estimated on W_1, respectively. We emphasize that, differently from the traditional Hotelling statistic [14], in our case S does not depend on τ. Hence it can be computed only once during the initial configuration of the CDT and it is not recomputed during the operational life, thus reducing the computational complexity of the proposed CDT. With this modification, we are implicitly assuming that the covariance of W_1 and W_2 are equal both before and after the change: an assumption that does not limit the performance of our CDT since it focuses on detecting changes in the mean vector.

The CDT computes the values of $\overline{T}_\tau^2$ for $\tau = \{\gamma p+1,\dots,2\gamma p\}$. Hence, $\overline{T}_\tau^2$ is computed only on W_2^t since there is no need to analyse W_1 that is assumed to be change-free. A change is detected at time t by the CDT when the maximum of the statistic (2) on $\gamma p+1 \leq \tau \leq 2\gamma p$ is larger than a threshold $h_{p,\gamma,\alpha}$, as follows

$$\max_{\tau=\gamma p+1,\ldots,2\gamma p} \overline{T}_{\tau}^{2} > h_{p,\gamma,\alpha}. \tag{4}$$

Threshold $h_{p,\gamma,\alpha}$s depends on p, γ and α, where α is a confidence parameter accounting for the probability of having a false positive detection by the CDT on a fixed data-sequence whose length is $2\gamma p$. Since an analytical derivation of the maximum of (2) is hard to obtain, we resort on simulations to compute such thresholds $h_{p,\gamma,\alpha}$s. In Appendix we detail the procedure and the computed thresholds and corresponding ARLs for different values of p and α and $\gamma = 1.5$.

5 Experimental Results

The aim of this section is to analyse the change detection ability of the proposed CDT in both small and large-dimension datastreams. In more detail, the effectiveness of the proposed CDT has been contrasted with the Rank-based procedure and evaluated on very-large normally-distributed synthetic datasets (with $p = 100$, 300, 500 and 1000) and a real dataset (with $p = 300$).

To measure the ability to promptly and correctly detect changes in large-dimension datastreams, we consider the following three figures of merits:

1. False positive (FP): it measures the times a change is detected, while it is not (percentage).
2. False negative (FN): it measures the times a change is not detected, while it is present (percentage).
3. Detection Delay (DD): it measures the time delay between when the change occurred and when it is detected (in number of samples).

FP, FN and DD are averaged over 500 iterations. In our experimental analysis we set $\gamma = 1.5$ and considered the following change model for the synthetic experiments:

$$x(t) \sim \begin{cases} \mathcal{N}(\mu, \Sigma), & \text{if } t < 2\gamma p + 100 \\ \mathcal{N}(\mu, \Sigma) + \Delta, & \text{if } t \geq 2\gamma p + 100 \end{cases} \tag{5}$$

where $\mu \in \mathbb{R}^p$ and $\Sigma \in \mathbb{R}^{p\times p}$ are randomly generated at each iteration and $\Delta \in \mathbb{R}^p$ is the additive perturbation inserted at time $t_0 = 2\gamma p + 100$ to be detected. The length of the training sequence W_1 is γp samples.

5.1 Comparison Between the Proposed CDT and the Rank-Based Procedure

We initially compared the detection abilities of the proposed CDT and the Rank-Based procedure present in the literature (and commented in Sect. 2). We opted for the Rank-Based procedure since, similarly to the proposed CDT, it is non-parametric. Unfortunately, this procedure has been designed for multivariate analysis on relatively small dimensions and parameter configurations for Big Data scenarios are not available. For this reason we considered $p = 4$, while two different kinds of changes have been defined:

1. $\mathit{\Delta}_1 = [-25, 0, \dots, 0]$: modelling the case where only one dimension is affected by a change;
2. $\mathit{\Delta}_2 = [-25, -25, \dots, -25]$: modelling the case where all the dimensions are affected by a change.

The Rank-based procedure has been configured to achieve DDs comparable with the ones provided by the proposed CDT.

The comparison of detection abilities is presented in Table 1. As we can see the proposed CDT outperforms the Rank-based procedure both in terms of FP/FN and DD. As expected, the Rank-based procedure is not able to detect changes that affect all the dimensions in the same way (i.e., FNs are very large in $\mathit{\Delta}_2$) due to the rank-based analysis. Differently, the proposed CDT is able to effectively detect changes affecting both just a single dimension and all the dimensions.

Table 1. Comparison between the proposed and the Anti-Rank CDT.

Dataset	Proposed CDT			Anti-rank CDT		
	FPs	FNs	DD	FPs	FNs	DD
$\mathit{\Delta}_1$	1.33	0	1.33	63.0	0.0	12.0
	0.66	0	1.86			
$\mathit{\Delta}_2$	1.33	0	1.0	65.0	29.0	6.0
	0.66	0	1.05			

5.2 Synthetic Experiments

We evaluated the detection abilities of the proposed CDT on synthetic experiments characterized by dimensions ranging from $p = 100$ to $p = 1000$. Here, we considered the following two kinds of change

1. $\mathit{\Delta}_3 = [\delta, 0, \dots, 0]$: modelling the case where only one dimension is affected by the additive term δ;
2. $\mathit{\Delta}_4 = [0.1\delta, 0.1\delta, \dots, 0.1\delta]$: modelling the case where all the dimensions are affected by the additive term 0.1δ,

where $\delta = \max(diag(S))$ and $diag(\bullet)$ is the operator extracting the elements of the main diagonal of matrix $\bullet$. The experimental results, which are detailed in Table 2, show the high effectiveness of the proposed CDT in promptly detecting changes (i.e., the DD is smaller that 15 samples in all the configurations and changes), while guaranteeing very low FPs and FNs for all the values of p.

Table 2. Synthetic experiments with dimensions ranging from $p = 100$ to $p = 1000$.

Δ	p	α	FPs	FNs	DD
Δ_3	100	0.1	2.4	0	6.06
	100	0.01	0.6	0	7.75
	300	0.1	0.8	0	7.72
	300	0.01	0	0	10.41
	500	0.1	1.3	0	8.29
	500	0.01	0	0	13.33
	1000	0.1	2.6	0	8.21
	1000	0.01	0	0	11.58
Δ_4	100	0.1	2.0	0	6.28
	100	0.01	0.6	0	7.97
	300	0.1	0.6	0	3.06
	300	0.01	0	0	3.96
	500	0.1	2.0	0	2.15
	500	0.01	0	0	2.62
	1000	0.1	2.6	0	1.39
	1000	0.01	0	0	1.62

As expected, a smaller value of α reduces FPs at the expenses of a slight increase of DDs.

In addition, we also evaluated the execution times of the proposed CDT w.r.t. p. More specifically, Fig. 2 shows the time (in seconds) required to process each new observation with p ranging from 100 to 1000[2]. It is possible to appreciate that the execution times remain acceptable even with very large dimensions.

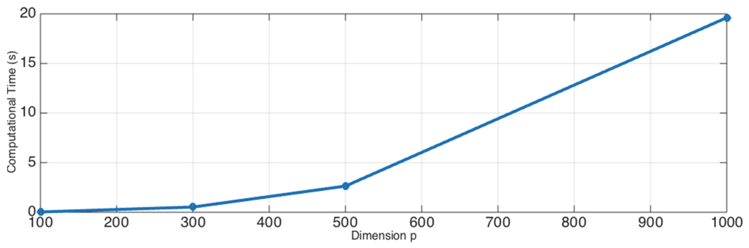


Fig. 2. Execution time (in seconds) required to process each new acquired observation by the proposed CDT w.r.t. the number of dimensions p.

[2] The computing platform is based on a 1,7 GHz Intel Core i5 with 4 GB 1333 MHz DDR3.

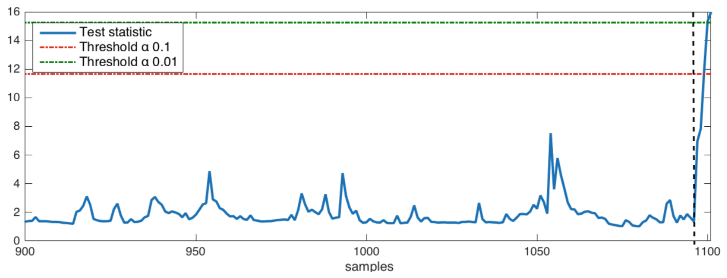


Fig. 3. The computed test statistic on the Mutant p53 proteins dataset.

5.3 Real Dataset

The proposed CDT has been validated on a real dataset, i.e., the biophysical models of mutant p53 proteins [4] available on the UCI website for classification tasks. It is composed of 16592 instances (143 belonging to active class and 16449 to inactive class) and 5408 attributes. We organised the dataset appending all the observations of the active class at the end of the inactive samples to model a change in the distribution as per Eq. (1). In our experiments we considered the first $p = 300$ features and we applied a preprocessing phase (i.e., computing the sample mean on non-overlapping windows of 15 samples for each attribute) since the original data distribution is not multinormal. This led to a 300-dimension dataset composed by 1106 samples. The first $\gamma p = 450$ samples are used to configure the CDT, while $t_0 = 1097$ (samples between 451 and 900 are used to fill W_2^t at time $t = 900$). Figure 3 reports the behaviour of the test statistic in Eq. (4) over time (solid blue line) and two different thresholds have been considered, i.e., those related to $\alpha = 0.1$ (red dotted line) and $\alpha = 0.01$ (green dotted line). As it can be seen, the test statistic overcomes both thresholds at $t = 1101$ for both thresholds (hence with a DD of four samples), without introducing FPs. This highlights the effectiveness of the proposed CDT in detecting changes in real and not-normal highly-dimension datastreams.

6 Conclusions

The aim of this paper was to introduce a CDT able to operate in Big Data scenarios. More specifically, the proposed test is able to operate without requiring any apriori information about the system under inspection or the possible changes and is able to manage datastreams characterize by very large dimensions (up to 1000). The performance of the proposed CDT has been contrasted with the rank-base procedure and tested with both synthetic and real datasets.

Future works will encompass the characterization of the probability of false positive detections of the proposed CDT, hence providing a additional “confidence” information when a change in the datastream is detected.

Appendix

As described in Sect. 4, thresholds $h_{p,\gamma,\alpha}$s have been computed through simulations since an analytical derivation of the test statistic in (4) is hard to obtain. Thresholds have been computed as follows: for each value of p and α and γ, we simulated 10,000 experiments in which we randomly generated a p-variate normal distribution with random mean vector and covariance matrix. The threshold $h_{p,\gamma,\alpha}$ is set to guarantee that the empirical probability of having a false positive detection by the proposed CDT on a fixed data-sequence whose length is $2\gamma p$ is equal to the confidence parameter α. Computed thresholds for different values of p and α and $\gamma = 1.5$ are shown in Table 3. Further values of $h_{p,\gamma,\alpha}$ for different configurations of p and α and γ can be found at the following url: http://roveri.faculty.polimi.it/software-and-datasets/.

Table 3. Thresholds $h_{p,\gamma,\alpha}$ for different values of p and α and $\gamma = 1.5$.

	α	$p = 100$	$p = 300$	$p = 500$	$p = 1000$
$h_{p,\gamma,\alpha}$	0.1	11.420	11.664	11.831	12.187
	0.01	14.675	15.252	15.659	15.528

To further clarify the effects of $h_{p,\gamma,\alpha}$ in the sequential scenario, in Table 4 we detail the empirically estimated ARL for $\alpha = 0.1$, $\gamma = 1.5$ and p ranging from 100 to 1000.

Table 4. *ARL* for different values of p, $\alpha = 0.1$ and $\gamma = 1.5$.

	$p = 100$	$p = 300$	$p = 500$	$p = 1000$
ARL	5907.1	3331.8	3514.5	4112.0

References

1. Agarwal, D.: An empirical bayes approach to detect anomalies in dynamic multidimensional arrays. In: Fifth IEEE International Conference on Data Mining, 8-pp. IEEE (2005)
2. Alippi, C., Roveri, M.: Just-in-time adaptive classifierspart i: detecting nonstationary changes. IEEE Trans. Neural Netw. **19**(7), 1145–1153 (2008)
3. Basseville, M., Nikiforov, I.V., et al.: Detection of Abrupt Changes: Theory and Application, vol. 104. Prentice Hall, Englewood Cliffs (1993)
4. Danziger, S.A., Swamidass, S.J., Zeng, J., Dearth, L.R., Lu, Q., Chen, J.H., Cheng, J., Hoang, V.P., Saigo, H., Luo, R., et al.: Functional census of mutation sequence spaces: the example of p53 cancer rescue mutants. IEEE/ACM Trans. Comput. Biol. Bioinform. (TCBB) **3**(2), 114–125 (2006)

5. Ferreira, L.N., Zhao, L.: A time series clustering technique based on community detection in networks. Procedia Comput. Sci. **53**, 183–190 (2015). INNS Conference on Big Data 2015, San Francisco, CA, USA, 8–10 August 2015
6. Galeano, P., Peña, D.: Covariance changes detection in multivariate time series. J. Stat. Plann. Infer. **137**(1), 194–211 (2007)
7. Hajj, N., Rizk, Y., Awad, M.: A mapreduce cortical algorithms implementation for unsupervised learning of big data. Procedia Comput. Sci. **53**, 327–334 (2015)
8. Hegedűs, I., Nyers, L., Ormándi, R.: Detecting concept drift in fully distributed environments. In: 2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics (SISY), pp. 183–188. IEEE (2012)
9. Kuncheva, L.I.: Change detection in streaming multivariate data using likelihood detectors. IEEE Trans. Knowl. Data Eng. **25**(5), 1175–1180 (2013)
10. Qiu, P., Hawkins, D.: A rank-based multivariate cusum procedure. Technometrics **43**(2), 120–132 (2012)
11. Sullivan, J.H., Woodall, W.H.: Change-point detection of mean vector or covariance matrix shifts using multivariate individual observations. IIE Trans. **32**(6), 537–549 (2000)
12. Wang, T.Y., Chen, L.H.: Mean shifts detection and classification in multivariate process: a neural-fuzzy approach. J. Intell. Manufact. **13**(3), 211–221 (2002)
13. Wu, X., Zhu, X., Wu, G.Q., Ding, W.: Data mining with big data. IEEE Trans. Knowl. Data Eng. **26**(1), 97–107 (2014)
14. Zamba, K., Hawkins, D.M.: A multivariate change-point model for statistical process control. Technometrics **48**(4), 539–549 (2006)
15. Zikopoulos, P., Eaton, C., et al.: Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data. McGraw-Hill Osborne Media, New York (2011)

Hadoop MapReduce Performance on SSDs: The Case of Complex Network Analysis Tasks

Marios Bakratsas[1], Pavlos Basaras[1], Dimitrios Katsaros[1,2(✉)], and Leandros Tassiulas[2]

[1] Department of Electrical and Computer Engineering, University of Thessaly, Volos, Greece
mmpakrat@gmail.com, pbasaras@gmail.com, dkatsar@inf.uth.gr

[2] Department of Electrical Engineering and Yale Institute for Network Science, Yale University, New Haven, USA
leandros.tassiulas@yale.edu

Abstract. This article investigates the relative performance of SSDs versus hard disk drives (HDDs) when they are used as underlying storage for Hadoop's MapReduce. We examine MapReduce tasks and data suitable for performing analysis of complex networks which present different execution patterns. The obtained results confirmed in part earlier studies which showed that SSDs are beneficial to Hadoop; we also provide solid evidence that the processing pattern of the running application plays a significant role.

1 Introduction

Processing of modern Online Social Networks on a single machine (centralized) is doomed to fail due to lack of resources. The Hadoop instead was designed to solve problems where the "same, repeated processing" had to be applied to peta-scale volumes of data. Hadoop's initial design was based on magnetic disk characteristics. With the advent of Solid State Drives (SSDs) research is emerging to test/exploit the potential of the new technologically advanced drive [4, 8]. The lack of seeking overhead gives them a significant advantage with respect to Hard Disk Drives (HDDs) for workloads whose processing requires random access instead of sequential access. Providing a clear answer to the question of whether SSDs significantly outperform or offer increased performance in same cases compared to HDDs in the Hadoop environment is not straightforward, because the results of a system-analysis-based investigation are affected by the network speed and topology, by the cluster (size, architecture) and by the nature of the benchmarks used (MapReduce algorithms, input data). This article starts the investigation from a new basis and attempts to provide a clear answer to the following question [7]: *Ignoring any network biases and storage media cost considerations, do SSDs provide improved performance over HDDs for real workloads that are not dominated by either reads or writes?*

2 Related Work

Investigating the usage of SSDs in Hadoop clusters has been a hot issue of discussion very recently. The most relevant work to ours is included in the following articles [4, 5, 8, 9, 11]. The first effort [5] to study the impact of SSDs on Hadoop was on a virtualized cluster (multiple Hadoop nodes on a single physical machine) and showed up to three times improved performance for SSDs versus HDDs. However, it remains unclear whether the conclusions still hold in non-virtualized environments. The work in [8] compared Hadoop's performance on SSDs and HDDs on hardware with non-uniform bandwidth and cost using the Terasort benchmark. The major finding is that SSDs can accelerate the shuffle phase of MapReduce. However, this work is confined by the very limited type of application/workload used to make the investigation and the intervention of data transfers across the network. Cloudera's employees in [4], using a set of same-rack-mounted machines (not reporting how many of them), focus on measuring the relative performance of SSDs and HDDs for equal-bandwidth storage media. The MapReduce jobs they used are either read-heavy (Teravalidate, Teraread, WordCount) or network-heavy (Teragen, HDFS data write), and the Terasort which is read/write/shuffle "neutral". Thus, neither the processing pattern is mixed nor the network effects are neutral. Their findings showed that SSD has higher performance compared to HDD, but the benefits vary depending on the MapReduce job involved, which is exactly where the present study aims at [7].

The analysis performed in [9] using Intel's HiBench benchmark [2] concluded that "… the performance of SSD and HDD is nearly the same", which contradicts all previously mentioned works. A study of both pure (only with HDDs or only with SSDs) and hybrid systems (combined SSDs and HDDs) is reported in [11] using a five node cluster and the HiBench benchmark. In contrast to the current work, the authors in [11] investigated the impact of HDFS's block size, memory buffers, and input data volume on execution time. The results illustrated that when the input data set size and/or the block size increases, the performance gap between a pure SSD system and a pure HDD system widens in favor of the SSD. Moreover, for hybrid systems, the work showed that more SSDs result in better performance. These conclusions are again expected since voluminous data imply increased network usage among nodes. Earlier work [3, 10] studied the impact of interconnection on Hadoop performance in SSDs identifying bandwidth as a potential bottleneck. Finally, some works propose extensions to Hadoop with SSDs. For instance, VENU [6] is a proposal for an extension to Hadoop that will use SSDs as a cache (of the HDDs) not for all data, but only for those that are expected to benefit from the use of SSDs. This work still leaves open the question about how to tell which applications are going to benefit from the performance characteristics of SSDs.

3 Investigated Algorithms

Complex network analysis comprises a large set of diverse tasks (algorithms for finding communities, centralities, epidemics, etc.) that cannot be enumerated here. Among all these problems and their associated MapReduce solutions, we had to select some of them based on (a) their usefulness in complex network analysis tasks, (b) in their suitability to the MapReduce programming paradigm, (c) the availability of their implementations (free/open code) for purposes of reproducibility of measurements, and (d) complexity in terms of multiple rounds of map-reduce operations. Based on these criteria, we selected three problems/algorithms for running our experimentations[1]. The first algorithm deals with a very simple problem which is at the same time a fundamental operation in Facebook, that of *finding mutual friends*. The second algorithm deals with a network-wide path-based analysis for *finding connected components* which finds applications in reachability queries, techniques for testing network robustness and resilience to attacks, epidemics, etc. The third algorithm is about *counting triangles* which is a fundamental operation for higher level tasks such as calculating the clustering coefficient, or executing community finding algorithms based on clique percolation concepts. Table 1 summarizes the "identity" of the tasks.

Table 1. Characterization of problems/algorithms examined.

Primitive task	Type of analysis	Type of analysis
Mutual friends	Neighbor-based	Local network (neighborhood) properties Recommendation queries
Connected Components	Path-based	Large-scale network properties Reachability queries Resilience queries
Triangle counting	Mixed (extended neighborhood & paths)	Large-scale network properties Clustering/communities finding queries

We deferred a more advanced method for measuring the performance for multi-job workload such as the one described in [1], because the standalone, one-job-at-the-time method allows for the examination of interaction between MapReduce and storage media without the interventions of job scheduling and task placement algorithms. We aim at showing that the conclusions about the relative performance of SSDs versus HDDs are strongly depended on the features of the algorithms examined, which has largely been neglected in earlier relative studies [4, 5, 8], and based on these features we draw some conclusions on the relative benefits of SSDs.

4 System Setup

A commodity computer (Table 2) was used for the experiments. Three storage media were used (Table 2) with capacities similar to that used in [8]. On each of the three drives (one HDD and two SSDs) a separate and identical installation of the latest

[1] The MapReduce codes (along with many experiments) can be found in the technical report at http://www.inf.uth.gr/~dkatsar/Hadoop-SSD-HD-for-SNA.pdf.

version of required software was used. We emphasize at this point that since we need to factor out the network effects, we used single machine installations. Three different incremental setting setups were used: (a) with default settings, allowing 6 parallel maps, (b) with modified containers, allowing 3 parallel maps, and (c) with custom settings (Table 3). In all these setups, speculative execution was disabled and no early shuffling was permitted.

Table 2. Computer specifications.

CPU	Intel i5 4670 3.4 GHz (non HT)
RAM	8 GB 1600 MHz DDR3 (1333 MHz with disabled XMP)
Disk 1 (HDD)	Western Digital Blue WD10EZEX 1TB
Disk 2 (SSD1)	Samsung 840 EVO 120 GB
Disk 3 (SSD2)	Crucial MX100 512 GB

Table 3. Custom settings.

mapreduce.reduce.shuffle.parallel.copies	5 -> 50
mapreduce.task.io.sort.factor	10 -> 100
mapreduce.map.sort.spill.percent	0.80 -> 0.90
io.file.buffer.size	4 kb -> 64 kb

5 Input Data and Performance Measures

For the evaluation of the two disk types, we used *ten real social network data* (Table 4). They were retrieved from https://snap.stanford.edu/ and http://konect.uni-koblenz.de/.

The two SSDs were of different size disallowing the execution of some datasets. The most important measures we captured were the Map and Reduce execution times,

Table 4. Social networks used for evaluation.

#	Social network name	#Nodes	#Edges
1	Brightkite location based online social network	58,228	214,078
2	Gowalla location based online social network	196,591	950,327
3	Amazon product co-purchasing network	334,863	925,872
4	DBLP collaboration network	317,080	1,049,866
5	YouTube online social network	1,134,890	2,987,624
6	YouTube (ver. 2) online social network	3,223,589	9,375,374
7	Flickr	1,715,255	15,550,782
8	LiveJournal online social network	3,997,962	34,681,189
9	LiveJournal (ver. 2) online social network	5,204,176	49,174,620
10	Orkut online social network	3,072,441	117,185,083

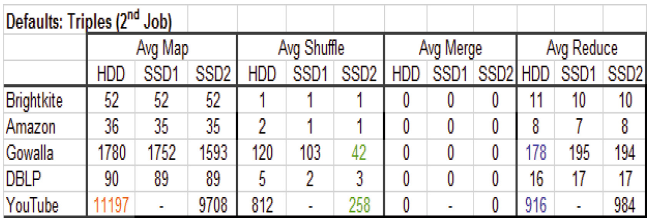
as also Sort (merge) and Shuffle phase. One common side effect is “cache hits” from previous executions, that was also experienced in [8]. In order to give each experiment an equal environment, Hadoop was halted and page cache was flushed, after each experiment. Before each test, HDFS was re-formatted.

6 The Results

1. Mutual Friends

The complexity of this algorithm is exponential due to the mapper of the 2nd MapReduce job. Thus, the 2nd MapReduce job is the most resource-intensive of the three jobs, rendering it a good inspection point for our measures (see Table 5), whereas the 1st and 3rd MapReduce jobs were fast-executed and almost identical for all disks. For Amazon, Brightkite and DBLP, the three disks performed almost equally. For the bigger datasets, the magnetic disk gives competitive (with respect to both SSD drives) execution times for the reduce phase, but the HDD performs worse for the map phase. The SSD2 displays superior performance at shuffling.

Table 5. Average times for each phase for 2nd job (creating triples) of “mutual friends” algorithm



Defaults: Triples (2nd Job)												
	Avg Map			Avg Shuffle			Avg Merge			Avg Reduce		
	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2
Brightkite	52	52	52	1	1	1	0	0	0	11	10	10
Amazon	36	35	35	2	1	1	0	0	0	8	7	8
Gowalla	1780	1752	1593	120	103	42	0	0	0	178	195	194
DBLP	90	89	89	5	2	3	0	0	0	16	17	17
YouTube	11197	-	9708	812	-	258	0	-	0	916	-	984

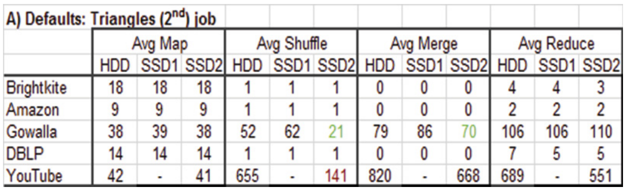
2. Counting Triangles

Here, the SSDs outperform the HDD for all evaluated datasets. At “forming the triads” job, HDD illustrated competitive behavior at reduce phase (Table 7). The “counting the triangles” job demonstrated greater variations in execution times. Our evaluation shows that with small datasets the performance differentiations between the two disk types are small (Table 6), whereas with larger ones (like YouTube dataset), SSDs capabilities become evident for shuffle and merge (sort) phases.

For the 1st MR job (creating triads), map, shuffle and merge phases finished quite fast and with almost zero differentiations among disks. Reduce phase lasted significantly longer with both disks performing equally (Table 7). With containers settings, the biggest dataset of Flickr gets significant improvement for both disk types (Table 8).

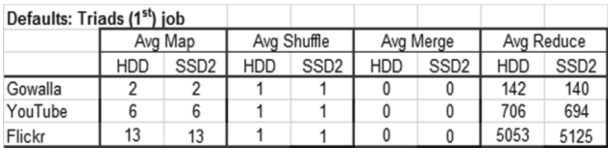
To optimize performance, increasing the following settings provided best results for the magnetic disk, compared to “containers” settings:

Table 6. Average times for each phase for 2nd job (calculate triangles) of "counting triangles"



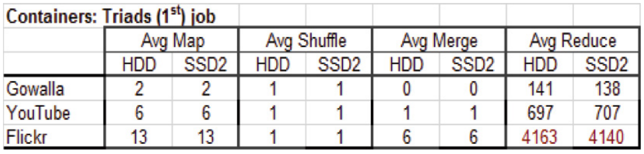
A) Defaults: Triangles (2nd) job												
	Avg Map			Avg Shuffle			Avg Merge			Avg Reduce		
	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2
Brightkite	18	18	18	1	1	1	0	0	0	4	4	3
Amazon	9	9	9	1	1	1	0	0	0	2	2	2
Gowalla	38	39	38	52	62	21	79	86	70	106	106	110
DBLP	14	14	14	1	1	1	0	0	0	7	5	5
YouTube	42	-	41	655	-	141	820	-	668	689	-	551

Table 7. Average times for each phase for 1st job (create triads) of "counting triangles" algorithm



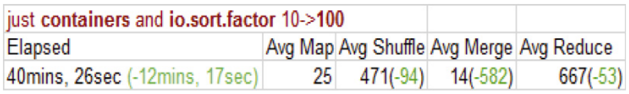
Defaults: Triads (1st) job								
	Avg Map		Avg Shuffle		Avg Merge		Avg Reduce	
	HDD	SSD2	HDD	SSD2	HDD	SSD2	HDD	SSD2
Gowalla	2	2	1	1	0	0	142	140
YouTube	6	6	1	1	0	0	706	694
Flickr	13	13	1	1	0	0	5053	5125

Table 8. Average times for each phase for 1st job (create triads) of "counting triangles" algorithm, with changed container's settings



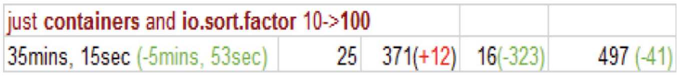
Containers: Triads (1st) job								
	Avg Map		Avg Shuffle		Avg Merge		Avg Reduce	
	HDD	SSD2	HDD	SSD2	HDD	SSD2	HDD	SSD2
Gowalla	2	2	1	1	0	0	141	138
YouTube	6	6	1	1	1	1	697	707
Flickr	13	13	1	1	6	6	4163	4140

Table 9. Performance difference for YouTube dataset at "Counting Triangles", increasing sort factor, for HDD



just containers and io.sort.factor 10->100				
Elapsed	Avg Map	Avg Shuffle	Avg Merge	Avg Reduce
40mins, 26sec (-12mins, 17sec)	25	471(-94)	14(-582)	667(-53)

Table 10. Performance difference for YouTube dataset at "Counting Triangles", increasing sort factor, for SSD2

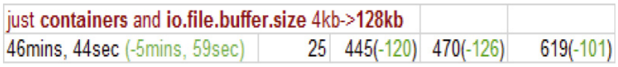


just containers and io.sort.factor 10->100				
35mins, 15sec (-5mins, 53sec)	25	371(+12)	16(-323)	497 (-41)

(a) *The number of streams to merge at once while sorting files. Minimizes merge time for both disk types. Improves HDD shuffling time as well* (Tables 9 and 10).
(b) *The buffer size for I/O (read/write) operations* (Table 11).

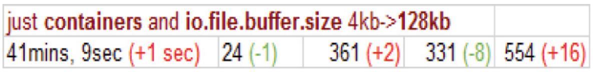
On the other hand, increasing the buffer size for I/O operations had minimal effect on SSD2 performance (Tables 12, 13 and 14).

Table 11. Performance difference for YouTube dataset at "Counting Triangles", increasing file buffer size, for HDD



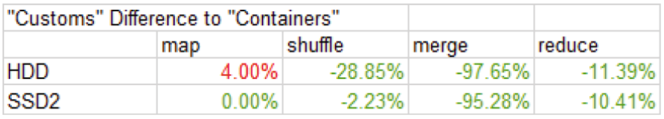
just containers and io.file.buffer.size 4kb->128kb				
46mins, 44sec (-5mins, 59sec)	25	445(-120)	470(-126)	619(-101)

Table 12. Performance difference for YouTube dataset at "Counting Triangles", increasing file buffer size, for SSD2



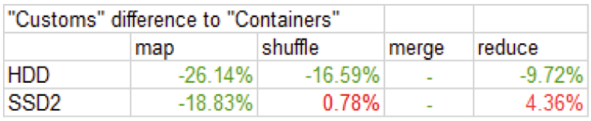
just containers and io.file.buffer.size 4kb->128kb				
41mins, 9sec (+1 sec)	24 (-1)	361 (+2)	331 (-8)	554 (+16)

Table 13. Percentage difference between "customs" and "containers settings for YouTube dataset, at "Counting Triangles" algorithm



"Customs" Difference to "Containers"				
	map	shuffle	merge	reduce
HDD	4.00%	-28.85%	-97.65%	-11.39%
SSD2	0.00%	-2.23%	-95.28%	-10.41%

Table 14. Percentage difference between "customs" and "containers settings for YouTube dataset, at "Mutual Friends" algorithm

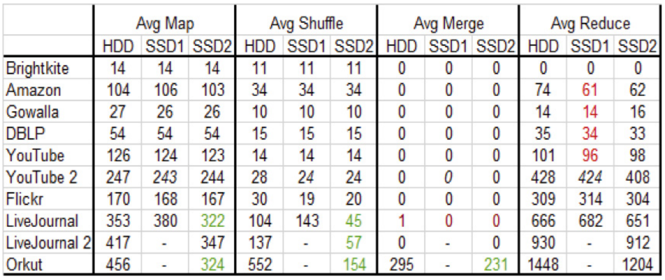


"Customs" difference to "Containers"				
	map	shuffle	merge	reduce
HDD	-26.14%	-16.59%	-	-9.72%
SSD2	-18.83%	0.78%	-	4.36%

3. Connected Components

Comparing SSD1 to the HDD, the Connected Components algorithm seems to slightly favor the SSD1 for small datasets, at reduce phase. Map, shuffle and phase times are close for both disk types (Table 15). For the datasets of Flickr and LiveJournal the magnetic disk takes the lead at reduce phase which is mostly characterized as "write" procedure for the Hadoop framework. Surprisingly, SSD1 performs quite slowly at shuffle phase for the LiveJournal dataset. The SSD2 generally delivers great performance especially at map and shuffle phase, noticeably as the datasets' size increase. For the reduce phase HDD falls behind SSD2, but not with a great margin.

Table 15. Sum of average times for each phase for the iterative Jobs of "Connected Components"



	Avg Map			Avg Shuffle			Avg Merge			Avg Reduce		
	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2
Brightkite	14	14	14	11	11	11	0	0	0	0	0	0
Amazon	104	106	103	34	34	34	0	0	0	74	61	62
Gowalla	27	26	26	10	10	10	0	0	0	14	14	16
DBLP	54	54	54	15	15	15	0	0	0	35	34	33
YouTube	126	124	123	14	14	14	0	0	0	101	96	98
YouTube 2	247	*243*	244	28	*24*	24	0	*0*	0	428	*424*	408
Flickr	170	168	167	30	19	20	0	0	0	309	314	304
LiveJournal	353	380	322	104	143	45	1	0	0	666	682	651
LiveJournal 2	417	-	347	137	-	57	0	-	0	930	-	912
Orkut	456	-	324	552	-	154	295	-	231	1448	-	1204

7 Conclusions

We compared the performance of solid state drives and hard disk drives for social network analysis. SSDs didn't come out as the undisputed winner. The second SSD performed significantly better. In many cases SSD1 and the magnetic disk came into a draw. Although SSD1 was slightly faster in many tests, in some cases the magnetic disk outperformed the SSD1. Even comparing to the faster SSD2, the magnetic disk provided competitive times for reduce phase, especially with the "mutual friends" algorithm, where it performed marginally better. Magnetic disk's shuffle times can be reduced. SSD's performance doesn't present further improvement. Nevertheless, HDD can't catch up with SSD's superior performance at shuffling. With tweaking merge-sort can be performed in less steps minimizing merge's phase times for both disk types, slightly favoring magnetic disk that would perform slower otherwise. For map phase both disk types can get similar performance improvement.

Acknowledgement. This work was supported by the Project "REDUCTION: Reducing Environmental Footprint based on Multi-Modal Fleet management System for Eco-Routing and Driver Behaviour Adaptation," funded by the EU.ICT program, Challenge ICT-2011.7.

References

1. Chen, Y., Ganapathi, A., Griffith, R., Katz, R.: The case for evaluating MapReduce performance using workload suites. In: Proceedings of IEEE MASCOTS, pp. 390–399 (2011)
2. Huang, S., Huang, J., Dai, J., Xie, T., Huang, B.: The HiBench benchmark suite: Characterization of the MapReduce-based data analysis. In: Proceedings of ICDE Workshops (2010)

3. Islam, N., Rahman, M., Jose, J., Rajachandrasekar, R., Wang, H., Subramoni, H., Murthy, C., Panda, D.: High performance RDMA-design of HDFS over InfiniBand. In: Proceedings of SC (2012)
4. Kambatla, K., Chen, Y.: The truth about MapReduce performance on SSDs. In: Proceedings of LISA, pp. 109–117 (2014)
5. Kang, S.-H., Koo, D.-H., Kang, W.-H., Lee, S.-W.: A case for flash memory SSD in Hadoop applications. Int. J. Control Autom. **6**, 201–210 (2013)
6. Krish, K.R., Iqbal, M.S., Butt, A.R.: VENU: orchestrating SSDs in Hadoop storage. In: Proceedings of IEEE BigData, pp. 207–212 (2014)
7. Min, C., Kim, K., Cho, H., Lee, S.-W., Eom, Y.I.: SFS: random write considered harmful in solid state drives. In: Proceedings of USENIX FAST (2012)
8. Moon, S., Lee, J., Kee, Y.S.: Introducing SSDs to the Hadoop MapReduce framework. In: Proceeding of IEEE CLOUD, pp. 272–279 (2014)
9. Saxena, P., Chou, J.: How much solid state drive can improve the performance of Hadoop cluster? Performance evaluation of Hadoop on SSD and HDD. Int. J. Mod. Commun. Technol. Res. **2**(5), 1–7 (2014)
10. Sur, S., Wang, H., Huang, J., Ouyang, X., Panda, D.: Can high-performance interconnects benefit Hadoop distributed file system. In: Proceedings of the Workshop MASVDC (2010)
11. Wu, D., Xie, W., Ji, X., Luo, W., He, J., Wu, D.: Understanding the impacts of solid-state storage on the Hadoop performance. In: Proceedings of Advanced Cloud and Big Data, pp. 125–130 (2013)

Designing HMMs in the Age of Big Data

Cesare Alippi[1,2], Stavros Ntalampiras[1(✉)], and Manuel Roveri[1]

[1] Politecnico di Milano, Piazza L. da Vinci 32, 20133 Milano, Italy
{cesare.alippi,stavros.ntalampiras,manuel.roveri}@polimi.it
[2] Università della Svizzera Italiana, Lugano, Switzerland

Abstract. The rise of the Big Data age made traditional solutions for data processing and analysis unsuitable due to the high computational complexity. To address this problem, novel solutions specifically-designed techniques to analyse Big Data have been recently presented. In this path, when such a large amount of data arrives in a streaming manner, a sequential mechanism for the Big Data analysis is required. In this paper we target the modelling of high-dimension datastreams through hidden Markov models (HMMs) and introduce a HMM-based solution, named *h-HMM*, suitable for datastreams characterized by high dimensions. The proposed *h-HMM* relies on a suitably-defined clustering algorithm (operating in the space of the datastream dimensions) to create clusters of highly uncorrelated dimensions of the datastreams (as requested by the theory of HMMs) and a two-layer hierarchy of HMMs modelling the datastreams of such clusters. Experimental results on both synthetic and real-world data confirm the advantages of the proposed solution.

1 Introduction

Not rarely when a great amount of data is made available e.g. by distributed and pervasive systems [2,7], traditional solutions for data processing and analysis revealed to be inadequate due to the excessive computational complexity. For this reason, a novel generation of solutions recently arose under the umbrella of *Big Data Analytics* [10], whose goal is to analyse Big Data to extract relevant/valuable information. Big Data often come in a streaming manner and a sequential analysis is the unique possibility to provide real time answers. The problem amplifies when the computational complexity per datum is high, as frequently is the case in neural networks processing say, when convolutional networks, deep learning or multidimensional HMM models need to be both generated and executed.

Among the solutions present in the literature for this purpose (e.g., hierarchical self-organizing maps [12], principal component analysis and compressive sampling [17]), we focus on hidden Markov models (HMMs) that represent a well-known and widely-used statistical tool for modelling the temporal evolution of

This work was supported by the Politecnico di Milano International Fellowship Program.

streams of data [14]. HMMs have been successfully applied to many application scenarios for event detection (e.g., speech [11] and emotion recognition [9], human behaviour analysis [5], cognitive fault detection and identification [4]) but, to the best of our knowledge, HMMs have not been considered in the Big Data scenario yet. In fact, in the HMM literature, the considered datastreams rarely exceed the 50 dimensions due to the bad scaling of the computational complexity w.r.t. the problem dimension. HMM-based solutions operating on high-dimension datastreams have been proposed for specific application scenarios (e.g., network anomaly detection [6] and identity management for social media networks [18]). Hence, modelling Big Data with HMMs still represents an open problem. The reason is twofold: (i) theoretically, the accuracy of the learning phase generally decreases with the dimension of the considered datastream since a large training set is required to obtain the modelling accuracy that is feasible in low dimensions [3]; (ii) modelling high-dimension datastreams through a HMM can be a time and resources consuming task.

The aim of this paper is to propose a novel HMM-based solution for the modelling of high-dimension datastreams. Similarly to other solutions present in the field of Big Data and Data Analytics [16], the proposed solution relies on the ability to transform the modelling problem into sub-problems, which can be more easily managed, and suitably aggregate their outcomes. In more detail, the proposed solution, named *hierarchical-HMM (h-HHM)*, is based on the use of a suitably-defined clustering algorithm to create a set of highly uncorrelated features, i.e., the dimensions of the datastreams, and a two-layer hierarchy of HMMs meant to learn, in a hierarchical way, the system model generating the high-dimension datastreams.

Through a suitably defined figure of merit, the proposed solution takes into account the learning accuracy as well as the computational time of the proposed *h-HMM* during the operational life, since it plays a critical role in Big Data Analytics.

The paper is organized as follows. Section 2 describes the architecture of the proposed *h-HMM*, while its learning algorithm is presented in Sect. 3. Experimental results are detailed in Sect. 4. Finally, Sect. 5 presents our conclusions and future work.

2 The Proposed Hierarchical HMM-based Solution: The Architecture

Let $X = \{x(t), t = 1, 2, \ldots\}$ with $x(t) \in \mathbb{R}^d$ be a stream of data generated by an a-priori unknown data-generating process, where the dimension d could be very large. The goal of the proposed *h-HHM* is to learn the system model generating X.

A general overview of the proposed solution is depicted in Fig. 1: the *h-HHM* relies on a clustering algorithm meant to create clusters of uncorrelated dimensions of the datastreams and a two-layer hierarchy of HMMs meant to model, in a hierarchical way, the datastreams of such clusters and, suitably aggregated, of

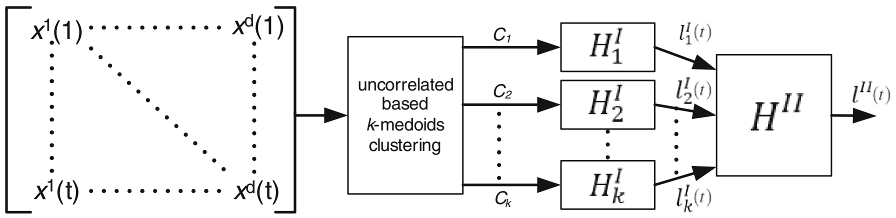


Fig. 1. The general idea of the proposed Hierarchical HMM-based solution for large-dimension datastreams.

the whole datastream. Through the clustering phase and the subsequent modelling of the datastreams coming from these clusters, the idea is to transform the problem of modelling X into a sub-problems that can be more easily managed by the proposed *h-HHM*.

In more detail, a set of clusters $C_j, j \in 1, \ldots, k$ meant to cluster the dimensions of X is provided by a suitably-defined clustering algorithm based on the analysis of uncorrelation among dimensions. The proposed clustering algorithm, which will be detailed in the next section, operates in the space of the dimensions of X by relying on the theoretical foundations of HMMs that assume to learn the stochastic behaviour of uncorrelated features [14].

Afterwards, the hierarchy of HMMs is organized into two layers, named H^I and H^{II}, respectively.

The first layer H^I comprises k HMMs, i.e.,

$$H^I = \{H_1^I \ldots H_k^I\}, \tag{1}$$

aiming at modelling the statistical behaviour of the k clusters identified by the clustering algorithm. These HMMs are initially trained on a training sequence (as described in the next section), while, during the operational life, they analyse acquired data $x(t)$'s to compute the associated log-likelihood values

$$l^I(t) = \{l_1^I(t), \ldots, l_k^I(t)\}. \tag{2}$$

These log-likelihood values measure the extent to which HMMs in H^I of Eq. (1) are able to recognize data from their respective clusters up to time t.

The second layer of the *h-HMM* relies on a single HMM, named H^{II}, whose goal is to learn the log-likelihood $l^I(t)$s of Eq. 2, which is a k-dimensional datastream and provide, as output, the loglikelihood $l^{II}(t)$. Hence, this second layer represents the aggregation of the outputs of the first-layer HMMs. Here, the idea is to capture the temporal behaviour of the likelihoods produced by the first-layer HMMs representing the extent to which each HMM is able to recognize the sequence of data acquired by the corresponding cluster. Hence, the second-layer HMM is meant to jointly learn the temporal behaviour of the recognition abilities of the first-layer HMMs: $l^{II}(t)$ measures the extent to which *h-HMM* is able to recognize the statistical behaviour of X over time.

Being HHMs probabilistic graphs, H^I and H^{II} are formalized through the following components:

- the number of states, S,
- the probability density function associated with each state modelled as a mixture of Gaussians (GMM), $P(y|x) = \sum_{k=1}^{K} p_k p(y|x_{(k)})$, where $p_k s$ are the mixture weights, y is a continuous-valued data vector (e.g. measurements or features), $x_{(k)}$ represents the $k - th$ component of the vector, $x = [\sigma, \mu]$, $p(y|x_{(k)}) = \frac{1}{(2\pi)^{d/2}|\sigma|} e^{-\frac{1}{2}(y-\mu_k)^t \sigma_k^{-1}(y-\mu_k)}$. In general HMMs assume the availability of independent features for the estimation of GMMs.
- the state transition probability matrix $A = \{a_{ij}\}$ where entry a_{ij} represents the probability of moving from state j at time t to state i at time *t+1*. For example, the transition probability of moving from state 1 to state 2 is represented by a_{12}. In case some transitions are not allowed, the respective a_{ij}s should be set to zero.
- the initial state distribution $\pi = \{\bar{\pi}_i\}$, where $\bar{\pi}_i$ corresponds to the probability that the HMM starts in state i, i.e. $\pi_i = P[q_1 = S_1], 1 \leq i \leq N$.

Hence, the first level HMMs are characterized by the components $H_j^I = \{S_j^I, P_j^I, A_j^I, \pi_j^I\}$, $j = 1 \dots k$, while the one at the second layer by $H^{II} = \{S^{II}, P^{II}, A^{II}, \pi^{II}\}$. The joint training of $\{H_1^I \dots H_k^I\}$ and H^{II} is achieved through a novel training algorithm that will be discussed in the next section. The proposed training algorithm allows also to automatically discover the correct number of cluster k, which is a crucial parameter for *h-HMM*. In fact, this choice heavily affects the learning ability of the *h-HHM* since large values of k would improve the learning of $\{H_1^I \dots H_k^I\}$ (working on less dimensions) at the expenses of an increased complexity of the learning of H^{II} (which must operate on larger number of dimensions). Differently, smaller numbers of clusters would ease the learning of H^{II} at the expenses of the learning abilities of $\{H_1^I \dots H_k^I\}$ that must operate on clusters characterized by larger dimensions. Hence, the correct trade-off between the accuracy of the two layers of the *h-HHM* must be identified to maximize the overall accuracy.

3 The Proposed Hierarchical HMM-based Solution: The Learning

The aim of this section is to describe the proposed learning algorithm for the *h-HMM*, whose architecture has been presented in the previous section.

The proposed algorithm, which is detailed in Algorithm 1, aspires at identifying the configuration k, $H_j^I = \{S_j^I, P_j^I, A_j^I, \pi_j^I\}$, $j = 1 \dots k$, and $H^{II} = \{S^{II}, P^{II}, A^{II}, \pi^{II}\}$ that maximise a suitably-defined figure of merit

$$F = \overline{l^{II}} - \gamma E_T \tag{3}$$

```
Input: $TS, \gamma$;
1. Create $T_{TS}, V_{TS}^1, V_{TS}^2$ as described in Sect. 3;
2. $F = []$;
3. Set $k_{max} = d$;
4. for $k$=1:$k_{max}$ do
    5. Apply $k$-medoids algorithm based on uncorrelation given $k$ and get
    $C^k = \{C^1, \dots, C^k\}$;
    6. Train $H^I = \{H_1^I, \dots, H_k^I\}$ on $TS$;
    for $j$=1:$k$ do
        7. Evaluate HMM $H_j$ on $V_{TS}^1$ and compute the log-likelihoods $l_j(t)$ on
        $V_{TS}^1$;
        8. $l_j^I = \{l_j^I(1), \dots, l_j^I(t)\}$ on $V_{TS}^1$;
    end
    9. $l^I = \{l_1^I, \dots, l_k^I\}$;
    10. Train $H^{II}$ on $l^I$ on $V_{TS}^1$ and compute $l^{II}(t)$ on $V_{TS}^2$;
    11. Measure execution time $E_T$ as described in Sect. 3;
    12. $F = [F; \overline{l^{II}} - \gamma \times E_T]$, where $\overline{l^{II}}$ is the average value of $l^{II}(t)$ on $V_{TS}^2$;
end
13. Determine $max(F)$ and the associated $k$;
```

Algorithm 1. The algorithm for modelling d features using k-medoids clustering and hierarchical HMMs.

taking into account the average log-likelihood $\overline{l^{II}}$ on validation set V_{TS}^2 and the execution time E_T of the *h-HMM* (normalized w.r.t. $\overline{l^{II}}$), multiplied by an application-specific coefficient factor $\gamma \geq 0$. E_T denotes the time required for the processing of one sample $x(t)$ to compute the associated log-likelihood $l^{II}(t)$, hence taking into account the whole processing of $x(t)$ through H_j^Is and H^{II}. F measures the modelling accuracy as well as the associated computational time as a penalty term.

The algorithm assumes the availability of a training sequence $TS = \{x(t), t = 1, \dots, t_T\}$, aiming at modelling the nominal state of the X. TS is then divided into $T_{TS}, V_{TS}^1, V_{TS}^2$ (line 1) as follows: $TS = [T_{TS}, V_{TS}^1, V_{TS}^2]$, whose lengths are l_{TS}, l_{V1}, and l_{V2} respectively, and the maximum number of clusters k_{max} to be considered is set equal to the dimension of the feature vector d (lines 3). The proposed algorithm explores the number of clusters in the range $[1, k_{max}]$, where $k_{max} = d$ (line 4) and for each k, the ad-hoc clustering algorithm based on k-medoids algorithm is applied (line 5) and clusters $C^k = \{C^1, \dots, Ck\}$ are produced.

This clustering algorithm is meant to create uncorrelated clusters of features, i.e. dimensions of the datastream, maximizing the extent to which the features belonging to the same cluster are uncorrelated to each other. As shown in Sect. 4, this allows to improve the HMM learning of each cluster, hence increasing the learning ability of the whole *h-HHM* solution.

The ad-hoc clustering algorithm working in the space of datastream dimensions is organized as follows: the core is the k-mediods algorithm that is meant to break up the datastreams into groups and tries to minimize the distance between the elements (which in our case are the dimensions of the datastreams) of a group and the element which is chosen to be the cluster center. Its implementation is based on the Partitioning Around Medoids algorithm explained in [19]. The proposed distance metric is essentially the cross-correlation, i.e. the distance $D_{x^i-x^j}$ between two dimensions x^i and x^j, i.e. the i-th and j-th dimension of X, that is captured by the following formula: $D_{x^i-x^j} = \frac{(x^i-\bar{x^i})(x^j-\bar{x^j})'}{\sqrt{(x^i-\bar{x^i})(x^i-\bar{x^i})'}\sqrt{(x^j-\bar{x^j})(x^j-\bar{x^j})'}}$ (eventually after alignment of x^i and x^j at $\tau_{delay} = \underset{TS}{\operatorname{argmax}}(D_{x^i-x^j})$), while $\bar{x^i}$ and $\bar{x^j}$ are the mean values of the respective dimensions over TS. This ad-hoc clustering algorithm minimizes D and produces clusters with uncorrelated dimensions.

This choice is motivated by the Gaussian mixtures $P(y|x)$ forming the states of the HMM described in the previous section, where diagonal covariance matrices are assumed, i.e. feature independence. Thus, when the HMM training algorithm is fed with uncorrelated features, the behaviour of the datastreams is captured more accurately with respect to utilizing correlated ones. This point will be experimentally evaluated in Sect. 4.

Once $C^k = \{C^1, \ldots, Ck\}$ has been computed the proposed algorithm learns $H^I = \{H^I_1, \ldots, H^I_k\}$ on TS (line 6). This learning phase is based on the Baum-Welch algorithm [8], while fully-connected HMMs are considered (i.e. the system may transit to any state at a given time instant, thus $A^I_j > 0, j = 1 \ldots k$ and $A^{II} > 0$). The matrix l^I is formed by collecting the vectors $\{l^I_1, \ldots, l^I_k\}$, which are computed by evaluating $H^I = \{H^I_1, \ldots, H^I_k\}$ on V^1_{TS} (line 7), i.e. $l^I = \{l^I_1, \ldots, l^I_k\}$ with k dimensions and length l_{V1} (line 8).

Subsequently, we capture the behaviour of l^I though H^{II}, which operates on the sequence of log-likelihoods generated by H^I on V^1_{TS}. H^{II} is evaluated on V^2_{TS} for obtaining l^{II}, which characterizes the extent to which $h - HMM$ is able to recognize the statistical behaviour of X (line 10). Finally, we compute the figure of merit $F = \overline{l^{II}} - \gamma \times E_T$ for the specific value of k', where E_T is computed on one sample $\overline{l^{II}}$ is the average value of $l^{II}(t)$ on V^2_{TS} (line 11). Its maximum value (line 12) over the range of $k = [1, \ldots, k_{max}]$ reveals the optimal *h-HMM* setting (the block diagram of the proposed system is depicted in Fig. 1).

4 Experiments

This section explains the experimental campaign designed to assess the effectiveness of the proposed *h-HMM* in both synthetic and real datasets. Here, the goal is to show that the proposed *h-HMM* is able to provide better modelling abilities (measured as larger log-likelihood values) than other solutions present in the literature on high-dimension datastreams.

Synthetic Datasets. We modelled X as a 200 dimensional data-generating process, whose data $x(t)$ are randomly drawn from a multivariate normal distribution with mean vector μ, and covariance matrix $\sum$ containing integers randomly drawn from the discrete uniform distribution on the interval $[0, 100]$.

In particular, we considered two different scenarios for X:

- independent dimensions F^{id}: the dataset is built by considering a diagonal covariance matrix $\sum_{id}$ leading to independent feature vectors;
- dependent dimensions F^{d}: a symmetric positive semi-definite covariance matrix $\sum_{d}$ is considered leading to correlated dimensions.

The length of each experiment is 10,000 samples, while results are averaged over 50 runs. The proposed *h-HMM* is compared with the following three solutions: (a) the traditional HMM; (b) c-HMM: $h - HMM$ with correlation based clustering, where the distance metric for the k-medoids is $D_{x^i - x^j} = 1 - Cor(x^i, x^j)$ aiming at clustering together the dimensions which are highly correlated; and (c) r-HMM: $h - HMM$ where the dimensions of X are randomly clustered using labels drawn from a uniform distribution on the interval $[1, k_{max}]$.

It should be noted that, during our experiments, we used the k-medoids algorithm, which is embedded in the MATLAB© programming platform and the HMM implementation provided in [13]. The length of TS is 6000 samples and l_{TS}, l_{V1}, and l_{V2} are 4800, 600, and 600 samples, respectively. HMMs are trained by exploring the following configurations: S ∈[3, 4, 5, 6, 7, 8, 9], and Gaussian functions ∈[2, 4, 8, 16, 32, 64, 128, 256, 512]. The HMM providing the highest log-likelihood on a validation set was the final choice. For the HMMs in H^{I} the validation set is V^{1}_{TS}, while for H^{II}, V^{2}_{TS}.

The experimental results are shown in Table 1 including the three considered solutions (h-HMM, c-HMM, r-HMM) as well as the traditional single HMM. All approaches are evaluated on both F^{id} and F^{d}. In addition, Fig. 2 depicts the considered figure of merit F as a function of k for the proposed *h-HMM* in both scenarios.

By observing Table 1 we can see that the *h-HMM* approach provides higher accuracy than all the other approaches (i.e., traditional HMM, c-HMM and r-HMM) at the expense of a slight increase of the E_T. Nevertheless, we emphasize that E_T could be managed by tuning the parameter γ in F.

Hence, the *h-HMM* approach considering uncorrelated features provides the best performance when applied to both F^{d} and F^{id}. This behaviour is justified by the fact that the GMMs assume diagonal covariance, i.e. feature independence. Interestingly, a small amount of clusters leads to quite satisfactory performance when modelled in a hierarchical fashion without needing an excessive amount of additional time. This is due to the fact that hierarchical HMMs take into consideration the temporal evolution of the log-likelihoods produced by the H^{I}'s over time.

Another interesting observation is that results on F^{id} are better than F^{d}, which is reasonable since the feature independence is assumed by HMMs as mentioned in Sect. 3. *h-HMM* is able to consistently provide higher log-likelihoods than c-HMM and r-HMM across every different clustering setting. At the same time, E_T is kept at relatively low values. In addition, the proposed *h-HMM*

Table 1. The modelling performances in terms of log-likelihoods with respect to two synthetic datasets and the four solutions. The standard deviations are given in parenthesis.

	k	c-HMM	E_T(s)	r-HMM	E_T(s)	h-HMM	E_T(s)	HMM	E_T(s)
F^d	1	−60.2 (1.4)	1.46	−60.2 (1.4)	1.46	−60.2 (1.4)	1.46	−2413.2(98.7)	1.4
	2	−99.5 (18.7)	4.4	−93.1 (11.6)	4.5	−81.6 (11.9)	3.92		
	3	−130.9 (9.6)	4.53	−136.3 (13.9)	4.55	−99.9 (13.4)	4.29		
	5	−171.1 (14.4)	4.59	−171.9 (14.8)	4.66	−122.3 (14.2)	4.44		
	10	−380.9 (22.4)	5.91	−380.5 (20.7)	5.9	−276.7 (27.9)	5.36		
	20	−695.5 (33.7)	8.81	−689.6 (30.9)	8.76	−533.4 (36.2)	7.33		
	50	−1263.7 (51.9)	16.01	−1274.8 (45.5)	15.4	−1045.0 (52.9)	12.73		
	100	−3113.8 (86.4)	37.19	−2406.9 (79.1)	38.9	−2705.9 (107)	33.66		
	200	−3981.2 (108.3)	61.62	−2981.2 (109.2)	61.6	−3602.1 (176.3)	60.8		
F^{id}	1	−101.2 (13.7)	1.42	−101.2 (13.7)	1.42	−101.2 (13.7)	1.42	−2386.4(103.2)	1.38
	2	−92.9 (9.9)	3.9	−90.1 (8.2)	3.7	−54.2 (6.9)	4.72		
	3	−134.7 (14)	4.52	−131.6 (9.4)	4.51	−90.1 (8.5)	4.85		
	5	−166.7 (13.4)	4.6	−171.3 (13.9)	4.57	−132.1 (12.9)	4.87		
	10	−377.2 (20.2)	7.13	−384.3 (24.3)	7.38	−321.9 (18.5)	7.34		
	20	−683.5 (27.7)	10.24	−683.2 (31.1)	10.12	−610.4 (28.4)	9.95		
	50	−1263.6 (49.7)	16.67	−1243.1 (46.8)	16.13	−1693.1 (41.1)	19.24		
	100	−2897.9 (124.1)	36.93	−2305.5 (101.7)	38.57	−3441.1 (87.3)	31.47		
	200	−3469 (193.4)	60.75	−3190.2 (154.2)	60.75	−3841.1 (104.2)	61.52		

approach provides higher modelling accuracy that the standard HMM, which as expected provides very poor learning abilities on datastreams characterized by high dimensions.

As expected, in case of $k = 1$ *h-HMM*, c-HMM and r-HMM provide the same loglikelihood values (as well as E_T) since they all operate on the same unique cluster comprising the whole set of dimensions of X.

A Real-World Data. In this experimental phase we considered a time-series multivariate dataset containing 13,910 measurements from 16 chemical sensors exposed to 6 gases at different concentration levels, which is publicly available for research purposes at the UCI Machine Learning Repository [1]. Full description of the dataset is provided in [15,20]. The proposed algorithm was applied with $\gamma = 1$ and TS length equal to 10.000, and it automatically divided 129 dimensions into 7 clusters, while H^{II} provided $l^{II} = -456.4$. At the same time the single HMM was unable to capture the behaviour of the dataset since it provided a log-likelihood of $-\infty$. Here TS is not long enough for learning just one HMM. Thus, the $h - HMM$ might be more appropriate as it operates on features of low dimensions. In Fig. 3 we can see how the log-likelihood $\overline{l^{II}}$ alters with respect to different number of clusters. When considering less than 7 or more than 12 clusters, $\overline{l^{II}} = -\infty$ while its maximum values is reached when k=7. This behaviour shows that for several clustering settings TS is not adequate for learning the associated statistical behaviour, thus the HMM is unable to explain the observed feature sequence.

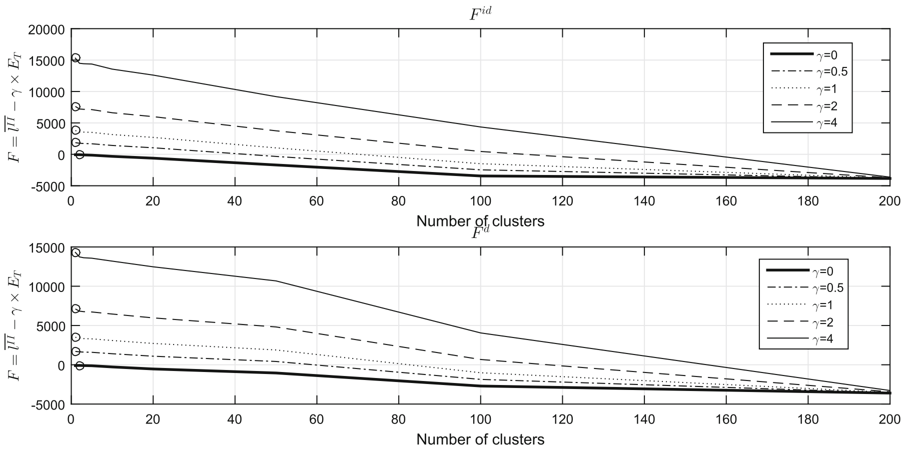


Fig. 2. The alteration of figure of merit F according to different clustering settings and γ values. The maximum points with respect to each γ value are highlighted with circles.

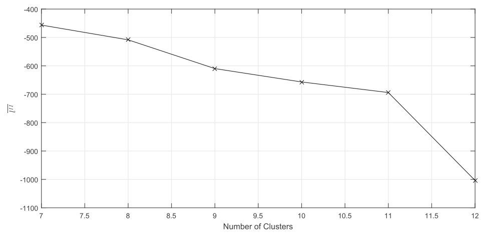


Fig. 3. The log-likelihood $\overline{l^{II}}$ with respect to different number of clusters computed on real-world data coming from 16 chemical sensors.

5 Conclusions

This paper presented an extensive study on statistical modelling of big data using HMMs. We proposed a hierarchical HMM-modeling approach employing a clustering algorithm for discovering groups including uncorrelated dimensions. The number of clusters is automatically selected by an algorithm maximizing a figure of merit, which considers both modelling accuracy and computational time. Thorough experiments on both synthetic and real-world data demonstrated the superiority of the proposed approach over the traditional HMM method and both the random and correlation-based clustering solutions.

This work comprises a first step towards HMM-based modelling of large scale datastreams, while our main goal is to extent the present modelling approach for learning data acquired by large scale cyber physical systems (LCPS) composed of heterogeneous units endowed with sensing, processing, communication and (possibly) actuation abilities. In such large scale scenario, the analysis of data to promptly detect changes in the environment and faults affecting sensors is of crucial importance to guarantee the quality-of-service of the LCPS-based application and activate proper reaction mechanisms.

References

1. http://archive.ics.uci.edu/ml/datasets/gas+sensor+array+drift+dataset+at+different+concentrations (2015)
2. Abolfazli, S., Sanaei, Z., Ahmed, E., Gani, A., Buyya, R.: Cloud-based augmentation for mobile devices: motivation, taxonomies, and open challenges. IEEE Commun. Surv. Tutorials **16**(1), 337–368 (2014). doi:10.1109/SURV.2013.070813.00285
3. Aladjem, Mayer: Projection pursuit fitting gaussian mixture models. In: Caelli, Terry, Amin, Adnan, Duin, Robert, P,W., Ridder, Dick, Kamel, Mohamed (eds.) SSPR/SPR 2002. LNCS, vol. 2396, pp. 396–404. Springer, Heidelberg (2002). doi:10.1007/3-540-70659-3_41
4. Alippi, C., Ntalampiras, S., Roveri, M.: A cognitive fault diagnosis system for distributed sensor networks. IEEE Trans. Neural Netw. Learn. Syst. **24**(8), 1213–1226 (2013)
5. Andersson, M., Ntalampiras, S., Ganchev, T., Rydell, J., Ahlberg, J., Fakotakis, N.: Fusion of acoustic and optical sensor data for automatic fight detection in urban environments. In: 2010 13th Conference on Information Fusion (FUSION), pp. 1–8 (2010)
6. Babaie, T., Chawla, S., Ardon, S., Yu, Y.: A unified approach to network anomaly detection. In: 2014 IEEE International Conference on Big Data (Big Data), pp. 650–655 (2014)
7. Bari, N., Mani, G., Berkovich, S.: Internet of things as a methodological concept. In: 2013 Fourth International Conference on Computing for Geospatial Research and Application (COM.Geo), pp. 48–55 (2013)
8. Baum, L.E., Petrie, T.: Statistical inference for probabilistic functions of finite state markov chains. Ann. Math. Stat. **37**(6), 1554–1563 (1966)
9. Guo, Y., Gao, H.: Emotion recognition system in images based on fuzzy neural network and hmm. In: 5th IEEE International Conference on Cognitive Informatics, ICCI 2006, vol. 1, pp. 73–78 (2006)
10. Hu, H., Wen, Y., Chua, T.S., Li, X.: Toward scalable systems for big data analytics: A technology tutorial. IEEE Access **2**, 652–687 (2014). doi:10.1109/ACCESS.2014.2332453
11. Kinjo, T., Funaki, K.: On hmm speech recognition based on complex speech analysis. In: 32nd Annual Conference on IEEE Industrial Electronics, IECON 2006, pp. 3477–3480 (2006)
12. Lee, C.H., Wu, C.H.: A novel big data modeling method for improving driving range estimation of EVS. IEEE Access **3**, 1980–1993 (2015). doi:10.1109/ACCESS.2015.2492923
13. Murphy, K.: https://www.cs.ubc.ca/murphyk/software/hmm/hmm.html (1998)
14. Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. In: Proceedings of the IEEE, pp. 257–286 (1989)
15. Rodriguez-Lujan, I., Fonollosa, J., Vergara, A., Homer, M., Huerta, R.: On the calibration of sensor arrays for pattern recognition using the minimal number of experiments. Chemometr. Intell. Lab. Syst. **130**, 123–134 (2014)

16. Shirkhorshidi, A.S., Aghabozorgi, S., Wah, T.Y., Herawan, T.: Big data clustering: a review. In: Murgante, B., Misra, S., Rocha, A.M.A.C., Torre, C., Rocha, J.G., Falcão, M.I., Taniar, D., Apduhan, B.O., Gervasi, O. (eds.) ICCSA 2014. LNCS, vol. 8583, pp. 707–720. Springer, Heidelberg (2014). doi:10.1007/978-3-319-09156-3_49
17. Slavakis, K., Giannakis, G.B., Mateos, G.: Modeling and optimization for big data analytics: (statistical) learning tools for our era of data deluge. IEEE Sig. Process. Mag. **31**(5), 18–31 (2014)
18. Subasinghe, K., Kodithuwakku, S.R.: A big data analytic identity management expert system for social media networks. In: 2015 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), pp. 126–129 (2015)
19. Theodoridis, S., Koutroumbas, K.: Pattern Recognition, 3rd edn. Academic Press, Inc., Orlando (2006)
20. Vergara, A., Vembu, S., Ayhan, T., Ryan, M.A., Homer, M.L., Huerta, R.: Chemical gas sensor drift compensation using classifier ensembles. Sensors Actuators B: Chem. **166–167**, 320–329 (2012)

Analyzing Big Security Logs in Cluster with Apache Spark

Talha Oktay(✉) and Ahmet Sayar

Kocaeli University, Umut Tepe Yerleskesi, 41380 Kocaeli, Turkey
{talha.oktay,ahmet.sayar}@kocaeli.edu.tr
http://www.kocaeli.edu.tr

Abstract. Cyber security is the major concern in today's highly networked environment and logging is the primary way of tracking compliance with the security policies. However analyzing the massive amount of logs has become a "Big Data" problem. Apache Spark is one of the latest and most notable incarnation of Data Flow Models in cluster computing. In terms of security log analysis, it provides an exceptional batch or interactive working environment. In this study, Apache Spark along with its distinctive features is briefly introduced, the challenges related to security logs analyzes are discussed and then some of Spark's security log analyzing capabilities are demonstrated through a problem related to big security logs. Finally, a sample Spark Application is presented that extracts statistics relevant to the problem.

Keywords: Big Data · Apache Spark · MapReduce · RDD · Resilient Distributed Dataset · Security log analyzing · Log analyzing in cluster

1 Introduction

A log can be defined as a record of an event [1]. A log is composed of log entries which are generally called as log messages. A log message or a log entry in broad terms is a message generated by computing, communications systems and applications in response to an event or events that happened to them. Logs are in general simple text files to which the new log messages are appended. Log messages are the primary source of information to identify operational problems, policy violations, security incidents. Logging is also used for audit and monitoring [2].

Logs, whether or not they are generated security in mind, generally contain valuable security related information. However, security log analytic has become a Big Data problem in terms of the log's massive size, various or undefined formats, various source, variable generation rate and veracity. In order to cope with these challenges, security log analytic should exploit Cloud Computing capabilities and Big Data analytic process.

In this study, Apache Spark is used as Cloud Big Data Analytic framework for security log analytic. It is shown that Apache Spark provides a very suitable platform for exploratory or batch security log analysis in the cloud environment. In the following two sections, properties of Apache Spark and security logs are briefly introduced, and then through a problem, Apache Spark security log analyzing capabilities are presented with a real-world dataset.

2 Apache Spark

Apache Spark[1] is an open source distributed or cluster computing platform and Big Data processing framework. It provides a fast general purpose computing environment that is easy to use and a sophisticated analytics capability that allows development of analytic applications in various popular languages such as Java, Scala, Python and R.

Apache Spark provides a very special distributed data abstraction called Resilient Distributed Dataset (RDD) which handles many complexities of distributed computing. RDD is a collection of objects or dataset distributed over cluster of nodes that can persist in memory. RDD is the core abstraction in Apache Spark which is described in dept at [3].

Apache Spark is a part of Hadoop[2] ecosystem and works seamlessly with Hadoop toolset. Apache Spark is distinguished with its speed from other Big Data processing frameworks including Hadoop because of its ability to run computations in memory. Apache Spark is also more efficient for the applications running on disk. It is stated that Apache Spark is faster than Apache Hadoop, 100 times for in memory processing, 10 times for on disk [4].

Apache Spark is suitable for batch as well interactive use. It can be used by data scientists or statisticians with limited or no knowledge of Cluster computing through its interactive shells similar to interactive shells of popular analytic programs such as MATLAB[3] or R[4]. It is especially well suited for interactive data mining of large datasets in cluster.

Apache Spark provides tightly integrated extensive toolset or stack that encompasses important needs of Big Data analytic work such as machine learning libraries, stream processing, SQL support and graph support. The detailed documentation relevant to modules are provided at [4].

3 Security Logs

Logs are generated by all sorts of applications, servers, network devices, even embedded devices. There are many types of logs created for different purposes such as measuring performance of a system, tracking security related information and debugging applications.

1 http://spark.apache.org.
2 http://hadoop.apache.org.
3 http://www.mathworks.com.
4 http://www.r-project.org.

Many types of logs can be used to track security related information. These logs in broad term can be classified as security or security related logs. These include;

Security software logs that are intentionally generated to track security issues. Some of the security related software are intrusion detection and prevention systems, antivirus and malware software, remote access software, proxy servers, vulnerability management software, authentication servers, routers and firewalls.

Operation system logs such as system events and audit records.

Application logs such as web, database, e-mail, ftp or specific application servers. There are many types of applications that generate logs. From security standpoint, logs of these applications contain useful data for analyzing or tracking security issues. Some of the valuable data found in these logs are account information, client/server interactions, used resources, some other significant events such as failures etc.

3.1 Challenges in Security Log Management

There are challenges in terms of log management, storage and analytics. Some of the challenges are as follows.

Variety of Sources: Logs are generated by different types of applications, even a single application can create different logs for different purposes. These logs need to be combined for management and analyzing purposes.

Variety of Log Contents: Logs created by different applications focus on different contents, level of details are much different.

Variety of Log Formats: There are very few common log formats and fields. It is usually difficult to correlate different logs' contents for specific events. Some logs are created for humans to read, some of them are for log processing applications. Most applications' log schemas are not well defined, and implementing parsers for them is difficult. The terminology used may be different for the same subject matter. The tags of the fields may be abbreviated, misspelled or plain different. The model or the schema of application logs may quickly change with the evolution of applications. Combined logs have generally mixed with structured, semi-structured or unstructured parts originating from different applications which make combined logs practically unstructured.

Variety of Time Formats: Time data is very important to correlate security related events. Different time formats or missing granularity in time greatly affect security analysis in combined logs.

Volume: Log management is becoming increasingly Big Data problem. Even a web sites with moderate traffic, one day log may consume gigabytes of disk space depending on the level of details logs are recorded. The volume of the activities that needs to be logged also impacts storage requirements. The elasticity and scalability of underlying infrastructure may help to handle storage needs.

Velocity: In high volume web sites, transactions create immense logs and aggregating these logs in real time may create bottleneck in the network as well as on log servers depending on the infrastructure. Real time analytic of important logs may not be possible because of the high velocity and the volume of the logs. Log reduction techniques may need to be applied.

Veracity: Authentication of log sources, integrity of logs, the level of noise in the logs, all affect the value of log data. It is easy not to record important event by giving wrong category or filtering the wrong level. It is also easy to generate too much noise and disguise the valuable information in the logs.

3.2 Relevant Work

NIST Special Publication on Log Management [1] is an authoritative resource on logs, regarding their definitions and classifications. The meaning and usage of logging and log unifying abstractions are explained in an instructive manner at [5]. Up-to-date evolution of business log processing is presented at [6] with some architectural examples about managing and unifying logs. At [7], a list of Big Data log processing tools are discussed. Most tools try to provide solutions to some of the challenges discussed above. Analyzing big logs with parallel machine learning algorithms in realtime, and learning from unified logs are presented as new challenges at [8]. Some insights are provided for what to look for during analyzes of security logs are discussed at [9]. A log analyzing application with Spark, which is specific to Apache Web Server logs, is discussed at [10]. However they deal with well defined structured data. In this work, we deal with relatively unstructured data, i.e. mix of structured and unstructured, and analyzing of them.

4 Security Log Analyzes with Spark

Most of the aforementioned challenges require cluster computing support for storage and processing. On top of that, tooling is needed to leverage the underlying cluster. Spark is a cluster or cloud enabled Big Data analytic tool that is very capable environment for tackling some of the challenges described in the previous section. Spark provides detailed API's that can handle text or database queries in parallel. It provides most of the tooling needed for building cluster enabled sophisticated security log analyzing environment including machine learning algorithms for detection and classification of security problems and realtime processing of streams.

To demonstrate Spark log analyzing capabilities a combined security log is analyzed which can be found in many home or small business environment. The network under consideration for this study has around 30 networked devices. The type of devices are a few Network Attached Storage Devices (NAS), several Linux and Windows labtops and PCs, tablets, mobile phones, one firewall, several access points, several security cameras, smart TVs and smart entertainment boxes etc. A syslog server is running on one of the Network attached Storage

(NAS). Nodes with the log forwarding capability, such as firewall, router, some of PCs that are running linux and the NAS' own logs, are directed to the NAS's syslog server. Logs are accumulated in one of the Network File System (NFS) accessible directory of the NAS. Even in such a relatively simple environment around 50 MB of logs are generated daily. At the time of this study around 45 GB of log was accumulated. 45 GB data may or may not be Big Data in terms of available analytic infrastructure. However, for an average analytic environment that is not clustered, it becomes easily Big Data problem in terms of processing capabilities and the size of the analytic data to work on.

For the demonstration of Apache Spark capabilities for analyzing security logs; the most communicated Internet addresses by the network devices will be extracted and the statistics will be gathered. Firewall logs are the primary source of security information. However firewall log messages are intermixed with other log messages from various devices in the combined syslog. The general problem will be tackled as follows.

- Creating a cluster with Hadoop and Apache Spark and storing the log data in Hadoop File System(HDFS)[5].
- Creating a Spark application that does (in parallel)
  - Filtering firewall logs from syslog for outbound traffic
  - Creating MapReduce source destination pairs
  - Grouping most used IPs per source
  - Outputing the statistics

A tiny Spark cluster, with four PC with 2^{nd} and 3^{rd} generation i5 processors and 8 GB of RAM on Ubuntu 14.04 LTS Server OS[6], is created to be able to analyze the network security logs. With Spark and Hadoop HDFS, a tiny cluster with 32 GB of RAM and 12 cores is easily setup from relatively inexpensive consumer hardware. Building Spark and configuring it on the cluster nodes manually are somewhat labor intensive endeavor. However there are many cluster provisioning tools available which handle automatic installation and configuration of cluster nodes. It would be easier to use one with available instructions that can be easily found on the Internet. One such tool is Ansible[7].

Apache Spark provides several programming language interfaces. Because of the very terse and easy-to-use syntax Python API is used. For the solution of the problem, a sample `mostusedips.py` Spark Application is developed. Some important parts of the source code and the explanations are provided at the following section.

5 Analyzing Security Related Data in Cluster

Creating a Spark context is the first step in any Spark application. Loading of data as RDD from HDFS is done via Spark context.

[5] Spark uses HDFS as distributed file system environment amongst cluster nodes.
[6] http://www.ubuntu.com.
[7] http://www.ansible.com.

`lines = sc.textFile("hdfs://path/to/file")`

Listing 1.1 shows a way of filtering a log file based on source and destination IP patterns. All the operations on RDDs are handled in parallel.

`filter_domain_ip` pattern search function passed to `lines.filter` function evaluates all lines in parallel and filter out every line that has no source and destination pattern. Resulting `domain_lines` is an RDD that has only lines with appropriate source and outbound destination IP addresses.

```
import re
srcpat = re.compile(r'src="(192.168.1.\d{1,3}):(\d+)')
dstpat_internal = re.compile(r'dst="(192.168.1.\d{1,3}):(\d+)')
dstpat = re.compile(r'dst="(\d{1,3}.\d{1,3}.\d{1,3}.\d{1,3}):(\d+)')
def filter_domain_ip(line):
    should_filter = (
        (srcpat.search(line) and
            dstpat.search(line)) and
                (not dstpat_internal.search(line))
    )
    return should_filter
domain_lines = lines.filter(filter_domain_ip)
```

Listing 1.1. Filtering Outbound Traffic from Firewall Log Messages

Listing 1.2 shows the extraction of source and destination IP pairs from filtered log lines `domain_lines`, and creating log messages as tuples. An expected output is presented at Listing 1.3. These pairs are the mapping part of the first map-reduce operation.

`extract_src_dst_info` transformation function passed to `domain_lines.map` evaluates all the lines in parallel and transforms every line to a `(src,dest)` pair.

```
def extract_src_dst_info(report):
    srcmatches = srcpat.search(report)
    if srcmatches:
        srcip, srcport = srcmatches.groups()
        dstmatches = dstpat.search(report)
        if dstmatches:
            dstip, dstport = dstmatches.groups()
            return (srcip, dstip)
        else:
            print('ERROR in line {}: dst not matches'.format(report))
    else:
        print('ERROR in line {}: src not matches'.format(report))

src_dst_pairs = domain_lines.map(extract_src_dst_info)
```

Listing 1.2. Extracting `(src,dst)` Pairs

```
    (src1, dst10)
    (src1, dst1)
    (src1, dst2)
    (src2, dst5)
    (src1, dst3)
    (src3, dst7)
    (src1, dst2)
    (src2, dst4)
     ...
```

Listing 1.3. Source-Destination Pairs

With the code presented at Listing 1.4, the (source,destination) pairs at Listing 1.3 are converted to (source,destination) pairs in parallel, which is presented at Listing 1.5.

```
src_grouped_dsts_pairs = src_dst_pairs.groupByKey().mapValues(list)
```

Listing 1.4. Grouping Source and Destination IP Pairs

```
[(src1, [dst10, dst1, dst2, dst3, dst2]),
 (src2, [dst5, dst4]),
 (src3, [dst7]),
 ...],
```

Listing 1.5. Source-Destination List Pairs

With the code presented in at Listing 1.6, (source,[list of destinations]) are converted to (source,[destination,count]) in parallel, which is presented at Listing 1.7.

```
from collections import Counter
src_grouped_counted_dsts_pairs = src_grouped_dsts_pairs.mapValues(lambda
    x: Counter(x).most_common())
```

Listing 1.6. Connection Statistics Generation

```
[(src1, [(dst2,2), (dst1,1), (dst3,1), (dst10,1)]),
 (src2, [(dst5,1), (dst4,1)]),
 (src3, [(dst7,1)]),
 ...],
```

Listing 1.7. Source-Destination List Pairs

5.1 Preliminary Results

The simple Spark application around 50 lines of effective code has proved to be very useful. Around 185 million lines of logs are filtered in parallel by the tiny Spark Cluster by utilizing all available memory and processing power in four nodes. It revealed unexpected results such that some of the devices in the network try to access some Internet addresses without clear reason. For example, it is found out that the security cameras with the same brand are trying to access to certain IP addresses registered in irrelevant countries, even though these addresses are not displayed anywhere in the firmware of the IP cameras.

As this study reveals, some of IP addresses are connected much more frequently and for much longer period of time than the other IPs. This can be a starting point of further security checks. The time and the duration of the communication can be considered for security analyses, and the legality of the connection might need to be checked as well. The current implementation of the application can be extended or other simple applications can also be easily integrated to the system to figure out the most used IP communication end points. The intermediate results such as (source,destination) pairs can be easily stored in a distributed file system and can be reused for further analysis as an input to other applications. The extracted data can be analyzed interactively for different issues in parallel through Apache Spark shell easily. It is just a matter of having average programming skills and some Apace Spark API knowledge. The heavy lifting of parallel operations, distribution of work is handled automatically in disguise by Spark.

6 Conclusion

In this study, it is demonstrated that Apache Spark enables large scale log analysis in interactive and batch processing environments. With its flexible and extensive programming interface and built-in libraries, it is easy to implement sophisticated parallel analytic tasks, without knowing much about the cluster computing details. Apache Spark provides excellent environment for security log analyzes or mining. The demo application can easily be extended and more sophisticated security analyzes can be performed. As a future work, more complex security analyzes will be performed along with benchmark comparisons on cluster with several nodes vs. single node multicore computer based on different sizes of logs. Moreover, by using MLib and streaming API, realtime log analyzing capabilities of Apache Spark will be investigated in security log analyzes.

References

1. Kent, K., Souppaya, M.: Guide to computer security log management. recommendations of the national institute of standards and technology (2006). http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-92.pdf. Accessed 10 Jan 2016
2. Fekete, R.: Log message classification with syslog-ng (2010). http://lwn.net/Articles/369075/. Accessed 10 Jan 2016
3. Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., Franklin, M.J., Shenker, S., Stoica, I.: Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. In: Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation, NSDI 2012, p. 2. USENIX Association, Berkeley (2012)
4. Spark, A.: Apache spark web site. http://spark.apache.org. Accessed 10 Jan 2016
5. Kreps, J.: The log: What every software engineer should know about real-time data's unifying abstraction (2013). https://engineering.linkedin.com/distributed-systems/log-what-every-software-engineer-should-know-about-real-time-datas-unifying. Accessed 14 Jan 2016
6. Dean, A.: The three eras of business data processing (2014). http://snowplowanalytics.com/blog/2014/01/20/the-three-eras-of-business-data-processing. Accessed 14 Jan 2016
7. codecondo.com: 8 tools for log monitoring and processing big data (2014). http://codecondo.com/8-tools-for-log-monitoring-and-processing-big-data. Accessed 14 Jan 2016
8. Kobielus, J.: Big data log analysis thrives on machine learning (2014). http://www.infoworld.com/article/2608064/big-data/big-data-log-analysis-thrives-on-machine-learning.html. Accessed 14 Jan 2016
9. Zeltser, L.: Critical log review checklist for security incidents (2015). https://zeltser.com/security-incident-log-review-checklist. Accessed 14 Jan 2016
10. databricks.com: Log analysis with spark. https://databricks.gitbooks.io/databricks-spark-reference-applications/content/logs_analyzer/index.html. Accessed 14 Jan 2016

Delay Prediction System for Large-Scale Railway Networks Based on Big Data Analytics

Luca Oneto[1](✉), Emanuele Fumeo[1], Giorgio Clerico[1], Renzo Canepa[2], Federico Papa[3], Carlo Dambra[3], Nadia Mazzino[3], and Davide Anguita[1]

[1] DIBRIS - University of Genoa, Via Opera Pia 11A, 16145 Genova, Italy
{luca.oneto,emanuele.fumeo,g.clerico,davide.anguita}@unige.it

[2] Rete Ferroviaria Italiana S.p.A., Via Don Vincenzo Minetti 6/5, 16126 Genoa, Italy
r.canepa@rfi.it

[3] Ansaldo STS S.p.A., Via Paolo Mantovani 3-5, 16151 Genoa, Italy
{federico.papa,carlo.dambra,nadia.mazzino}@ansaldo-sts.com

Abstract. State-of-the-art train delay prediction systems do not exploit historical train movements data collected by the railway information systems, but they rely on static rules built by expert of the railway infrastructure based on classical univariate statistic. The purpose of this paper is to build a data-driven train delay prediction system for large-scale railway networks which exploits the most recent Big Data technologies and learning algorithms. In particular, we propose a fast learning algorithm for predicting train delays based on the Extreme Learning Machine that fully exploits the recent in-memory large-scale data processing technologies. Our system is able to rapidly extract nontrivial information from the large amount of data available in order to make accurate predictions about different future states of the railway network. Results on real world data coming from the Italian railway network show that our proposal is able to improve the current state-of-the-art train delay prediction systems.

Keywords: Intelligent transportation systems · Railway · Delay prediction · Big data · Extreme learning machine · Apache spark

1 Introduction

Big Data Analytics is one of the current trending research interests in the context of railway transportation systems. Indeed, many aspects of the railway world can greatly benefit from new technologies and methodologies able to collect, store, process, analyze and visualize large amounts of data [34,37], e.g. condition

L. Oneto—This research has been supported by the European Union through the projects Capacity4Rail (European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement 605650) and In2Rail (European Union's Horizon 2020 research and innovation programme under grant agreement 635900).

based maintenance of railway assets [9,25], alarm detection with wireless sensor networks [20], passenger information systems [23], risk analysis [8], and the like. In particular, this paper focuses on predicting train delays in order to improve traffic management and dispatching using Big Data Analytics, scaling to large railway networks.

Although trains should respect a fixed schedule called "nominal timetable", delays occur daily because of accidents, repair works, extreme weather conditions, etc., and affect negatively railway operations, causing service disruptions and losses in the worst cases. Rail Traffic Management Systems (TMSs) have been developed to support managing complex rail services and networks and to increase operations efficiency by allowing train dispatching through remote control of signaling systems. By providing accurate train delay predictions to TMSs, it can be possible to greatly improve traffic management and dispatching in terms of passenger information systems and perceived reliability [24], freight tracking systems for improved customers' decision-making, timetable planning [5] by detecting recurrent delays, and delay management (rescheduling) [6].

Due to its key role, TMS stores the information about every "train movement", i.e. every train arrival and departure timestamp and delays at "checkpoints" monitored by signaling systems (e.g. a station, a switch, etc.). Datasets composed of train movements records have been used as fundamental data sources for every work addressing the problem of train delays predictions. For instance, Milinkovic et al. [22] developed a Fuzzy Petri Net (FPN) model to estimate train delays based both on expert knowledge and on historical train movements data. Berger et al. [2] presented a stochastic model for delay propagation and forecasts based on directed acyclic graphs. Goverde, Keckman et al. [11,12,17,18] developed an intensive research in the context of delay prediction and propagation by using process mining techniques based on innovative timed event graphs, on historical train movements data, and on expert knowledge about railway infrastructure. However, their models are based on classical univariate statistics, while our solution integrates multivariate statistical concepts that allow our models to be extended in the future by including other kind of data (e.g. weather forecasts, passenger flows, etc.). Moreover, these models are not especially developed for Big Data technologies, possibly limitating their adoption for large scale networks. Last but not least, S. Pongnumkul et al. [29] worked on data-driven models for train delays predictions, treating the problem as a time series forecast problem. The described system investigates the application of ARIMA and k-NN models over limited train data, making it unsuitable for Big Data.

For these reasons, this paper investigates the problem of predicting train delays for large scale railway networks by treating it as a time series forecast problem where every train movement represents an event in time, and by exploiting Big Data Analytics methodologies. Delay profiles for each train are used to build a set of data-driven models that, working together, make possible to perform a regression analysis on the past delay profiles and consequently to predict the future ones. The data-driven models exploit a well-know Machine

Learning algorithm, i.e. the Extreme Learning Machines (ELMs), which has been adapted to exploit typical Big Data parallel architectures. Moreover, the data have been analyzed by using state-of-art Big Data technologies, i.e. Apache Spark on Apache Hadoop, so that it can be used for large scale railway networks. The described approach and the prediction system performance have been validated based on the real historical data provided by Rete Ferroviaria Italiana (RFI), the Italian Infrastructure Manager (IM) that controls all the traffic of the Italian railway network. For this purpose, a set of novel Key Performance Indicators (KPIs) agreed with RFI has been designed and used. Several months of records from the entire Italian railway network have been exploited to show that the new proposed methodology outperforms the current technique used by RFI to predict train delays in terms of overall accuracy.

2 Train Delay Prediction Problem: The Italian Case

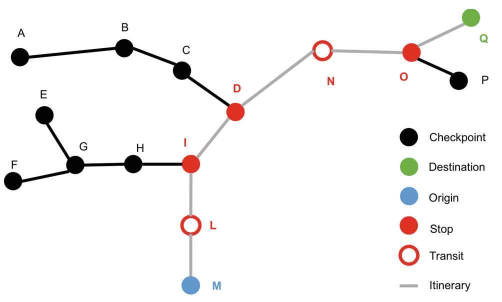


Fig. 1. A railway network depicted as a graph, including a train itinerary from checkpoint M to Q

A railway network can be considered as a graph where nodes represent a series of checkpoints connected one to each other. Any train that runs over the network follows an itinerary defined by a series of n_c checkpoints $\mathscr{C} = \{C_1, C_2, \cdots, C_{n_c}\}$, which is characterized by a station of origin, a station of destination, some stops and some transits (see Fig. 1). For any checkpoint C, the train should arrive at time t_A^C e and should depart at time t_D^C, defined in the nominal timetable (with a precision of 30 s or 1 min). The actual arrival and departure times of the train are defined as $\hat{t}_A^C$ and $\hat{t}_D^C$. The differences $(\hat{t}_A^C - t_A^C)$ and $(\hat{t}_D^C - t_D^C)$ are defined as arrival and departure delays respectively, and a train is considered a "delayed train" if its delay is greater than 30 s or 1 min. A dwell time is defined as the difference between the departure time and the arrival time for a fixed checkpoint $(\hat{t}_D^C - \hat{t}_A^C)$, while a running time is defined as the amount of time needed to depart from the first of two subsequent checkpoints and to arrive to the second one $(\hat{t}_A^{C+1} - \hat{t}_D^C)$.

In order to tackle the problem of train delays predictions, we propose the following solution. Taking into account the itinerary of a train, for each checkpoint C_i where $i \in \{0, 1, \cdots, n_c\}$, we want to be able to predict the train delays for each subsequent checkpoint C_j with $j \in \{i + 1, \cdots, n_c\}$. Note that C_0 represents the train before its departure from the origin station.

In this solution, the train delays predictions problem is treated as a time series forecast problem, where a set of predictive models perform a regression analysis over the delay profiles for each train, for each checkpoint C_i of the itineraries of these trains, and for each subsequent checkpoint C_j with $j \in \{i+1, \cdots, n_c\}$. The models are built by exploiting a slightly modified version of the popular ELM algorithm. The historical records about arrivals and departures for each train at each checkpoint, as well as additional derived information such as dwell times and running times, are used as the input of the algorithm. For example, Fig. 2 shows the data needed to build forecasting models based on the railway network depicted in Fig. 1. Basically, based on the state of the railway network between time $(t - \delta^-)$ and time t, we want to predict the network state from time t to $(t + \delta^+)$ and this is nothing but a classical regression problem [28]. To sum up, for each train characterized by a specific itinerary of n_c checkpoints, we need to build n_c models for C_0, $(n_c - 1)$ for C_1, and so on. Consequently, the total number of models to be built for each train can be calculated as $n_c + (n_c - 1) + \cdots + 1 = {}^{n_c(n_c-1)}/_2$. These models work together in order to make possible to estimate the delays of a particular train during its entire itinerary.

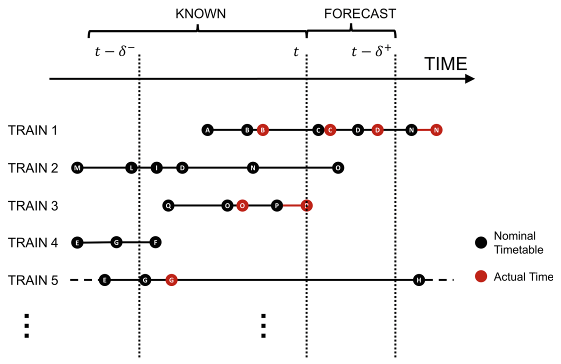


Fig. 2. Data for the train delay forecasting models for the network of Fig. 1

Considering the case of the Italian railway network, RFI controls every day approximately 10 thousand trains traveling along the national railway network. Every train is characterized by an itinerary composed of approximately 12 checkpoints, which means that the number of train movements is greater than or equal to 120 thousands per day. This results in roughly one message per second and more than 10 GB of messages per day to be stored. Note that every time that we retrieve a complete set of messages describing the entire planned itinerary of a particular train for one day, the predictive models associated with that train must be retrained. Since for each train we need to build at least ${}^{n_c(n_c-1)}/_2 \approx 60$ models, the number of models that has to be retrained every day in the Italian case is greater than or equal to 600 thousands.

3 Delay Prediction System for Large-Scale Railway Networks

Let us consider a regression problem [35] where $\mathscr{X} \in \mathbb{R}^d$ is the input space and $\mathscr{Y} \in \mathbb{R}$ the output one. Moreover, let us consider a set of examples of the mapping $\mathscr{D}_n : \{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_n\}$ of cardinality n, where $\boldsymbol{z}_{i \in \{1,\cdots,n\}} = (\boldsymbol{x}_i, y_i)$ with $\boldsymbol{x} \in \mathscr{X}$ and $y \in \mathscr{Y}$. A learning algorithm $\mathscr{A}_{\mathscr{H}}$, characterized by a set of hyperparameters $\mathscr{H}$ that must be tuned, maps $\mathscr{D}_n$ into a function $f : \mathscr{A}_{(\mathscr{D}_n, \mathscr{H})}$ from $\mathscr{X}$ to $\mathscr{Y}$. The accuracy of a function $f : \mathscr{A}_{(\mathscr{D}_n, \mathscr{H})}$ in representing the hidden relationship between input and output space is measured with reference to a loss function $\ell(f, \boldsymbol{z}) : \mathscr{F}_{\mathscr{H}} \times (\mathscr{X} \times \mathscr{Y}) \to \mathbb{R}$. The quantity which we are interested in is the generalization error [1,35], namely the error that a model will perform on new data generated by μ and previously unseen $L(f) = \mathbb{E}_{\boldsymbol{z}} \ell(f, \boldsymbol{z})$. Unfortunately, since μ is unknown, $L(f)$ cannot be computed and, consequently, must be estimated. The most common empirical estimator is the empirical error $\widehat{L}(f) = \frac{1}{n} \sum_{\boldsymbol{z} \in \mathscr{D}_n} \ell(f, \boldsymbol{z})$.

The Extreme Learning Machines (ELM) algorithm is a state-of-the-art tool for regression problems [3,13,15] and was introduced to overcome problems posed by back-propagation training algorithm [32]: potentially slow convergence rates, critical tuning of optimization parameters, and presence of local minima that call for multi-start and re-training strategies. ELM was originally developed for the single-hidden-layer feedforward neural networks. The weight of the hidden layer, contrarily to the back-propagation, are randomly assigned while the weights of the output layer are found via Regularized Least Squares (RLS) [14–16]. More formally a vector of weighted links, $\boldsymbol{w} \in \mathbb{R}^h$, connects the hidden neurons to the output neuron without any bias

$$f(\boldsymbol{x}) = \sum_{i=1}^{h} w_i \boldsymbol{\varphi} \left(W_{i,0} + \sum_{j=1}^{d} W_{i,j} x_j \right). \tag{1}$$

where $\boldsymbol{\varphi} : \mathbb{R} \to \mathbb{R}$ is a nonlinear activation function of the hidden neurons and $W \in \mathbb{R}^{h \times (0,\cdots,d)}$, the weight of the hidden units, are set randomly. Finally the weight of the output layer $\boldsymbol{w} \in \mathbb{R}^h$ are found by solving the following RLS problem

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \|A\boldsymbol{w} - \boldsymbol{y}\|^2 + \lambda \|\boldsymbol{w}\|^2 = (A^T A + \lambda I)^{-1} A^T y, \tag{2}$$

where $A \in \mathbb{R}^{n \times h}$, $A_{u,v} = \boldsymbol{\varphi} \left(W_{v,0} + \sum_{j=1}^{d} W_{v,j} x_{u,j} \right)$. Consequently we can see A as a concatenation of the random projection of vectors $\boldsymbol{x}_{\{i \in \{1,\cdots,n\}\}}$ based on the hidden neuron of the ELM: $A = [\boldsymbol{\phi}(\boldsymbol{x}_1), \cdots, \boldsymbol{\phi}(\boldsymbol{x}_n)]^T$.

Algorithm 1. SGD for ELM.

```
   Input: $\mathscr{D}_n$, $\lambda$, $\tau$, $n_{\text{iter}}$
   Output: $\boldsymbol{w}$
1  Read $\mathscr{D}_n$ ;
2  Compute $A$ ;
3  $\boldsymbol{w} = \boldsymbol{0}$ ;
4  for $t \leftarrow 1$ to $n_{iter}$ do
5      $\boldsymbol{w} = \boldsymbol{w} - \frac{\tau}{\sqrt{t}} \frac{\partial}{\partial \boldsymbol{w}} \left[ \|A\boldsymbol{w} - \boldsymbol{y}\|^2 + \lambda \|\boldsymbol{w}\|^2 \right]$;
6  return $(\boldsymbol{w}, b)$;
```

Spark [21,36] is a state-of-the-art framework for high performance in-memory parallel computing. The main idea behind the Spark technology is that we have to reduce access to the disk as much as possible and make as much computation as possible in memory. Moreover, since Spark is designed to efficiently deal with iterative computational procedures that recursively perform operations over the same data, it may not be efficient to compute the solution in the form of Eq. (2). Consequently, instead of solving the inversion problem of Eq. (2), let us adopt a Stochastic Gradient Descent (SGD) algorithm. The SGD algorithm is a very general optimization algorithm, which is efficiently able to solve a problem in the following form: $\min_{f \in \mathscr{F}_{\mathscr{H}}} \widehat{L}(f) + \lambda R(f)$, where $R(f)$ is a regularizer [21]. λ balances the tradeoff between the over- and under-fitting tendency of the algorithm. Based on the choice of $R(f)$ and $\widehat{L}(f)$ we can retrieve different algorithms [21]. If we set $R(f) = \|\boldsymbol{w}\|^2$ and $\widehat{L}(f) = {}^1\!/_n \sum_{i=1}^{n} [f(\boldsymbol{x}_i) - y_i]^2$ we get the ELM formulation of Eq. (2). The Stochastic Gradient Descent (SGD) algorithm for ELM is reported in Algorithm 1 [33].

Algorithm 2. SGD for ELM on Spark ($d \geq h$)

```
  Input: $\mathscr{D}_n$, $\lambda$, $\tau$, $n_{\text{iter}}$
  Output: $\boldsymbol{w}$
1 Read $\mathscr{D}_n$ ;
2 Compute $A$ /* Compute the projection $\boldsymbol{\phi}$ */
3 $\boldsymbol{w} = 0$;
4 for $t \leftarrow 1$ to $n_{iter}$ do
5     $\boldsymbol{g} = (A, \boldsymbol{y})$.map(Gradient())
      /* Compute the gradient for each sample */
6     .reduce(Sum())
      /* Sum all the gradients of each sample */
7     $\boldsymbol{w} = \boldsymbol{w} - \frac{\tau}{\sqrt{t}}\boldsymbol{g}$ ;
8 return $\boldsymbol{w}$;
```

In Algorithm 1 τ and n_{iter} are parameters related with the speed of the optimization algorithms. Therefore, usually τ and n_{iter} are set based on the experience of the user. In any case τ and n_{iter} can be seen as other regularization terms as λ since they are connected with the early stopping regularization technique [4,30].

Algorithm 1 is well-suited for implementation in Spark and many of these tools are already available in MLlib [21]. Basically, the implementation of Algorithm 1 reported in Algorithm 2 is an application of two functions: a map for the computation of the gradient and a reduction function for the sum of each single gradient.

The main problem of Algorithm 2 is the computation and storage of A. If $h \ll d$ it means that $A \in \mathbb{R}^{n \times h}$ will be much smaller than the dataset which belongs to $\mathbb{R}^{n \times d}$. In this case, it is more appropriate to compute it before the SGD algorithms starts the iterative process and keep it in memory (note that the computation of A is fully parallel). In this way all the data $\mathbb{R}^{n \times d}$ projected by $\boldsymbol{\phi}$ into to matrix $A \in \mathbb{R}^{n \times h}$ can be largely kept in volatile memory (RAM) instead of reading from the disk. If instead $h \gg d$, employing Algorithm 2 we risk that $A \in \mathbb{R}^{n \times h}$ does not fit into the RAM, consequently making too many accesses to the disk. For this reason, we adopt two different strategies:

– if h is approximately the same magnitude or smaller than d we use Algorithm 2 and we compute the matrix A at the beginning;

– if $h \gg d$ we adopt Algorithm 3 where $\boldsymbol{\phi}(\boldsymbol{x}_i)$ is computed online in order to avoid to read the data from the disk.

Quite obviously, the limit is given by the size of the RAM of each node and the number of nodes. Until the algorithm is able to keep most of the data in memory, it is better to use Algorithm 2. Algorithm 3 allows us to partially reduce the effect of having to access the data on the disk by paying the price of computing $\boldsymbol{\phi}(\boldsymbol{x}_i)$ online. In fact, Algorithm 3 does not precompute $A \in \mathbb{R}^{n \times h}$ at the beginning but it keeps in memory the data $\mathscr{D}_n$ and, at every iteration of the SGD algorithm, it computes online both the projection induced by $\boldsymbol{\phi}$ and the gradient. Consequently, there is no need to store $A \in \mathbb{R}^{n \times h}$.

Algorithm 3. SGD for ELM on Spark ($d \leq h$).

```
   Input: $\mathscr{D}_n$, $\lambda$, $\tau$, $n_{\text{iter}}$
   Output: $\boldsymbol{w}$
1  Read $\mathscr{D}_n$ ;
2  $\boldsymbol{w} = 0$;
3  for $t \leftarrow 1$ to $n_{iter}$ do
4      $\boldsymbol{g} = \mathscr{D}_n$.map($\boldsymbol{\phi}$&Gradient())
       /* Compute both the projection $\phi$ and the gradient for each sample    */
5      .reduce(Sum())
       /* Sum all the gradients of each sample                                   */
6      $\boldsymbol{w} = \boldsymbol{w} - \frac{\tau}{\sqrt{t}}\boldsymbol{g}$;
7  return $\boldsymbol{w}$;
```

In this context, the selection of the optimal λ and h remains a fundamental problem, which is still the target of current research [1]. Resampling methods such as k–Fold Cross Validation (KCV), the Leave–One–Out, and the Nonparametric Bootstrap (BTS) [7,19] are favored by practitioners because they work well in many situations and allow the application of simple statistical techniques for estimating the quantities of interest. Unfortunately the computational burden required by KCV and BTS is quite high for this reason the Bag of Little Bootstraps (BLB) will be exploited [26,27].

4 Results on the Italian Case Study

In order to validate our methodology and to assess the performance of the new prediction system, we exploited real data provided by RFI. For the purpose of this work, RFI gave access to almost one year of data related to two main areas in Italy. The data included more than a thousand trains and two hundred checkpoints. Note that, the information has been anonymized for privacy and security concerns.

Our approach to the experiments consisted in (i) building for each train in the dataset the needed set of models based on the ELM algorithm, (ii) applying the models to the current state of the trains, and finally (iii) validating the models in terms of performance based on what really happens in a future instant. Consequently, we performed simulations for all the trains included in the dataset simulating the online-approach that updates predictive models every day, so to take advantage of new information as soon as they becomes available.

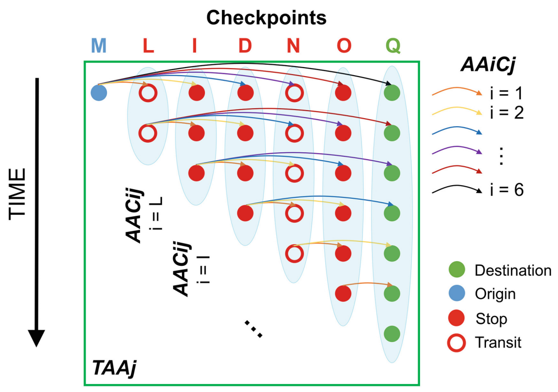


Fig. 3. KPIs for the train and the itinerary depicted in Fig. 1

We compared the results of our simulations with the results of the current train delay prediction system used by RFI, which is quite similar the one described in [12]. In order to fairly assess the performance of the two systems, a set of novel KPIs agreed with RFI has been designed and exploited. Since the purpose of this work was to build predictive models able to forecast the train delays, these KPIs represent different indicators of the quality of these predictive models. Based on these considerations, three different indicators have been used, which are also proposed in Fig. 3 in a graphical fashion:

- *Average Accuracy at the i-th following Checkpoint for train j (AAiCj):* for a particular train j, we average the absolute value of the difference between the predicted delay and its actual delay, at the i-th following Checkpoint with respect to the actual Checkpoint.
- *AAiC:* is the average over the different trains j of AAiCj
- *Average Accuracy at Checkpoint-i for train j (AACij):* for a particular train j, the average of the absolute value of the difference between the predicted delay and its actual delay, at the i-th checkpoint, is computed.
- *AACi:* is the average over the different trains j of AACij
- *Total Average Accuracy for train j (TAAj):* is the average over the different checkpoint i of AACij (or equivalently the average over the index i of AAiCj).
- *TAA:* is the average over the different trains j of TAAj

We ran the experiments by exploiting the Google Compute Engine [10] on the Google Cloud Platform. We employed a four-machines cluster, each one equipped with two cores (machine type n1-highcpu-2), 1.8 GB of RAM and an HDD of 30 GB. We use Spark 1.5.1 running on Hadoop 2.7.1 configured analogously to [31]. The set of possible configurations of hyperparameters is defined as a set $\mathscr{H}$ where $\mathscr{H} = \{(h, \lambda) : h \in \mathscr{G}_h, \lambda \in \mathscr{G}_\lambda\}$ with $\mathscr{G}_h = \left\{\lfloor 10^{\{1,1.2,\cdots,3.8,4\}} \rfloor\right\}$ and $\mathscr{G}_\lambda = 10^{\{-6,-5.5,\cdots,3.5,4\}}$. Finally, as suggested by the RFI experts, $t_0 - \delta^-$ is set equal to the time, in the nominal timetable, of the origin of the train.

Table 1. ELM based and RFI prediction systems KPIs (in minutes).

AAiCj	ELM	RFI	ELM	RFI	ELM	RFI	ELM	RFI	ELM	RFI	
j \ i	1st		2nd		3rd		4th		5th		
1	1.6 ± 0.1	1.8 ± 0.5	1.8 ± 0.2	2.1 ± 0.8	2.1 ± 0.1	2.3 ± 0.3	2.3 ± 0.2	2.5 ± 0.6	2.4 ± 0.1	2.7 ± 0.4	
2	1.8 ± 0.1	3.2 ± 1.7	1.9 ± 0.1	3.4 ± 0.1	2.2 ± 0.1	3.8 ± 3.2	2.4 ± 0.0	4.2 ± 0.6	2.6 ± 0.5	4.6 ± 0.1	
3	1.4 ± 0.1	1.9 ± 0.2	1.6 ± 0.5	2.0 ± 0.2	1.8 ± 0.1	2.3 ± 0.1	1.9 ± 0.4	2.6 ± 1.0	2.0 ± 0.2	2.8 ± 0.4	
4	1.5 ± 0.1	2.0 ± 0.1	1.6 ± 0.0	2.2 ± 1.1	1.9 ± 0.1	2.6 ± 0.8	2.1 ± 0.4	3.0 ± 0.7	2.3 ± 0.4	3.4 ± 0.4	
5	0.9 ± 0.0	1.4 ± 0.2	1.0 ± 0.2	1.7 ± 0.1	1.2 ± 0.2	2.0 ± 0.3	1.4 ± 0.0	2.3 ± 1.1	1.6 ± 0.1	2.6 ± 0.0	
6	1.3 ± 0.1	1.4 ± 0.3	1.5 ± 0.2	1.7 ± 0.3	1.8 ± 0.2	2.0 ± 1.9	2.1 ± 0.0	2.3 ± 0.4	2.3 ± 0.0	2.6 ± 0.9	
7	1.0 ± 0.1	1.3 ± 0.4	1.1 ± 0.1	1.4 ± 0.5	1.3 ± 0.1	1.6 ± 0.5	1.5 ± 0.1	1.8 ± 0.4	1.6 ± 0.1	2.0 ± 0.2	
8	1.0 ± 0.2	1.3 ± 1.1	1.3 ± 0.0	1.6 ± 1.0	1.4 ± 0.1	1.9 ± 0.0	1.6 ± 0.1	2.1 ± 0.3	1.7 ± 0.1	2.3 ± 0.4	
9	0.8 ± 0.0	1.2 ± 0.7	0.9 ± 0.0	1.2 ± 0.7	1.0 ± 0.2	1.4 ± 0.0	1.1 ± 0.0	1.5 ± 1.0	1.2 ± 0.1	1.5 ± 0.3	
10	1.0 ± 0.2	1.5 ± 0.1	1.1 ± 0.1	1.6 ± 0.1	1.3 ± 0.0	2.0 ± 0.3	1.5 ± 0.2	2.3 ± 0.3	1.6 ± 0.0	2.4 ± 0.6	
11	1.2 ± 0.2	1.4 ± 0.1	1.3 ± 0.2	1.5 ± 0.3	1.5 ± 0.1	1.7 ± 0.0	1.6 ± 0.1	1.9 ± 0.2	1.7 ± 0.3	2.1 ± 0.2	
12	1.6 ± 0.0	2.1 ± 0.8	1.9 ± 0.2	2.6 ± 0.8	2.1 ± 0.2	3.1 ± 1.6	2.3 ± 0.3	3.5 ± 0.6	2.6 ± 0.0	3.8 ± 0.1	
13	0.9 ± 0.1	1.2 ± 0.5	1.0 ± 0.2	1.3 ± 0.4	1.1 ± 0.1	1.4 ± 0.5	1.3 ± 0.0	1.6 ± 0.4	1.4 ± 0.1	1.6 ± 0.4	
14	2.1 ± 0.2	3.1 ± 0.7	2.3 ± 0.3	3.5 ± 0.8	-	-	-	-	-	-	… -
15	1.6 ± 0.2	2.2 ± 0.4	1.8 ± 0.1	2.4 ± 0.5	2.0 ± 0.2	2.8 ± 1.1	2.1 ± 0.1	3.1 ± 0.7	2.1 ± 0.0	3.2 ± 1.2	
16	1.8 ± 0.1	2.7 ± 0.3	2.1 ± 0.1	2.9 ± 0.4	2.4 ± 0.0	3.4 ± 0.0	2.6 ± 0.6	3.9 ± 1.6	2.9 ± 0.0	4.1 ± 0.6	
17	1.7 ± 0.1	2.3 ± 1.5	1.9 ± 0.2	2.5 ± 0.8	2.2 ± 0.1	2.9 ± 0.1	2.3 ± 0.3	3.3 ± 1.4	2.4 ± 0.2	3.4 ± 0.4	
18	1.1 ± 0.1	2.8 ± 1.6	1.4 ± 0.0	3.3 ± 0.6	1.6 ± 0.0	4.3 ± 0.7	1.8 ± 0.0	4.7 ± 0.0	2.1 ± 0.2	4.2 ± 1.2	
19	1.7 ± 0.0	2.3 ± 0.6	1.7 ± 0.1	2.4 ± 0.9	1.9 ± 0.0	2.5 ± 1.2	2.0 ± 0.0	2.7 ± 1.8	2.1 ± 0.2	2.7 ± 0.1	
20	2.3 ± 0.5	3.7 ± 1.7	2.5 ± 0.1	4.2 ± 0.8	2.8 ± 0.3	4.7 ± 1.4	3.0 ± 0.2	5.2 ± 0.3	3.3 ± 0.2	5.6 ± 0.1	
21	2.5 ± 0.0	2.5 ± 0.5	2.6 ± 0.1	2.6 ± 0.1	2.8 ± 0.2	2.8 ± 1.3	3.0 ± 0.4	3.0 ± 0.5	3.2 ± 0.2	3.2 ± 0.1	
22	1.9 ± 0.3	3.7 ± 0.5	2.1 ± 0.3	4.0 ± 0.9	2.3 ± 0.0	4.3 ± 1.4	2.5 ± 0.3	4.6 ± 2.1	2.7 ± 0.1	5.0 ± 0.7	
					…						
AAiC	1.6 ± 0.1	3.0 ± 0.8	1.7 ± 0.1	2.9 ± 1.3	2.0 ± 0.4	3.2 ± 1.6	2.2 ± 0.5	3.4 ± 0.0	2.4 ± 0.2	3.4 ± 0.0	

AACij	ELM	RFI	ELM	RFI	ELM	RFI	ELM	RFI	
j \ i	1		2		3		4		
1	2.3 ± 0.4	2.9 ± 0.4	-	-	-	-	-	-	
2	0.1 ± 0.0	0.0 ± 0.0	-	-	-	-	-	-	
3	0.0 ± 0.0	0.2 ± 0.1	-	-	-	-	-	-	
4	1.5 ± 0.0	1.7 ± 0.2	-	-	1.8 ± 0.2	2.3 ± 0.7	-	-	
5	-	-	-	-	1.1 ± 0.0	1.8 ± 0.2	-	-	
6	-	-	-	-	1.5 ± 0.1	1.8 ± 0.8	-	-	
7	-	-	-	-	0.8 ± 0.0	1.1 ± 0.1	-	-	
8	-	-	-	-	1.2 ± 0.0	1.7 ± 0.4	-	-	
9	-	-	-	-	0.7 ± 0.1	0.7 ± 0.1	-	-	
10	-	-	-	-	1.0 ± 0.0	1.3 ± 0.0	-	-	
11	-	-	-	-	-	-	-	-	
12	-	-	1.8 ± 0.2	2.3 ± 0.5	-	-	-	-	
13	-	-	-	-	-	-	-	-	
14	-	-	-	-	-	-	-	-	…
15	-	-	-	-	-	-	-	-	
16	-	-	-	-	-	-	-	-	
17	-	-	-	-	-	-	-	-	
18	-	-	-	-	-	-	-	-	
19	-	-	-	-	-	-	1.3 ± 0.1	1.6 ± 0.1	
20	-	-	-	-	-	-	3.0 ± 0.6	5.1 ± 0.6	
21	-	-	-	-	-	-	2.3 ± 0.0	1.9 ± 1.4	
22	-	-	-	-	-	-	1.7 ± 0.1	2.1 ± 1.0	
				…					
AACi	1.5 ± 0.1	3.3 ± 1.5	1.3 ± 0.1	1.8 ± 0.1	1.1 ± 0.1	1.6 ± 0.1	2.1 ± 0.1	2.9 ± 0.6	

	TAAj	
j	ELM	RFI
1	2.0 ± 0.1	2.2 ± 1.5
2	2.2 ± 0.1	4.3 ± 0.1
3	1.6 ± 0.2	2.3 ± 0.1
4	1.7 ± 0.5	2.4 ± 0.2
5	1.1 ± 0.0	1.7 ± 0.8
6	1.7 ± 0.1	1.9 ± 0.0
7	1.2 ± 0.0	1.5 ± 0.1
8	1.5 ± 0.2	1.9 ± 0.2
9	0.9 ± 0.1	1.4 ± 0.0
10	1.2 ± 0.1	1.8 ± 0.3
11	1.5 ± 0.1	1.8 ± 0.6
12	2.0 ± 0.1	2.8 ± 0.4
13	1.1 ± 0.2	1.4 ± 0.4
14	2.1 ± 0.3	3.1 ± 0.2
15	1.8 ± 0.3	2.6 ± 0.9
16	2.2 ± 0.1	3.1 ± 0.5
17	2.2 ± 0.0	2.7 ± 0.7
18	1.3 ± 0.2	3.3 ± 0.7
19	1.9 ± 0.3	2.5 ± 0.1
20	2.7 ± 0.2	4.3 ± 1.0
21	2.9 ± 0.5	3.0 ± 0.1
22	2.3 ± 0.5	4.8 ± 1.3
	…	
TAA	2.0 ± 0.3	3.3 ± 1.6

In Table 1 we have reported the KPIs for the proposed prediction system and for the one currently used by RFI. Note that Trains and Checkpoints IDs have been anonymized, and that only part of the results have been included in the paper because of space constraints. Although the RFI system has shown to be robust and accurate during our simulations, our data-driven prediction system managed to outperform it by a factor of $\times 1.65$ for TAA, since it is able to infer a model which takes into account the entire state of the network and not just the local dependencies. The accuracy of the ELM-based method is quite homogeneous with respect to different trains and stations. Moreover, AAiCj grows with i as expected (the further is the prediction in the future, the more uncertain is the prediction itself). Finally, note that there are blank spaces in the tables: this means that, for example, for AAiCj some trains have less checkpoints (at least in the dataset that we analysed), or for AACij different trains run over different checkpoints.

As final issue we would like to underline that if just Algorithm 2 is exploited and not the proposal of combining Algorithms 2 and 3 based on d and h we have that in the first case in order to train all the models of our simulation 15 h are needed while in the second case approximately one hour is enough. In other words, our optimization strategy is 15 time faster than the naive approach.

Our future works will take into account also exogenous information available from external sources. For example, we will include in the models information about passenger flows by using touristic databases, about weather conditions by using the Italian National Weather Service, about railway assets conditions, or any other source of data which may affect railway dispatching operations.

References

1. Anguita, D., Ghio, A., Oneto, L., Ridella, S.: In-sample and out-of-sample model selection and error estimation for support vector machines. IEEE Trans. Neural Netw. Learn. Syst. **23**(9), 1390–1406 (2012)
2. Berger, A., Gebhardt, A., Müller-Hannemann, M., Ostrowski, M.: Stochastic delay prediction in large train networks. In: OASIcs-OpenAccess Series in Informatics (2011)
3. Cambria, E., Huang, G.B.: Extreme learning machines. IEEE Intell. Syst. **28**(6), 30–59 (2013)
4. Caruana, R., Lawrence, S., Lee, G.: Overfitting in neural nets: backpropagation, conjugate gradient, and early stopping. In: Neural Information Processing Systems (2001)
5. Cordeau, J.F., Toth, P., Vigo, D.: A survey of optimization models for train routing and scheduling. Transp. Sci. **32**(4), 380–404 (1998)
6. Dollevoet, T., Corman, F., D'Ariano, A., Huisman, D.: An iterative optimization framework for delay management and train scheduling. Flex. Serv. Manuf. J. **26**(4), 490–515 (2014)
7. Efron, B., Tibshirani, R.J.: An Introduction to the Bootstrap. Chapman & Hall, New York (1993)
8. Figueres-Esteban, M., Hughes, P., Van Gulijk, C.: The role of data visualization in railway big data risk analysis. In: European Safety and Reliability Conference (2015)
9. Fumeo, E., Oneto, L., Anguita, D.: Condition based maintenance in railway transportation systems based on big data streaming analysis. In: The INNS Big Data conference (2015)
10. Google: Google Compute Engine (2016). https://cloud.google.com/compute/. Accessed 3 May 2016
11. Goverde, R.M.P.: A delay propagation algorithm for large-scale railway traffic networks. Transp. Res. Part C: Emerg. Technol. **18**(3), 269–287 (2010)
12. Hansen, I.A., Goverde, R.M.P., Van Der Meer, D.J.: Online train delay recognition and running time prediction. In: IEEE International Conference on Intelligent Transportation Systems (2010)

13. Huang, G., Huang, G.B., Song, S., You, K.: Trends in extreme learning machines: a review. Neural Netw. **61**, 32–48 (2015)
14. Huang, G.B., Chen, L., Siew, C.K.: Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans. Neural Netw. **17**(4), 879–892 (2006)
15. Huang, G.B., Zhou, H., Ding, X., Zhang, R.: Extreme learning machine for regression and multiclass classification. IEEE Trans. Syst. Man Cybern. Part B: Cybern. **42**(2), 513–529 (2012)
16. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: a new learning scheme of feedforward neural networks. In: IEEE International Joint Conference on Neural Networks (2004)
17. Kecman, P.: Models for predictive railway traffic management (Ph.D. thesis). TU Delft, Delft University of Technology (2014)
18. Kecman, P., Goverde, R.M.P.: Online data-driven adaptive prediction of train event times. IEEE Trans. Intell. Transp. Syst. **16**(1), 465–474 (2015)
19. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: International Joint Conference on Artificial Intelligence (1995)
20. Li, H., Parikh, D., He, Q., Qian, B., Li, Z., Fang, D., Hampapur, A.: Improving rail network velocity: a machine learning approach to predictive maintenance. Transp. Res. Part C: Emerg. Technol. **45**, 17–26 (2014)
21. Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Amde, M., Owen, S., Xin, D., Xin, R., Franklin, M.J., Zadeh, R., Zaharia, M., Talwalkar, A.: MLlib: machine learning in apache spark. J. Mach. Learn. Res. **17**(34), 1–7 (2016)
22. Milinković, S., Marković, M., Vesković, S., Ivić, M., Pavlović, N.: A fuzzy petri net model to estimate train delays. Simul. Model. Prac. Theor. **33**, 144–157 (2013)
23. Morris, C., Easton, J., Roberts, C.: Applications of linked data in the rail domain. In: IEEE International Conference on Big Data (2014)
24. Müller-Hannemann, M., Schnee, M.: Efficient timetable information in the presence of delays. In: Ahuja, R.K., Möhring, R.H., Zaroliagis, C.D. (eds.) Robust and Online Large-Scale Optimization. LNCS, vol. 5868, pp. 249–272. Springer, Heidelberg (2009). doi:10.1007/978-3-642-05465-5_10
25. Núñez, A., Hendriks, J., Li, Z., De Schutter, B., Dollevoet, R.: Facilitating maintenance decisions on the dutch railways using big data: the aba case study. In: IEEE International Conference on Big Data (2014)
26. Oneto, L., Orlandi, I., Anguita, D.: Performance assessment and uncertainty quantification of predictive models for smart manufacturing systems. In: IEEE International Conference on Big Data (Big Data) (2015)
27. Oneto, L., Pilarz, B., Ghio, A., D., A.: Model selection for big data: algorithmic stability and bag of little bootstraps on gpus. In: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (2015)
28. Packard, N.H., Crutchfield, J.P., Farmer, J.D., Shaw, R.S.: Geometry from a time series. Phys. Rev. Lett. **45**(9), 712 (1980)
29. Pongnumkul, S., Pechprasarn, T., Kunaseth, N., Chaipah, K.: Improving arrival time prediction of thailand's passenger trains using historical travel times. In: International Joint Conference on Computer Science and Software Engineering (2014)
30. Prechelt, L.: Automatic early stopping using cross validation: quantifying the criteria. Neural Netw. **11**(4), 761–767 (1998)
31. Reyes-Ortiz, J.L., Oneto, L., Anguita, D.: Big data analytics in the cloud: spark on hadoop vs mpi/openmp on beowulf. In: The INNS Big Data Conference (2015)

32. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Cogn. Model. **5**(3), 1 (1988)
33. Shoro, A.G., Soomro, T.R.: Big data analysis: apache spark perspective. Glob. J. Comput. Sci. Technol. **15**(1) (2015)
34. Thaduri, A., Galar, D., Kumar, U.: Railway assets: a potential domain for big data analytics. In: The INNS Big Data conference (2015)
35. Vapnik, V.N.: An overview of statistical learning theory. IEEE Trans. Neural Netw. **10**(5), 988–999 (1999)
36. Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., Franklin, M.J., Shenker, S., Stoica, I.: Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. In: USENIX Conference on Networked Systems Design and Implementation (2012)
37. Zarembski, A.M.: Some examples of big data in railroad engineering. In: IEEE International Conference on Big Data (2014)

An Empirical Comparison of Methods for Multi-label Data Stream Classification

Konstantina Karponi(✉) and Grigorios Tsoumakas

Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece
dkarponi@gmail.com, greg@csd.auth.gr

Abstract. This paper studies the problem of multi-label classification in the context of data streams. We discuss related work in this area and present our implementation of several existing approaches as part of the Mulan software. We present empirical results on a real-world data stream concerning media monitoring and discuss and draw a number of conclusions regarding their performance.

Keywords: Multi-label learning · Data streams · Classification · Media monitoring

1 Introduction

So far most of the work of the literature has dealt separately with the classification of data streams and with the classification of multi-label data. However, many real world problems include data which are considered as data streams with multiple labels. This paper focuses on learning from multi-labeled data streams. It reviews the related literature and experimentally compares state-of-the-art multi-label methods, deriving several interesting conclusions.

In Sect. 2, we present some of the existing work regarding multi-label classification and data stream classification, while in Sect. 3 we discuss related work on the classification of multi-label data streams as well as the evaluation methods and metrics that are suggested from the literature. This extensive literature study gives a fairly complete picture of the problem of multi-label data stream classification.

With a view to carrying out benchmarking experiments, some of the algorithms studied in the literature were implemented under the MULAN library [13]. In Sect. 4, we present the set of experiments that were conducted using these algorithms. More specifically, we discuss the real-world data stream we used, which is from the WISE 2014 Challenge [12] and list all of the algorithms along with their necessary settings in order to produce the best results. Moreover, we provide an analysis of the results and discuss the conclusions.

Finally, Sect. 5 discusses the general conclusions of this study along with some interesting future steps of this research.

2 Background

2.1 Multi-label Learning

A large amount of supervised learning methods deal with the analysis of data that are associated with a single label from a set of labels (i.e. values of a discrete target variable). However, in many applications the training examples can be associated with a set of labels from a larger finite set of labels. This kind of data is called multi-label data [11].

More specifically, multi-label learning has caught the attention of many researchers and has begun providing solutions to a growing number of new applications such as semantic annotation of images and videos and music categorization based on emotion. Moreover, the data that are related to documents and websites are most often associated with more than one label.

There are two different main tasks in supervised learning from multi-label data. The first one is multi-label classification, which involves training a model that outputs a bipartition of the set of labels into relevant and irrelevant with respect to a new instance. The second task is called label ranking, namely training a model that outputs a ranked list of labels according to their relevance to a new instance.

Multi-label learning methods can be grouped into two categories: The first one is called Problem Transformation and it consists of independent algorithms that transform the multi-label learning process into one or more classification procedures of one class. On the other hand, the second category is called Algorithm Adaptation. The methods of this case extend some already existing algorithms in order to manage directly the multi-label data.

Two of the most popular libraries for multi-label learning are MULAN and MEKA. MULAN [13] is a Java open source library based on Weka and used for training models from multi-label datasets. This library provides a variety of state of the art algorithms for multi-label classification as well as an evaluation framework which calculates a wide variety of multi-label metrics. Similarly, MEKA [8] allows the use of open source implementations of methods in Java and it is based on Weka toolkit. The library contains several basic methods as well as methods for pruned sets and classifier chains, methods of the scientific community, and a wrapper for the MULAN platform.

2.2 Data Stream Classification

Due to advances in hardware and software technology, the automatic on-line data generation and storage has become much more widespread than in previous years. Such data streams impose a number of constraints in terms of memory and execution time to data mining algorithms. Moreover, there is a natural tendency of the data to evolve over time. Therefore data mining algorithms must be able to deal and catch up with this evolution of the data.

Some popular stream classification techniques are:

- Ensemble based classification, which was developed with a view to address the concept drift, the efficiency and the robustness of the classification of data streams [14].
- Very Fast Decision Trees, which is an incremental decision tree learning method based on Hoeffding trees [3].
- On demand classification, where in this case the concept of micro-clusters is applied [1].

In order to solve real-world problems within applications that use data mining and machine learning, several technologies for distributed processing of data streams were designed. This includes the S4 [6], the Apache Storm[1], SAMOA [5] and Apache Samza[2].

On the other hand, MOA (Massive Online Analysis) [2], is a non-distributed software for data stream mining. It is a programming environment that implements algorithms and performs experiments for training a model in real time on evolving data streams.

3 Related Work

Multi-label stream classification recently emerged as an extension of data stream classification in response to applications where the examples are associated with one or more labels. This arises for example in categorizing incoming email or business documents: examples might contain a thematic label as well as a label regarding confidentiality. These labels are explicitly predefined but there might be some correlations between them (i.e. examples belonging to a specific topic, may be classified as confidential). The most significant algorithms that exist for the Multi-label Stream Classification are presented below, while the next section discusses evaluation issues.

3.1 Techniques

In [16], the authors suggest the Multiple Windows Classification method where they deal with concept drift and class imbalance by creating two fixed-sized windows, one for the positive examples and one for the negative. They also used the Binary Relevance transformation along with kNN algorithm as the base classifier.

Another approach on Multi-label Stream Classification, used several random trees in order to maintain the labels that appeared in the data stream. This algorithm is called SMART [4] and it deals with the problem of concept drift and it is able to model the correlation between the labels as well as their joint sparseness.

[1] http://storm.apache.org/.
[2] http://samza.apache.org/.

In [15], a new method that deals with concept drift and class imbalance was introduced. The Binary Relevance model was trained using Active Learning and the method also used a Minimal Classifier Uncertainty sampling function to choose the most informational instance. Moreover, the concept drift problem was addressed using Maximum Posterior weight schema.

MOA extensions that were described in [7] can also be useful in classifying multi-label data streams. The authors compare different types of existing transformations and present several improvements concerning them. Among those methods were Pruned Sets, Pair Wise Classification and Ranking, and Threshold.

One more tree suggestion made in [7], which is based in the Hoeffding Tree, is called Multi-label Hoeffding Tree. Splitting each node is based on the calculation of the Multi-label Information Gain and the base classifier at each leaf consists of Pruned Sets transformation with Naive Bayes.

An improvement of the Multi-label Hoeffding Tree was introduced in [9] and it is using the Class Incremental technique. The authors propose an extra filtering method in order to improve the performance of the model by choosing to train it only with the instances that contain the most frequent combinations of labels.

3.2 Evaluation

An important factor for any 'intelligent' system is the methodology of its evaluation. The solutions provided by the literature regarding data streams are either the Holdout or the Prequential method. In the first case, the evaluation is based on a dataset that has been withheld. This dataset is applied to the model at different times in order to evaluate it. In the latter case, each instance is separately used for the evaluation of the model prior to using it for training it and thus it may improve the accuracy.

The multi-label data classification requires different evaluation metrics from the simple data classification. They are divided in three categories as mentioned in [10,17]: Example-based, Label-based and Ranking-based metrics. The authors of [18] provide an extensive analysis of an evaluation methodology that can be applied to data streams with temporal dependence. The performance of a classifier under consideration is compared to the performance of baseline classifiers. This way we set limits to succeed the desired performance for the classifier that w e examine. The evaluation metric suggested for data streams without temporal dependence is called Kappa Statistic, while the one for data streams with temporal dependence is called Kappa Temporal Statistic.

4 Empirical Comparison

4.1 Data

The initial set of data was collected by DataScouting[3], which scanned a number of Greek articles from May 2013 until September 2013. These articles were

[3] http://www.datascouting.com/.

manually partitioned and their text was extracted via OCR (Optical Character Recognition) software. The text of the articles is represented using the bag-of-words model. For each token that was found in the text for all articles, tf-idf statistic was calculated. Each tf-idf value was normalized by using unit normalization. More information on this dataset and the corresponding challenge can be found in [12].

To effectively carry out the experiments, we selected 16 out of the 203 labels. To maintain balance in the distribution, we ensured that 12 out of 16 tags appeared frequently in the data set, while the remaining 4 rarely occurred. The selection both of the frequent and rare labels was random. As a result, the training set was reduced to 23240 out of 64857 articles and the test set was reduced to 12162 out of 34923 articles.

4.2 Algorithms and Settings

We applied several algorithms on the same dataset in order to draw some conclusions. Regarding transformations, the algorithms that we used were: Binary Relevance with Naive Bayes as the base classifier, Binary Relevance with SGD and more specifically we set Hinge Loss as the loss function. We also used Binary Relevance with SGD but Logistic Regression as the loss function this time and we set the regularization parameter to 6. On the other hand, we used the same algorithms with the difference that the Binary Relevance method was executing incrementally. Thus, the algorithms in this case were Binary Relevance Updateable with Naive Bayes Updateable, Binary Relevance Updateable with SGD and Hinge Loss as the loss function and finally Binary Relevance Updateable with SGD and Logistic Regression as the loss function as well as the regularization parameter set to 6. For every incremental algorithm we used prequential evaluation to test the test set.

In addition, we tried the Multi-label Windows Classifier with kNN as the base classifier. The number of neighbors was set to 11 and we preferred to set the size of the positive examples window to 400 while the negative examples window to 600. Furthermore, the value of thresholding update was set to 1000 examples.

Regarding trees, we tried three different algorithms. Firstly, we tested the Multi-label Hoeffding Tree with Pruned Sets Updateable transformation and Naive Bayes Updateable at the leaves. The pruning value was declared to 1 and the tree leaves were initialized with 1/3 of the training set. The rest of the training set was used to train the whole model and we applied prequential evaluation. Similarly, we tried the Multi-label Hoeffding Tree, but this time we used the Class Incremental method at the leaves along with the modified Naive Bayes Class Incremental method. The size of the buffer for the Class Incremental method was set to 200 and the pruning value to 1. As previously the tree leaves were initialized with 1/3 of the training set while the rest of it was used to train the whole model. Prequential evaluation was also applied in this case.

Lastly, we trained a model with the SMART algorithm where the number of random trees was set to 20, their height to 15 and the fading factor to 200. As in all of the incremental cases, the test set was evaluated prequentially.

4.3 Results and Analysis

Table 1 presents the experimental results. Rows correspond to different evaluation measures and columns to different methods. Color highlighting is used to show the top (green), bottom (red) and average (orange) results.

Table 1. Experimental results.

	BR (NB)	BRU (NBU)	BR (SGD-HL)	BRU (SGD-HL)	BR (SGD-LL)	BRU (SGD-LL)	MWC (KNN)	MHT (PS, NBU)	SMART	MHT (CI, NBCI)
Hamming Loss	0,7968	0,7843	0,0188	0,0162	0,0236	0,0203	0,7177	0,1162	0,0947	0,0661
Subset Accuracy	0,0039	0,0054	0,7597	0,7853	0,6815	0,727	0,0001	0,1342	0,2563	0,4596
Example-based Precision	0,0951	0,0998	0,8316	0,8498	0,7503	0,7947	0,0554	0,3368	0,2898	0,5184
Example-based Recall	0,9905	0,9919	0,8228	0,8402	0,7318	0,7837	0,7671	0,5143	0,2726	0,4888
Example-based F Measure	0,1641	0,1701	0,8196	0,8382	0,7342	0,7822	0,1025	0,3876	0,2782	0,4983
Example-based Accuracy	0,0946	0,0993	0,8046	0,825	0,721	0,7683	0,0553	0,3201	0,2726	0,4885
Example-based Specificity	0,1466	0,16	0,9943	0,9958	0,9958	0,9954	0,2464	0,9111	0,9524	0,9677
Micro-averaged Precision	0,0783	0,0795	0,9122	0,9346	0,9254	0,9237	0,0697	0,2982	0,2899	0,5183
Micro-averaged Recall	0,9878	0,9895	0,803	0,8214	0,7128	0,7667	0,7689	0,5157	0,2647	0,4737
Micro-averaged F Measure	0,1451	0,1472	0,8542	0,8744	0,8053	0,8379	0,1279	0,3778	0,2767	0,495
Micro-averaged Specificity	0,1455	0,1588	0,9943	0,9958	0,9958	0,9953	0,2465	0,9108	0,9524	0,9677
Macro-averaged Precision	0,0833	0,0846	0,9316	0,9426	0,9367	0,9326	0,0683	0,3156	0,0743	0,3505
Macro-averaged Recall	0,9916	0,9926	0,6725	0,7047	0,514	0,5829	0,7504	0,483	0,0685	0,2563
Macro-averaged F Measure	0,1406	0,1426	0,7604	0,7893	0,6128	0,6762	0,1158	0,3127	0,0456	0,2584
Macro-averaged Specificity	0,1508	0,1641	0,9935	0,9953	0,9952	0,9948	0,2453	0,9092	0,9389	0,962
Kappa Temporal Statistic	-6,1284	-5,9649	0,7948	0,8144	0,7695	0,7809	-7,2329	-0,048	0,3328	0,43
Kappa Statistic	0,0179	0,0221	0,8115	0,8325	0,778	0,8004	0	0,3272	0,4006	0,4879
Average Precision	0,4502	0,4533	0,8492	0,8705	0,9249	0,9368	0,3875	0,5316	0,4784	0,5731
Coverage	5,9515	5,8554	1,8089	1,522	0,3479	0,307	6,2667	4,3359	3,2921	4,8903
OneError	0,6242	0,6222	0,1634	0,1393	0,125	0,1048	0,739	0,6055	0,7102	0,4817
IsError	0,6692	0,6675	0,2133	0,184	0,1409	0,1192	0,7692	0,6445	0,7237	0,5377
ErrorSetSize	6,0525	5,9466	1,7627	1,4725	0,259	0,2173	6,4053	4,3388	3,3832	4,9749
Ranking Loss	0,3741	0,3683	0,1001	0,0833	0,0158	0,0132	0,3916	0,268	0,208	0,2973
Mean Average Precision	0,0852	0,086	0,649	0,6896	0,8567	0,8655	0,069	0,2085	0,089	0,1905
Geometric Mean Average Precision	0,0439	0,0442	0,6122	0,6563	0,8473	0,857	0,0388	0,1425	0,0491	0,0858
Mean Average Interpolated Precision	0,1013	0,1021	0,6451	0,6738	0,8196	0,8278	0,0869	0,2355	0,0963	0,2202
Geometric Mean Average Interpolated Precision	0,0535	0,0538	0,6146	0,6427	0,8119	0,8208	0,0489	0,1678	0,0538	0,1124
Micro-averaged AUC	0,5667	0,5742	0,8987	0,9086	0,9894	0,9916	0,534	0,7136	0,7949	0,7221
Macro-averaged AUC	0,5712	0,5784	0,833	0,85	0,9831	0,9865	0,5019	0,6965	0,5861	0,6094
Logarithmic Loss	234,8518	231,1672	5,5314	4,7619	1,0164	0,9154	NaN	34,1912	NaN	19,4231

The most significant result observed in the experiments that were conducted is that the algorithms that use incremental transformations outweigh the non-incremental transformations. More specifically, this observation is clearer in the following comparisons:

- Binary Relevance with Naive Bayes - Binary Relevance Updateable with Naive Bayes Updateable
- Binary Relevance in SGD (Hinge Loss) - Binary Relevance Updateable with SGD (Hinge Loss)
- Binary Relevance in SGD (Logistic Regression) - Binary Relevance Updateable with SGD (Logistic Regression)

The results of the experiments show that in general the Binary Relevance Updateable with SGD (Hinge Loss) has the best performance in Example-based and Label-based metrics. Second best appears to be the static version of the aforementioned model. Next in the list we observe the models of Binary Relevance Updateable with SGD (Logistic Regression) and non-incremental form. It is important to emphasize that the last two models mentioned above have the best performance regarding Ranking-based evaluation metrics and therefore they outweigh the ability of ranking classification compared to the rest of the models.

While observing that the values of Example-based F-Measure, Micro-averaged F-measure and Macro-averaged F-measure, we notice that the experiments that contain trees do not perform as good as the transformations mentioned earlier. This is primarily because the trees are not performing at the desired level when it comes to multi-dimensional data, as in our case. However, if we make a comparison between these three tree models, we observe that regarding the Label based and the Example based metrics, Multi-label Hoeffding Tree with Class Incremental learning has the best performance due to the extensive filtering of the examples. For the Ranking based metrics, SMART model outweighs the rest of the models.

Finally, regarding the evaluation measures Kappa Statistic and Kappa Temporal Statistic which evaluate the classification of data streams, the best performance was observed for Binary Relevance Updateable with SGD (Hinge Loss) as well as for Binary Relevance Updateable with SGD (Logistic Regression).

5 Conclusions

In this work we studied and implemented methods to classify data streams with multiple labels. We surveyed existing methods proposed in the literature both for multi-label data classification and for data stream classification. Moreover, we analyzed the evaluation methods and metrics which concern multi-label data classification. Then we implemented the most important methods and as we performed a number of experiments based on the DataScouting's dataset. The experiments contained both models with transformations and base classifiers as well as various other algorithms and models based on trees.

From there the following conclusions were emerged:

– The incremental transformations have better performance than non-incremental (batch learning) transformations, whichever algorithm we use as the base classifier.
– Trees do not perform well when it comes to terms of many dimensions. However, the classification of Multi-label Hoeffding Tree shows improvements if we filter the training examples to contain the most important labels.
– The best performance among all the models was observed by the Binary Relevance Updateable with SGD (Hinge Loss) for Example-based and Label-based evaluation metrics and for the Ranking-based evaluation metrics, Binary Relevance Updateable with SGD (Logistic Regression).

References

1. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: On demand classification of data streams. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2004, pp. 503–508. ACM, New York (2004). http://doi.acm.org/10.1145/1014052.1014110
2. Bifet, A., Holmes, G., Pfahringer, B., Kranen, P., Kremer, H., Jansen, T., Seidl, T.: Moa: massive online analysis, a framework for stream classification and clustering. In: Invited Presentation at the International Workshop on Handling Concept Drift in Adaptive Information Systems in Conjunction with European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2010), pp. 3–16 (2010)
3. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2000, pp. 71–80. ACM, New York (2000). http://doi.acm.org/10.1145/347090.347107
4. Kong, X., Yu, P.S.: An ensemble-based approach to fast classification of multi-label data streams. In: Georgakopoulos, D., Joshi, J.B.D. (eds.) 7th International Conference on Collaborative Computing: Networking, Applications and Worksharing, CollaborateCom 2011, Orlando, FL, USA, pp. 95–104. ICST/IEEE, 15-18 October 2011. http://dx.doi.org/10.4108/icst.collaboratecom.2011.247086
5. Morales, G.D.F., Bifet, A.: Samoa: scalable advanced massive online analysis. J. Mach. Learn. Res. **16**, 149–153 (2015). http://jmlr.org/papers/v16/morales15a.html
6. Neumeyer, L., Robbins, B., Nair, A., Kesari, A.: S4: distributed stream computing platform. In: Proceedings of the 2010 IEEE International Conference on Data Mining Workshops, ICDMW 2010, pp. 170–177. IEEE Computer Society, Washington, DC (2010). http://dx.doi.org/10.1109/ICDMW.2010.172
7. Read, J., Bifet, A., Holmes, G., Pfahringer, B.: Scalable and efficient multi-label classification for evolving data streams. Mach. Learn. **88**(1–2), 243–272 (2012). http://dx.doi.org/10.1007/s10994-012-5279-6
8. Read, J., Reutemann, P., Pfahringer, B., Holmes, G.: MEKA: a multi-label/multi-target extension to Weka. J. Mach. Learn. Res. **17**(21), 1–5 (2016). http://jmlr.org/papers/v17/12-164.html
9. Shi, Z., Xue, Y., Wen, Y., Cai, G.: Efficient class incremental learning for multi-label classification of evolving data streams. In: 2014 International Joint Conference on Neural Networks, IJCNN 2014, Beijing, China, pp. 2093–2099. IEEE, 6-11 July 2014. http://dx.doi.org/10.1109/IJCNN.2014.6889926
10. Tsoumakas, G., Vlahavas, I.: Random *k*-labelsets: an ensemble method for multilabel classification. In: Kok, J.N., Koronacki, J., Mantaras, R.L., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 406–417. Springer, Heidelberg (2007). doi:10.1007/978-3-540-74958-5_38
11. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. In: Maimon, O., Rokach, L. (eds.) Data Mining and Knowledge Discovery Handbook, 2nd edn, pp. 667–685. Springer, Heidelberg (2010). Chap. 34
12. Tsoumakas, G., et al.: WISE 2014 challenge: multi-label classification of print media articles to topics. In: Benatallah, B., Bestavros, A., Manolopoulos, Y., Vakali, A., Zhang, Y. (eds.) Web Information Systems Engineering - WISE 2014. LNCS, vol. 8787, pp. 541–548. Springer, Heidelberg (2014)

13. Tsoumakas, G., Spyromitros-Xioufis, E., Vilcek, J., Vlahavas, I.: Mulan: a java library for multi-label learning. J. Mach. Learn. Res. (JMLR) **12**, 2411–2414 (2011)
14. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2003, pp. 226–235. ACM, New York (2003). http://doi.acm.org/10.1145/956750.956778
15. Wang, P., Zhang, P., Guo, L.: Mining multi-label data streams using ensemble-based active learning. In: SDM, pp. 1131–1140 (2012)
16. Xioufis, E.S., Spiliopoulou, M., Tsoumakas, G., Vlahavas, I.P.: Dealing with concept drift and class imbalance in multi-label stream classification. In: Walsh, T. (ed.) IJCAI, pp. 1583–1588. IJCAI/AAAI (2011)
17. Zhang, M.L., Zhou, Z.H.: ML-KNN: a lazy learning approach to multi-label learning. Pattern Recogn. **40**(7), 2038–2048 (2007)
18. Zliobaite, I., Bifet, A., Read, J., Pfahringer, B., Holmes, G.: Evaluation methods and decision theory for classification of streaming data with temporal dependence. Mach. Learn. **98**(3), 455–482 (2015). http://dx.doi.org/10.1007/s10994-014-5441-4

Extended Formulations for Online Action Selection on Big Action Sets

Shaona Ghosh[1(✉)] and Adam Prügel-Bennett[2]

[1] University of Oxford, Oxford, UK
shaona.ghosh@oerc.ox.ac.uk
[2] University of Southampton, Southampton, UK
apb@ecs.soton.ac.uk

Abstract. There are big data applications where there is an abundance of latent structure in the data. The online action selection learning algorithms in the literature use an exponential weighting for action selection. However, such strategies are provably suboptimal or computationally inefficient or both. The complexity of addressing such action selection problem is attributed to the combinatorial structure of the action set $\mathscr{A} = n^d$, where n is the number of instances and d is the dimensionality of the problem. Here, we develop an online algorithm for structured big data by adapting striking techniques from discrete optimization and approximation algorithms called 'extended formulations'. Such formulations appeal to the underlying geometry of the set with efficient exploration of feasible actions with a guaranteed logarithmic dependence on the dimensionality. An empirical evaluation over simulated and real dataset show our method outperforms the state-of-the-art online algorithms over combinatorial action sets.

Keywords: Online learning · Bandits · Combinatorial action set

1 Introduction

Consider an example of large scale wireless network. The data obtained from such a network has both intrinsic structure (paths far apart has similar network traffic density) or extrinsic structure (paths locally connects to each other). Such data are often represented as a combinatorial graph, where every path (or set of edges) denote an action in an action set; an action being a route from the source to the destination. Now consider the problem of sequentially selecting a route for transferring packets of information over the wireless network, that is optimal at every time step. If the number of edges in the network is given by m, the search space for selecting a path is given 2^m, where every edge might or might not be present on the path selected. The space is exponential in the size of the problem; which is overwhelming in the case of big data from multiple sensors and streaming data cloud servers. Thus it is important to design sequential algorithms that can adaptively select optimal path at every time step.

Typically, online learning techniques in machine learning address the sequential and continuous decision making or action selection problem. Online linear optimization is a natural generalization of the basic adversarial (non-stochastic) or worst case multi-arm bandit framework [4] to the domain of convex optimization, where the set of actions is replaced by a compact action set $\mathscr{A} \subset \mathbb{R}^d$ and the loss is a linear function on the action set $\mathscr{A}$. Linearity is a fair assumption as in certain application like network routing problem, the total cost of selecting a path (action) is linear in the cost of the edges that form the path [7].

Despite being a compelling framework, the problems become challenging to address when the forecaster's decision set is the set of all possible overlapping actions, thus having a combinatorial structure; with added complexity for big data with large n and large dimensionality d. It is well known that in the bandit setting, efficient tradeoff between exploitation and exploration is required for better prediction [4]. Efficient exploration of very large action set becomes impossible with large scale data. Online linear optimization techniques prove good in such problems when the number of actions N is exponential in the natural parameters of the problem. However, the existing methods do not guarantee a logarithmic dependency on the dimension. We address this problem by using a notion from convex optimization that simplifies the complexity of the big action set problem by computing an efficient approximate representation of the set, that is used in exploration within an online learning algorithm. Particularly, we use a technique called 'extended formulation'- a lift and projection technique where the complicated combinatorial convex set (action set) is approximated by a simpler convex set whose linear image is the original convex set having natural barrier penalty [11,16,17]. Our technique outperforms standard approaches over simulated and real combinatorial action set data.

1.1 Literature Review

Dani et al. [13] showed that optimal regret bounds of the order of $\mathscr{O}(\sqrt{n})$ (where n is the number of rounds) was first obtained by using a variant of the original adversarial bandit Exp3 [3] strategy. Their exploration strategy was later refined by Bianchi et al. [10], for combinatorial symmetric action sets $|\mathscr{A}|$ by using a uniform distribution over the actions. In both Dani et al. [13] and Bianchi et al. [10], the forecaster strategy is based on the probability distribution as in Exp3 by Auer [3]. However, for a set of k arms, Exp3 scales with $\sqrt{nN \ln n}$, and for discretised k into d as in the combinatorial setting, N possible actions is exponential in d. This work was completed by Bubeck et al. [8], where an optimal exploration using John's theorem [6] from convex geometry is used on an adaptation of Exp3 called Exp2, to obtain optimal regret bound of $\sqrt{dn \log \mathscr{A}}$ for any set of finite actions. Simultaneously, there are other results in the same direction of research using gradient descent, mirror descent and perturbation approaches [7]. Using mirror descent for online linear optimization with bandit feedback on a computationally efficient strategy was provided by [1] using self concordant barriers. Their result is improved by Bubeck [8] using Exp2 strategy to attain $\mathscr{O}(d\sqrt{n})$. However, Audibert [2] proved that Exp2 is a provably suboptimal strategy in the combinatorial setting. The optimal regret bound is also obtained by the mirror descent strategies on the simplex and the Euclidean ball action sets. OSMD for the Euclidean ball in Bubeck [6,8], achieves regret of the order $\sqrt{dn}$. For more information on combinatorial linear optimization, interested reader may refer [7]. Cesa Bianchi et al. have established in [9], the unified analysis of mirror descent and fixed share of weights between experts in the online setting. Although, they have showed a logarithmic dependence on the dimension for the generalized case, our work is different in that we capture the loss estimation and propose that in cases, the dependency is logarithmic on r where $r \ll d$; by providing an empirical analysis of our work.

2 Extended Formulations in Linear Bandits

2.1 Extended Formulations

An extended formulation [12] is a way of representing the feasible set of solutions that is naturally exponential in the size of the data, to a formulation polynomial in the natural parameters of the problem by the introduction of a polynomial number of new variables. Where possible, the extended formulation or its approximation, simplifies the computing complexity associated with the problem in the linear programming paradigm. Yannakakis's [18] seminal work naturally characterizes the size of the extended formulation by giving the smallest size of the extension that represents the original set well. The recent work of Gouveia et al. [16] and Fiorini et al. [14] generalizes the theorem from linear programming paradigm for convex optimization. The techniques are also known as lift-and-project [5] and approximation techniques [11]. Our work employs this striking technique to the online linear optimization setting to exploit the underlying geometry of the big action set that is exponential in d in the hope of recovering some structure in the combinatorial action set. In the illustration shown in Fig. 1, P is the original compact set and Q is the extension, such that $P \subseteq Q$.

2.2 Problem Setup

In the formal setting of the combinatorial online bandit optimization problem, we consider an adversarial environment with limited feedback. The prediction game as described in the previous sections proceeds in a series of rounds $t = 1, \dots, T$. The action set forms a subset of a hypercube $\mathscr{A} \subset [0,1]^d$ that comprises set of all possible actions, where d is the dimension of the hypercube. When the feasible solutions of a combinatorial optimization problem are encoded as 0/1-points in $\mathbb{R}^d$, they yield a convex hull polytope of the resulting points. The loss is assumed to be non-negative $\mathscr{L} = [0,1]^d$. At every round t, the forecaster chooses an action $a_t \in \mathscr{A}$; the action is represented by its incidence vector as corners of the hypercube. The cardinality of all possible actions is denoted by N. The adversary secretly selects the loss that is incurred by the forecaster and is defined as $a_t^T l_t$. The forecaster's strategy at every round is to choose a probability distribution $p_{t-1}(1), \ldots, p_{t-1}(N)$ over the set of actions such that $p_{t-1}(k) \geq 0$ for all $k = 1, \ldots, N$. Note that $\sum_{k=1}^{N} p_{t-1}(k) = 1$, and action $a_t = k$ is drawn with probability $p_{t-1}(k)$. The objective of the forecaster is to minimize the pseudo regret defined in Eq. 1. The expectation in 1 is with respect to the forecaster's internal randomization and possible adversarial randomization. Note that this problem formulation is identical to the adversarial bandit framework when $d = N$ and the action selected at each round a_t forms the canonical basis.

$$\overline{R_T} = \mathbb{E}\sum_{t=1}^{T} a_t^T z_t - \min_{a \in \mathscr{A}} \mathbb{E}\sum_{t=1}^{T} a^T z_t. \tag{1}$$

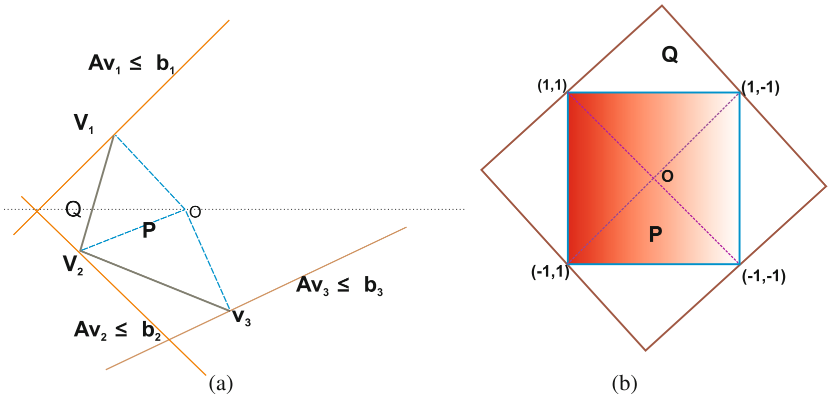


Fig. 1. Illustration of extended formulation of convex compact set P (a) Q is the lifted or extended set, both P and Q are in the non-negative orthant cone. Vertices are denoted by v_i, b_i's are the distances of the hyperplanes from the origin (b) Toy example of the infinite polytope P and its extended formulation Q. Slack measures the distance from the origin O to the vertices.

2.3 Slack Based Regularization

Definition 1. Let $P = conv\left\{a | a \in \mathscr{A} \subset \{\mathbb{R}\}^d\right\}$ be the convex hull of the action set with a non-empty interior, then for every loss estimate sampled randomly from the simplex $l \in \{0,1\}^d$ by the adversary, there is an extended set $Q = \{x | Ax \leq l\}$, where x is the action played, A is the action set, l is the loss observed. Both P and Q are defined on the positive orthant.

The above definition implies that, if P is the convex hull of all actions defined on the positive quadrant, let there be a set of hyperplanes given by the loss observed at every round; then Q defines a set of inequalities that define on which side of the hyperplanes defined by the losses, the vertices lie.

Definition 2. The slack regularity for all action $a \in \mathscr{A}$, is the non-negative measure of how far the action is from the origin of the set P. It is the difference of the distance of the hyperplane from the origin defined by the loss observed l proportional to the distance of the action from the last action played:$\left|a_t^T l_t - a_t^T a\right|$, where $a_t^T l_t$ defines the hyperplane for round t, a_t is the action played.

The slack regularity measure is the linear projection between P and Q, and defines a much simpler slack set called the slack matrix. The idea is the original complex set P is extended to Q by introducing inequalities and slack variables - the hyperplanes, then a linear projection technique is used to project the lift back. The linear projection defines the slack matrix.

Definition 3. The non-negative slack matrix defined by the slack regularity measure is the measure of how much any particular action is breaking the inequality or is far from the hyperplane proportional to the action played and is given by $M_{t,i} = \left|a_t^T l_t - a_t^T a_i\right|$ for round t is a $(t+1) \times N$ matrix

3 Algorithms and Results

In Algorithm 1, an exponentially weighted technique is used to weigh the actions based on the slack regularity measure defined in Definition 2. The actions are weighted based on the absolute difference in their slack regularity between the loss suffered and the last action played in the update step. The weights thus updated define the weight matrix or the slack matrix M_{t+1} at each round that has a positive rank. The slack matrix of weights has a minimum non-negative rank r that induces the probability distribution p_t in the next trial.

Definition 4. As long there is a positive rank for the weights matrix M_t, there is a guaranteed non-negative factorization possible such that $M = RU$, where R and U are non-negative factors. In general, there is a minimum k, such that $M_{i,j} = \sum_k R_{i,k} U_{k,j}$

The Fig. 2 illustrates the non-negative factorization of the slack matrix.

$$M = \begin{bmatrix} 0 & 0 & 1 & 2 & 2 & 1 \\ 1 & 0 & 0 & 1 & 2 & 2 \\ 2 & 1 & 0 & 0 & 1 & 2 \\ 2 & 2 & 1 & 0 & 0 & 1 \\ 1 & 2 & 2 & 1 & 0 & 0 \\ 0 & 1 & 2 & 2 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 2 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 1 & 2 & 1 \\ 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Fig. 2. Slack matrix factorization. Slack matrix for the regular hexagon M factored into non-negative matrices. Example taken from [16].

3.1 Complexity of Extension

It turns out that the non-negative rank of the slack matrix M is the extension complexity of the polytope (hypercube in our case) of actions P. In reality, the non negative rank of a slack matrix S of the order $\log_2 d$ is the extension complexity for dimensionality d. The rank of the slack matrix provides lower bounds to the complexity of the extension. The lower bound on the non-negative rank for a regular n-gon that has a $\mathbb{R}^r_+$ lift is given by $r = \mathscr{O}(\log_2(d))$ [16]. Similarly, the minimum r-lift representation of the combinatorial d-dimensional hypercube is bounded by $2d$.

4 Experimental Evalutation

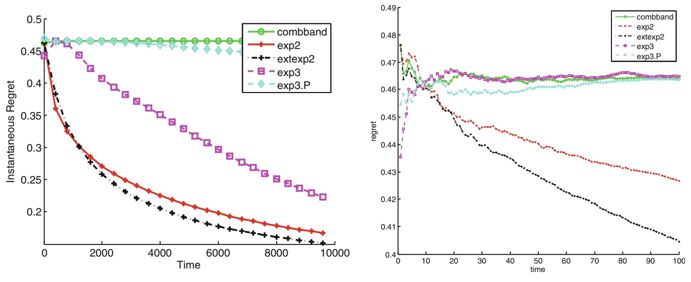
We compared the extended exponential weighted approach with the state-of-the-art exponential weighted algorithms in the adversarial linear optimization bandit setting.

Algorithm 1. Extended Exp

```
Input: learning rate $\eta \geq 0$, mixing coefficient $\alpha \geq 0$, action set dimensional rank $\delta = \rho(\mathscr{A})$.
repeat
  Initialize $p_1 = \left(\frac{1}{\delta}, \ldots, \frac{1}{\delta}\right) \in \mathbb{R}^{|\mathbb{A}|}$. Let non-negative rank $r = \delta$
  for $t = 1$ to $T$ do
    $p_t = \frac{\alpha}{r} + (1-\alpha) w_{i,t}$
    Play action $a_t$ from $p_t$
    Observe loss $a_t^T l_t$
    Update loss $w_{i,t+1} = \frac{p_{i,t} e^{-\eta \left| a_t^T l_t - a_t^T a_{i,t} \right|}}{\sum_{j=1}^{N} p_{j,t} e^{-\eta \left| a_t^T l_t - a_t^T a_{j,t} \right|}}$
    $M_{t+1} = [w_{t,i}]_{1 \leq t \leq t+1, 1 \leq i \leq N}$
    Find minimum non-negative rank $r$ such that $M = \sum_{k=1}^{r} R^k U_k$
  end for
until Time horizon T or no regret
```

4.1 Simulations

In the first experiment, among a d-dimensional network of routes, the optimal route should be selected by the learning algorithm. Typically, we choose d to vary between 10 and 15. The environment is an oblivious adversary to the player's actions, simulated to choose fixed but unknown losses at each round. Losses are bounded and in the range $[0, 1]$. The learning algorithm is executed using our basic Extended Exp algorithm. Each action is represented a d-dimension incidence vector, with 1 indicating if an edge or path is present in the route or 0 otherwise. Figure 3 displays the results of the cases where $d = 10, 15$ and $d = 20$. The performance measure is the instantaneous regret over time; we use psuedo



(a) (b)

Fig. 3. Simulation results with the network dataset with 10 dimensions. Results averaged over 100 runs. Extended Exp2 (BLACK) beats the baseline Exp2 (RED) (a) Network dataset with 10 dimensions (b) Network dataset with 15 dimensions.

regret here. The number of iterations in the game are 10000 for $d = 10$ and 100 for $d = 15$. In both cases, the results are averaged over 100 runs of the games. We implemented Algorithm 1, Exp2 [7], Exp3 [4], Exp3.P [4], and CombBand [10]. Our baseline is Exp2 which has the best known performance but provably suboptimal. We could not compare with Exp2 with John's exploration [8] as the authors state its computational inefficiency. All the experiments are implemented using Matlab on a notebook with i7-2820QM CPU 2.30 GHz with 8 GB RAM. We see that Extended Exp 2 clearly beats the baseline comfortably against the oblivious adversary. We repeated the experiments with different configurations of the network and different dimensions. Each time, the complexity of the problem increases exponentially with the dimension. Extended Exp2 performs best in all our experiments, the results of the other trials are excluded for brevity.

4.2 Empirical Datasets

The dataset is the Jester Online Joke Recommendation dataset [15] from the University of Berkeley, which is data collected from 24,983 users with ratings on 36 or more jokes. We consider the ratings of 24,983 users on 20 jokes, constituting the dense matrix. The ratings vary in the range $[-10.00, 10.00]$, including not rated jokes. We scale and normalize the ratings in the range $[0, 1]$. For the purpose of our problem, each user represents a d dimensional decision problem, where $d = 20$. We do not make the ratings available to the algorithm, instead the ratings are provided by the environment based on the user. In other words, we assume the non-oblivious setting for the adversary, the loss changes with the user or action selected. The goal of the algorithm is to be able to identify the user who has rated the worst on all the jokes on average. As before we execute all the instances of the common exponential weighted algorithms and the basic version of Extended Exp2. Figure 4 displays the result on the dataset, all the plots are averaged over 100 runs of the games. We run each game for 10000 rounds. All the experiments are implemented on Matlab on a notebook with i7-2820QM

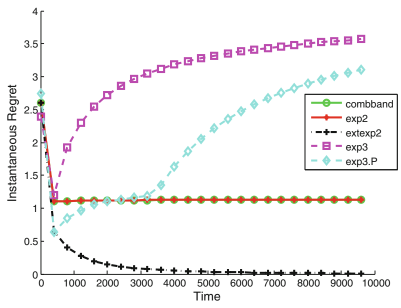


Fig. 4. Dataset results with the Jester Online Joke Recommendation dataset with 20 dimensions. Results averaged over 100 runs. Extended Exp2 (BLACK) beats the baseline Exp2 (RED).

CPU 2.30 GHz with 8 GB RAM. We observe that once again Extended Exp2 beats all the others. Quite surprisingly, Exp2 and Combband seem to perform equivalently in the non-oblivious setting. It will be interesting to compare with mirror descent based linear optimization, that is for future work.

5 Conclusion and Future Work

We have proposed a novel slack based regularization approach to the online linear optimization with bandit information in the exponential weighted setting. Namely, we develop an online algorithm that is capable of predicting sequential actions from a set of actions that varies exponentially to the dimension of the data. In case of Big Data, the complexity scales combinatorially with the size of the data set and the dimensionality making the existing online approaches intractable or computationally expensive. We propose a logarithmic dependence on the dimension complexity of the problem by exploiting the notion of extended formulations and the non negative factorization of the associated slack matrix. We propose that in the exponential weighted setting, logarithmic dependence is achievable by a clever regularization, that measures how far an action in the set is, given the loss and the other actions. This work can be generalized into a all such compact action sets where efficient exploration is non-trivial. As future work, we would like to derive the lower bounds and attempt to provide an analysis of our method. We would also like to investigate how the extended formulation characterizes different action sets. Further, it would be interesting to derive an information theoretic entropic analysis of our method.

References

1. Abernethy, J., Hazan, E., Rakhlin, A.: Competing in the dark: an efficient algorithm for bandit linear optimization. In: Proceedings of the 21st Annual Conference on Learning Theory (COLT), vol. 3, p. 3 (2008)

2. Audibert, J-Y., Bubeck, S., Lugosi, G.: Minimax policies for combinatorial prediction games. arXiv preprint arXiv:1105.4871 (2011)
3. Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.E.: Gambling in a rigged casino: the adversarial multi-armed bandit problem. In: Proceedings of the 36th Annual Symposium on Foundations of Computer Science, 1995, pp. 322–331. IEEE (1995)
4. Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.E.: The nonstochastic multi-armed bandit problem. SIAM J. Comput. **32**(1), 48–77 (2002)
5. Balas, E., Ceria, S., Cornuéjols, G.: A lift-and-project cutting plane algorithm for mixed 0–1 programs. Math. Program. **58**(1–3), 295–324 (1993)
6. Ball, K.: An elementary introduction to modern convex geometry. Flavors Geom. **13**, 13 (1997)
7. Bubeck, S., Cesa-Bianchi, N.: Regret analysis of stochastic and nonstochastic multi-armed bandit problems. Found. Trends Mach. Learn. **5**(1), 64–87 (2012)
8. Bubeck, S., Cesa-Bianchi, N., Kakade, S.M.: Towards minimax policies for online linear optimization with bandit feedback. In: JMLR Workshop and Conference Proceedings Volume 23: COLT 2012 (2012)
9. Cesa-Bianchi, N., Gaillard, P., Lugosi, G., Stoltz, G.: Mirror descent meets fixed share (and feels no regret). arXiv preprint arXiv:1202.3323 (2012)
10. Cesa-Bianchi, N., Lugosi, G.: Combinatorial bandits. J. Comput. Syst. Sci. **78**(5), 1404–1422 (2012)
11. Chandrasekaran, V., Jordan, M.I.: Computational and statistical tradeoffs via convex relaxation. In: Proceedings of the National Academy of Sciences (2013)
12. Conforti, M., Cornuéjols, G., Zambelli, G.: Extended formulations in combinatorial optimization. 4OR **8**(1), 1–48 (2010)
13. Dani, V., Hayes, T., Kakade, S.M.: The price of bandit information for online optimization. In: Advances in Neural Information Processing Systems, vol. 20, pp. 345–352 (2008)
14. Fiorini, S., Massar, S., Pokutta, S., Tiwary, HR., de Wolf, R.: Linear vs. semidefinite extended formulations: exponential separation and strong lower bounds. In: Proceedings of the 44th Symposium on Theory of Computing, pp. 95–106. ACM (2012)
15. Goldberg, K.: Anonymous Ratings from the Jester Online Joke Recommender System (2003). Accessed 03 Oct 2013
16. Gouveia, J., Parrilo, P.A., Thomas, R.: Lifts of convex sets and cone factorizations. Math. Oper. Res. **38**, 248–264 (2011)
17. Nesterov, Y., Nemirovskii, A.S., Ye, Y.: Interior-Point Polynomial Algorithms in Convex Programming, vol. 13. SIAM, Philadelphia (1994)
18. Yannakakis, M.: Expressing combinatorial optimization problems by linear programs. J. Comput. Syst. Sci. **43**(3), 441–466 (1991)

A-BIRCH: Automatic Threshold Estimation for the BIRCH Clustering Algorithm

Boris Lorbeer(✉), Ana Kosareva(✉), Bersant Deva, Dženan Softić, Peter Ruppel, and Axel Küpper

Service-centric Networking, Telekom Innovation Laboratories, Technische Universität Berlin, Berlin, Germany
{lorbeer,ana.kosareva,bersant.deva,peter.ruppel, axel.kuepper}@tu-berlin.de, softic.dzenan@gmail.com

Abstract. Clustering algorithms are recently regaining attention with the availability of large datasets and the rise of parallelized computing architectures. However, most clustering algorithms do not scale well with increasing dataset sizes and require proper parametrization for correct results. In this paper we present A-BIRCH, an approach for automatic threshold estimation for the BIRCH clustering algorithm using Gap Statistic. This approach renders the global clustering step of BIRCH unnecessary and does not require knowledge on the expected number of clusters beforehand. This is achieved by analyzing a small representative subset of the data to extract attributes such as the cluster radius and the minimal cluster distance. These attributes are then used to compute a threshold that results, with high probability, in the correct clustering of elements. For the analysis of the representative subset we parallelized Gap Statistic to improve performance and ensure scalability.

1 Introduction

Clustering is an unsupervised learning method that groups a set of given data points into well separated subsets. Two prominent examples of clustering algorithms are k-means [9] and the EM algorithm [4]. This paper addresses two issues with clustering: (1) clustering algorithms usually do not scale well and (2) most algorithms require the number of clusters (cluster count) as input. The first issue is becoming more and more important. For applications that need to cluster, e.g., millions of documents, huge image or video databases, or terabytes of IoT sensor data, scalability is essential. The second issue severely reduces the applicability of clustering in situations where the cluster count is very difficult to predict, such as data exploration, feature engineering, and document clustering.

An important clustering method is BIRCH [17], which is one of the fastest clustering algorithms available. It outperforms most of the other clustering algorithms by up to two orders of magnitude. Thus, BIRCH already solves the first issue mentioned above. However, to achieve sufficient clustering quality, BIRCH requires the cluster count as input, therefore failing to solve the second issue. This paper describes a method to use BIRCH without having to provide the cluster count, yet preserving cluster quality and speed. We achieve this by omitting the global clustering step and carefully choosing the *threshold* parameter of BIRCH. Following an idea by Bach and Jordan [7], we propose to learn this parameter from representative data. Our approach aims at datasets drawn from two-dimensional isotropic Gaussian distributions which are typical when dealing with, for example, geospatial data.

2 Related Work

Clustering algorithms usually do not scale well, because often they have a complexity of $O(N^2)$ or $O(NM)$, where N is the number of data points and M is the cluster count. Scalability is typically achieved by parallelization of the algorithm in compute clusters, e.g., Mahout's k-means clustering [11] or Spark's distributed versions of k-means, EM clustering, and power iteration [10]. Other parallelization attempts use the GPU. This has been done for k-means [16], EM clustering [8], and others. The bottleneck here is the relatively slow connection between host and device memory if the data does not fit into device memory.

The second issue we are concerned with is the identification of the cluster count. A standard approach is to use one of the clustering algorithms that require the cluster count to be input as a parameter, then run it for each count k inside an interval of likely values. Then, the "elbow method" [14] is used to determine the optimal number k. For probabilistic models, one can apply information criteria like AIC [1] or BIC [13] to rate the different clustering results, see, for example, [18]. But all those methods increase the computation time considerably, especially if there is not enough prior information to keep the range of possible cluster counts small. Some clustering algorithms find the number of clusters directly, without being required to run the algorithm for all possible counts. Two of the more well-known examples are DBSCAN [5] and Gap Statistic [15]. There are also some attempts to improve the clustering quality of BIRCH by changing the algorithm itself, e.g. with non-constant thresholds [6], with two different global thresholds [2], or by using DBSCAN on each CF level to reduce noise [3]. However, while sometimes improving the quality, those approaches slow BIRCH down and still require the cluster count as input.

3 BIRCH

We shortly describe BIRCH, mainly to fix notations. For details, see [17]. BIRCH requires three parameters: the branching factor Br, the threshold T, and the cluster count k. While the data points are entered into BIRCH, a height-balanced tree, the *CF tree*, of hierarchical clusters is built. Each node contains the most important information of the belonging cluster, the *cluster features* (CF). From those, the cluster center $C = 1/n \sum_i^n x_i$, where $\{x_i\}_{i=1}^n$ are the elements of the cluster, and cluster radius $R = \sqrt{1/n \sum_i^n x_i^2}$ can be computed for each cluster. Every new point starts at the root and recursively walks down the tree entering the subcluster with the nearest center.

When adding a point at the leaves, a new cluster is created if the cluster radius R increases beyond the threshold T, otherwise the point is added to the nearest cluster. If the creation of a new cluster leads to more than Br child nodes of the parent, the parent is split. To ensure that the tree stays balanced, the nodes above might need to be split recursively. Once all points are submitted to BIRCH, the centers of the leaf clusters are, in the *global clustering* phase, entered into k-means with the cluster count k. This last step improves the cluster quality by merging neighboring clusters. In this paper, we will refer to the BIRCH algorithm as *full-BIRCH* and to BIRCH without its global clustering phase as *tree-BIRCH.*

Tree-BIRCH is very fast. It clusters 100,000 points into 1000 clusters in 4 s, on a 2,9 GHz Intel Core i7, using scikit-learn [12]. The k-means implementation of the same library needs over two minutes to complete the same task on the same architecture. Furthermore, tree-BIRCH doesn't require the cluster count as input, which in full-BIRCH is only needed for the global clustering phase. However, tree-BIRCH usually suffers from bad clustering quality. Therefore, this paper focuses on improving the clustering quality of tree-BIRCH.

4 Concept

In the following, we present a method that automatically chooses an optimal threshold parameter for tree-BIRCH. First, note that the CF-tree depends on the order in which the data is entered. If the points of a single cluster are entered in the order of increasing distance from the center, tree-BIRCH is more likely to return just one cluster than if the first two points are from opposite sides of the cluster. In the latter case, the single cluster is likely to split, a situation we will refer to as *cluster splitting* (Fig. 1A). Next, consider two neighboring clusters. If the first two points are from opposite clusters but still near each other, they could be collected into the same cluster, given the threshold is large enough, which we refer to as *cluster combining* (Fig. 1B). *Cluster combining* often co-occurs with *cluster splitting.* To reduce *splitting* of a single cluster, the threshold parameter of tree-BIRCH has to be increased, whereas a decreased threshold parameter reduces *cluster combining.* Datasets with a large ratio of cluster distance (the distance between the cluster centers) to cluster radius are less likely to produce such errors (Fig. 1C). In the next section, we derive a condition for the error probability to be less than one percent. Also, a formula for an appropriate threshold is provided. Note that there is, in fact, another source of splitting. This happens if a cluster overlaps with two regions belonging to two non-leaf nodes in the CF-tree. Therefore, we choose the branching factor Br to be larger than the highest possible cluster count.

5 Automatic Estimation of the BIRCH Threshold

In this paper it is presumed that the clusters are samples from two-dimensional isotropic Gaussian distributions of roughly the same variance. To find conditions for tree-BIRCH to work well, we first need to determine the common radius R of

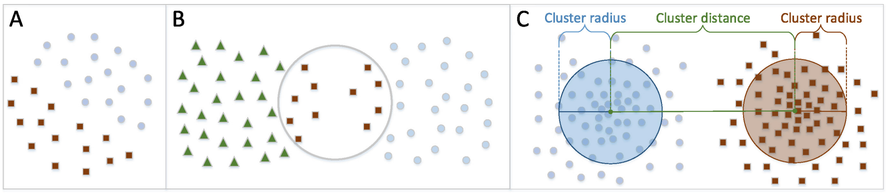


Fig. 1. *Cluster splitting* (A), *cluster combining* where the combining cluster is circled (B), depiction of cluster radius and cluster distance (C). Different forms and colors of the observations correspond to different clusters they belong to.

the clusters and the minimum cluster distance D_{min}. It is presumed that there exists a small but representative subset of the data that has the same cluster radius and minimum cluster distance D_{min} as the full dataset. On this small dataset, Gap Statistic is applied to obtain the cluster count k. This k in turn is given to k-means to produce a clustering of the subset, which finally yields the two values R and D_{min}.

For the determination of R and D_{min} one could also use any other clustering algorithm that finds the cluster centers and radii without requiring the cluster count k as an input. However, Gap Statistic is chosen here, due to its high precision. Our approach is heuristic. In each case, tree-BIRCH is run often enough to deduce estimates of *cluster splitting* and *combining* probabilities with sufficiently small 95 % confidence intervals (with 10,000 repetitions of each tree-BIRCH clustering, the confidence interval of our error estimate at 0.01 is roughly 0.01 ± 0.002).

Avoiding Cluster Splitting

We create many clusters containing the same number of elements n by sampling from a single isotropic two dimensional Gaussian probability density function. The units are chosen such that the radius R of this cluster will be one. Then, tree-BIRCH is applied with the same threshold T to each of those datasets. After each iteration, we determine whether tree-BIRCH returns the correct number of clusters, namely one. From this, we assess if the error probability estimate for tree-BIRCH is less than 0.01, or one percent. We also investigate the impact of varying the number of elements n in the cluster on the resulting error probability. Finally, we repeat all the above for several thresholds T. For this heuristic analysis, we use the python library scikit-learn and its implementation of BIRCH.

According to the results presented in Fig. 2a, there is no indication that the number of objects in the cluster impacts the error probability. However the error probability is clearly affected by the threshold parameter; the error drops below one percent when the threshold value is greater than or equal to $1.6 \cdot R$.

Avoiding Cluster Combining

We use a mixture of two Gaussians with a cluster distance $D = 6$, both with radius $R = 1$. Again, the error probabilities are not dependent on the total number of data points, as shown in Fig. 3a. This figure pertains only to the cluster

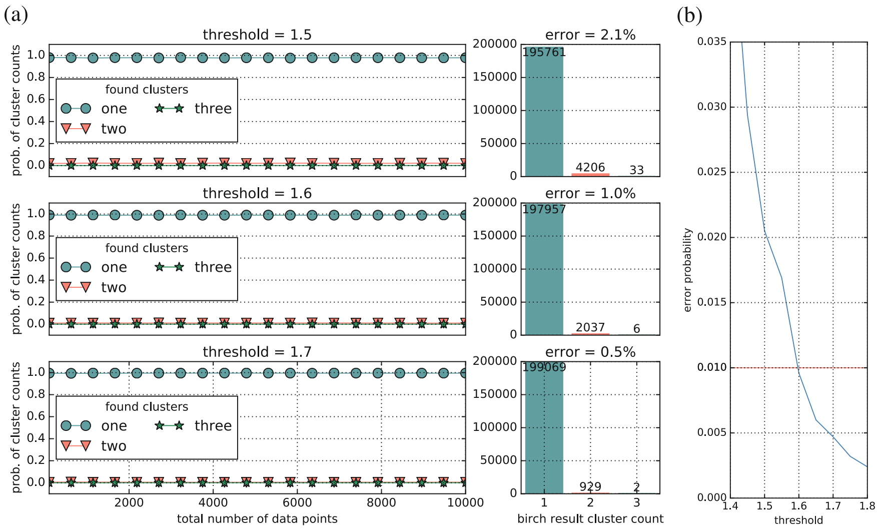


Fig. 2. (a) For each pair (n, T) of total number n of objects, running from 100 to 10,000, and threshold $T \in \{1.5, 1.6, 1.7\}$, we sampled n elements from a Gaussian of radius $R = 1$, and applied tree-BIRCH 10,000 times to compute the probabilities for each of the cluster counts 1, 2, and 3. Every count different from 1 is an error. (b) For each threshold we sample 500 points from a single Gaussian of radius $R = 1$, apply tree-BIRCH and record how often it returns the right number of clusters. This is repeated 10,000 times for each T to obtain an estimate for the error probability.

distance 6.0. To understand the situation for different cluster distances, consider Fig. 3b. Here, we see the dependence of the error probability on the threshold for several different cluster distances. For small thresholds $(T < 1.7)$ we witness *cluster splitting* which results in higher cluster counts and a higher error. This can also be deduced from Fig. 3a, where for the small threshold $T = 1.5$ we see many cluster counts of three and four. With $T = 2.0$ less splitting occurs and the error probability decreases. If the threshold continues growing $(T = 3.0)$, *cluster combining* occurs more frequently, which increases the cluster counts of three and therefore increases the error probability. The fact that the graphs in Fig. 3b are dropping below one percent later than in Fig. 2b is due to *cluster combining*, that was not possible with just one cluster. For $D \geq 6.0$ there are thresholds where the error probability drops below one percent. The minimum is located near 1.9. From this observation it can be inferred, that, if $D_{min} \geq 6.0 \cdot R$, a choice of

$$T = 1.9 \cdot R \tag{1}$$

would ensure that for each pair of neighboring clusters in the dataset, the error probability would be less than one percent.

Of course, if D_{min} is clearly larger than $6.0 \cdot R$, it would be beneficial to increase the threshold beyond (1). While the lower bound on the threshold is nearly the same for all cluster distances, the upper bound increases linearly,

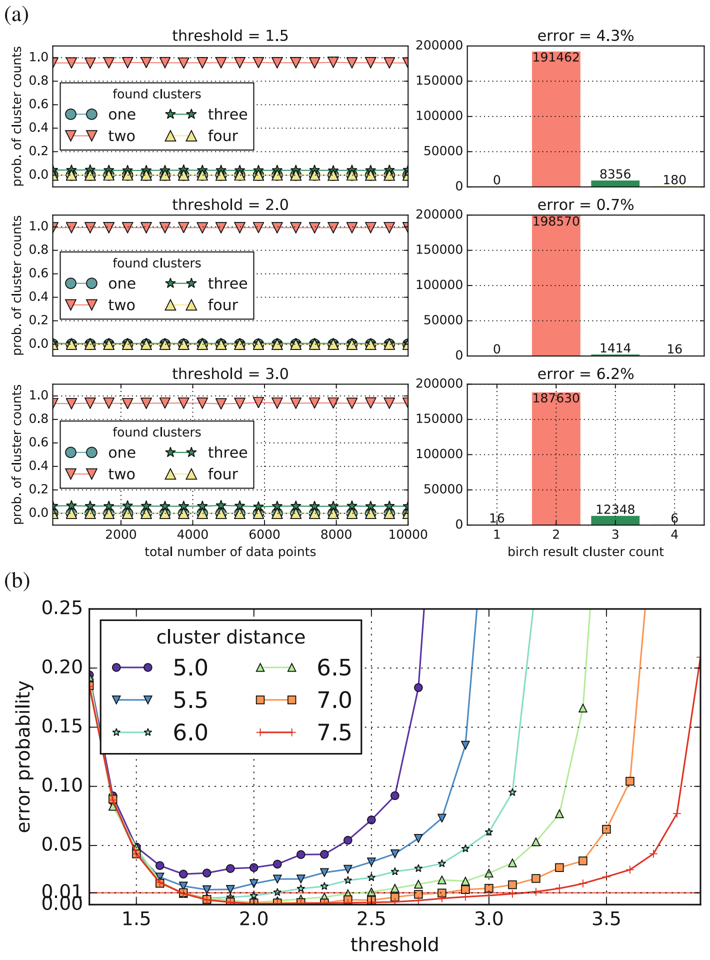


Fig. 3. (a) For each pair (n, T) of total number n of objects, running from 100 to 10,000, and threshold $T \in \{1.5, 2.0, 3.0\}$, we sampled $10,000$ times n elements from a mixture of two Gaussians of distance 6.0, and each time applied tree-BIRCH to compute the probabilities for each of the cluster counts 1, 2, 3, and 4. Every count different from 2 is an error. (b) For each pair (D, T) of cluster distance D and threshold T, we sampled $10,000$ times 500 elements from a mixture of two Gaussians of radius $R = 1$ and distance D. Each time we applied tree-BIRCH with the threshold set to T and computed the error probabilities.

roughly with half the increase of the distance (Fig. 4). This is intuitively clear since two times the threshold should fit comfortably between two clusters to avoid *cluster combining*. To place the threshold in the middle between lower and upper bound, we choose $\frac{1}{4}$ as the ratio of ΔD_{min} and ΔT. We then fit an intercept of 0.7, which yields the following expression for choosing the threshold

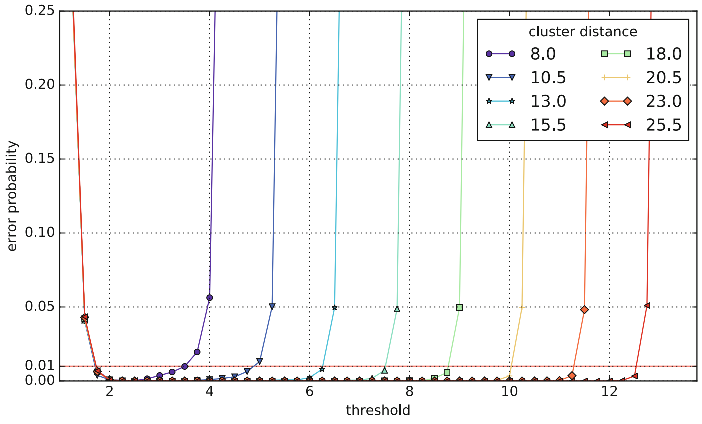


Fig. 4. For each pair (D, T) of cluster distance D and threshold T, we sampled $10,000$ times 500 elements from a mixture of two Gaussians of radius $R = 1$ and distance D. Each time we applied tree-BIRCH to compute the error probabilities.

(in arbitrary units), provided $D_{min} \geq 6.0 \cdot R$:

$$T = \frac{1}{4} D_{min} + 0.7 \cdot R. \tag{2}$$

A-BIRCH with parallel Gap Statistic
We want to build a fast clustering algorithm in order to enable scalability. While tree-BIRCH is very fast, Gap Statistic is not. Therefore, we developed a parallel version of Gap Statistic. Recall that Gap Statistic runs k-means for each cluster count $k \in \{1, \ldots, k_{max}\}$ not only on the dataset itself, but also on many Monte Carlo simulations (the R and MATLAB implementations choose 100 simulations as default value). Therefore, we parallelized the loop over the Monte Carlo reference simulations. The distribution of work and collection of the results are performed by Apache Spark. The proposed approach and the results from Sect. 5 are summarized in Algorithm 1.

6 Evaluation

We evaluated the accuracy of A-BIRCH with the threshold estimation as stated in Eq. (2). Figure 5 shows that A-BIRCH performs correctly with different sizes of D_{min} and different numbers of clusters. The evaluation datasets contain samples from two-dimensional isotropic Gaussian distributions with equal variance and a $D_{min} \geq 6.0 \cdot R$, which fulfills the previously defined requirement.

In an additional step, we evaluated the scalability of A-BIRCH. While tree-BIRCH is considered scalable with an increasing number of elements and clusters, Gap Statistic is the bottleneck as described in 5. We have tested the parallelized implementation of Gap Statistic on an Apache Spark cluster on Microsoft

Algorithm 1. A-BIRCH: Automatic threshold for the BIRCH algorithm

```
Data: $N$ 2-dimensional data points $\{X_i\}$, $k_{max}$, number of Monte Carlo
      simulations $B$, distance metric $D$, branching factor $Br$
Result: CF-tree
begin
    $k^* \leftarrow$ parallel Gap Statistic( subsample($\{X_i\}$), $k_{max}$, $B$ )
    labels $\leftarrow$ k-means( subsample($\{X_i\}$), $k^*$ )
    compute the common radius $R$ and the minimal $D_{min}$ from the clustered
     data
    if $D_{min} < 6 \cdot R$ then
        Warning: the clusters are too close - BIRCH result might be inaccurate
    $T \leftarrow \frac{1}{4} D_{min} + 0.7 \cdot R$
    CF-tree $\leftarrow$ tree-BIRCH( $\{X_i\}$, distance metric $D$, branching factor $Br$, $T$ )
```

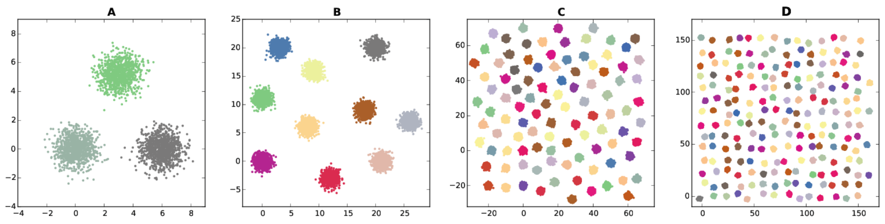


Fig. 5. The datasets A, B, C and D contain 3, 10, 100 and 200 clusters, respectively. Each cluster consists of 1000 elements, the radius of the clusters is $R = 1$, and the D_{min} is in all cases larger than 6: in A - 6.001, in B - 7.225, in C - 6.025, in D - 6.410.

Azure. Two cluster configurations have been evaluated, each with two master nodes and with four and eight workers, respectively, each of which running on `Standard_D3` virtual machines. A `Standard_D3` virtual machine currently provides 4 CPU cores, 14 GB of memory, running the Linux operating system. The parallelization has been implemented using the Spark Python API (PySpark). The computation of Gap Statistic was run on a dataset with 10 clusters, each consisting of 1000 two-dimensional data points. The computation times for varying numbers B of reference datasets and maximal number of clusters k_{max} are shown in Table 1.

Table 1. Speedup of gap statistic by parallelization on Spark

	Sequential	Spark: 4 workers	Spark: 8 workers
$B = 100$, $k_{max} = 20$	1775 s	349 s	197 s
$B = 100$, $k_{max} = 40$	7114 s	1425 s	795 s
$B = 500$, $k_{max} = 20$	8803 s	1470 s	725 s
$B = 500$, $k_{max} = 40$	35242 s	5953 s	2909 s

The results show that the parallelized implementation of Gap Statistic with Spark is scalable as the computation times decrease linearly with an increasing number of worker nodes. Although the Gap Statistic phase is considered computationally expensive, it increases the correctness of BIRCH significantly and does not require any prior knowledge on the dataset.

7 Conclusion

In this paper we introduced A-BIRCH: a parameter-free variant of BIRCH. Choosing the correct parameters for clustering algorithms is often difficult as it requires information about the dataset, which is often not available. This is also true for BIRCH, which requires the cluster count k as well as a threshold T in order to compute the clusters correctly. For this reason, we removed the global clustering phase, thus rendering the cluster count parameter k unnecessary, and proposed a method that automatically estimates the threshold T, which is achieved using Gap Statistic to determine cluster properties. The evaluation proved the applicability of our approach in a very robust manner for two-dimensional isotropic Gaussian distributions with roughly the same variance, regardless of the number of clusters or it's elements. In future work, we plan to use other methods such as DBSCAN to either verify the found cluster properties or to decrease the computational complexity.

References

1. Akaike, H.: Information theory and an extension of the maximum likelihood principle. In: Parzen, E., Tanabe, K., Kitagawa, G. (eds.) Selected Papers of Hirotugu Akaike, pp. 199–213. Springer, New York (1998)
2. Burbeck, K., Nadjm-Tehrani, S.: Adaptive real-time anomaly detection with incremental clustering. Inf. Secur. Tech. Rep. **12**(1), 56–67 (2007)
3. Dash, M., Liu, H., Xu, X.: '$1 + 1 > 2$': Merging distance and density based clustering. In: Proceedings of Seventh International Conference on Database Systems for Advanced Applications, 2001, pp. 32–39 (2001)
4. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. Roy. Stat. Soc. Ser. B (Methodol.) **39**(1), 1–38 (1977)
5. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Simoudis, E., Han, J., Fayyad, U.M. (eds.) Second International Conference on Knowledge Discovery and Data Mining, pp. 226–231. AAAI Press (1996)
6. Ismael, N., Alzaalan, M., Ashour, W.: Improved multi threshold birch clustering algorithm. Int. J. Artif. Intell. Appl. Smart Devices **2**(1), 1–10 (2014)
7. Jordan, M.I., Bach, F.R.: Learning spectral clustering. In: Advances in Neural Information Processing Systems 16. MIT Press (2003)
8. Kumar, N.S.L.P., Satoor, S., Buck, I.: Fast parallel expectation maximization for gaussian mixture models on gpus using cuda. In: 11th IEEE International Conference on High Performance Computing and Communications, pp. 103–109 (2009)

9. Macqueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Math, Statistics, and Probability, vol. 1, pp. 281–297. University of California Press (1967)
10. Meng, X., Bradley, J.K., Yavuz, B., Sparks, E.R., Venkataraman, S., Liu, D., Freeman, J., Tsai, D.B., Amde, M., Owen, S., Xin, D., Xin, R., Franklin, M.J., Zadeh, R., Zaharia, M., Talwalkar, A.: MLlib: machine learning in apache spark. CoRR (2015)
11. Owen, S., Anil, R., Dunning, T., Friedman, E.: Mahout in Action. Manning Publications Co., Shelter Island (2011)
12. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
13. Schwarz, G.: Estimating the dimension of a model. Ann. Stat. **6**(2), 461–464 (1978)
14. Sugar, C.A.: Techniques for Clustering and Classification with Applications to Medical Problems. Ph.D. Dissertation, Department of Statistics, Stanford University (1998)
15. Tibshirani, R., Walther, G., Hastie, T.: Estimating the number of clusters in a data set via the gap statistic. J. Roy. Stat. Soc. B (Stat. Methodol.) **63**(2), 411–423 (2001)
16. Zechner, M., Granitzer, M.: Accelerating k-means on the graphics processor via cuda. In: First International Conference on Intensive Applications and Services, pp. 7–15 (2009)
17. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: a new data clustering algorithm and its applications. Data Min. Knowl. Disc. **1**(2), 141–182 (1997)
18. Zhou, B., Hansen, J.: Unsupervised audio stream segmentation and clustering via the bayesian information criterion. In: Proceedings of ISCLP-2000: International Conference of Spoken Language Processing, pp. 714–717 (2000)

Playlist Generation via Vector Representation of Songs

Burak Köse, Süleyman Eken(✉), and Ahmet Sayar

Computer Engineering Department, Kocaeli University, 41380 Izmit, Turkey
burakks41@gmail.com,
{suleyman.eken,ahmet.sayar}@kocaeli.edu.tr

Abstract. This study proposes a song recommender system. The architecture is based on a distributed scalable big data framework. The recommender system analyzes songs a person listens to most and recommends a list of songs as a playlist. To realize the system, we use Word2vec algorithm by creating vector representations of songs. Word2vec algorithm is adapted to Apache Spark big data framework and run on distributed vector representation of songs to produce a playlist reflecting a person's personal tastes. The performance results are evaluated in terms of hit rates at the end of the paper.

Keywords: Playlist generation · Word2vec · Word embedding · Music recommendation retrieval

1 Introduction

With the help of improvements in the Internet and related technologies, users face an astonishing array of choices regarding listening to songs online or with their portable devices. The Internet enables users to access various remote music resources and to listen to songs online. A music recommender system helps users filter and discover songs according to their tastes. A good music recommender system should be able to automatically detect preferences and generate playlists accordingly. Due to huge data over the internet and not easy to search and filter them according to a person's music taste, a recommender system becomes a necessity day after day, as the data sizes increase. Recommender systems can be seen everywhere in every domain in different formats and shapes [1]. It is not only about music but also everything. We can even use them in our life without even realizing. Amazon and Spotify can be given as good examples of employing recommender systems in their web sites.

It may be harsh and troublesome to elect convenient tracks to listen or to organize them as a playlist. The fundamental concept of a music recommendation system is to reduce users' effort and increase their satisfaction.

Playlist generation is a significant resource area in music recommendation system that allows listening to music based on user feedbacks. There are various approaches to generate playlists automatically. A random approach, given a song or an artist, is based on users' behavior [2]. The most general approach to creating automatic playlists is based on estimating music similarities between pairs of songs, which is established with human experts. Various similarity functions have been suggested such as social tag information [3], the web document mining [4], analysis of audio content [5, 6], or a combination of these approaches as a hybrid system [7].

The remainder of this paper is organized as follows. Section 2 gives relevant works about playlist generation. Section 3 explains the proposed technique for playlist generation using word embedding. Section 4 evaluates the proposed recommender system by performing some tests on real world Cornell playlist dataset. Section 5 concludes the paper and presents the future work.

2 Related Works

An increasingly amount of researches on playlist generation has been conducted on recommender systems. An automatic playlist generation might be a difficult task according to provided seed songs. For the most part, there are two different approaches for playlist generation. One of them considers access to metadata to find related songs while the other one relies on audio similarity [9]. In recommender systems, the context is a significant perspective, in which the generated playlists have to match the audience's demands. For example; we notice some examples of application that aim to produce playlists while driving [10], doing some activities like sports [11], or discovering new songs [12]. Collaborative Filtering is the another useful method in music recommendation to generate playlist [13], but the scalability is a big problem for collaborative filtering, so in most time, the algorithm of alternating least squares is utilized.

Eventually, Chen et al. [8] presents Latent Markov Embedding (LME) for generating playlists. Their model does not require songs description as features a priori, and it can learn a representation from existing playlists. To our knowledge, there is no work creating playlist using Word2vec algorithm and scalable machine learning implementation about it. The following section explains the proposed approach in detail.

3 Architecture of Automated Playlist Generation

In its typical form, playlists are defined to be a list of songs. They can be in sequential or shuffled order. However, in the most time, they are sequential and semantic. In this work, we try to generate playlists based on a song set which can contain only one or more songs. The principal purpose is to find vector representation of each song by Word2vec algorithm. Our approach is based on the idea that if music is a communication language and songs are words, then playlists will be sentences. Therefore, we serialize playlists to sentences and then indexing them from 0 to n, where n is the total number of our song space. Then, we invoke the Word2vec algorithm for finding vector representations of each song. In our work, we use Apache Spark's Word2vec implementation.

Apache Spark is a fast, powerful and exciting open source platform to process big data, which is developed at the University of California, Berkeley's AMPLab, and afterward, donated to Apache Software Foundation. It has lots of usefully supported operations for machine learning, structured query language, streaming, and graph processing.

As we mentioned before, our work is based on Word2vec algorithm which is simply a two-layer neural network and it processes text data. It takes a text corpus as an input data, and then gives a set of vector representations as an output data [14]. After all, if we have set of vector representations of songs, then we can quickly determine a playlist vector using Eq. 1.

$$\mathrm{V(SongSet)} = \frac{1}{\|\mathrm{SongSet}\|}\sum_{(\mathrm{song})\in \mathrm{SongSet}} \mathrm{V(song)} \tag{1}$$

where V is a vector representation of songs. Moreover, we use cosine similarity (Eq. 2) as a measure between two vectors. Here, V_1 and V_2 indicate vectors of songs.

$$\mathrm{similarity} = \cos(\theta) = \frac{V_1 V_2}{\|V_1\|\|V_2\|} \tag{2}$$

As a result, finding vector representation of each song, we are able to find top-N recommendation via cosine similarity. In this approach, due to the usage of vector representations, the model is not limited to take only one start point to generate a playlist. Users can directly specify their music taste with many songs because our

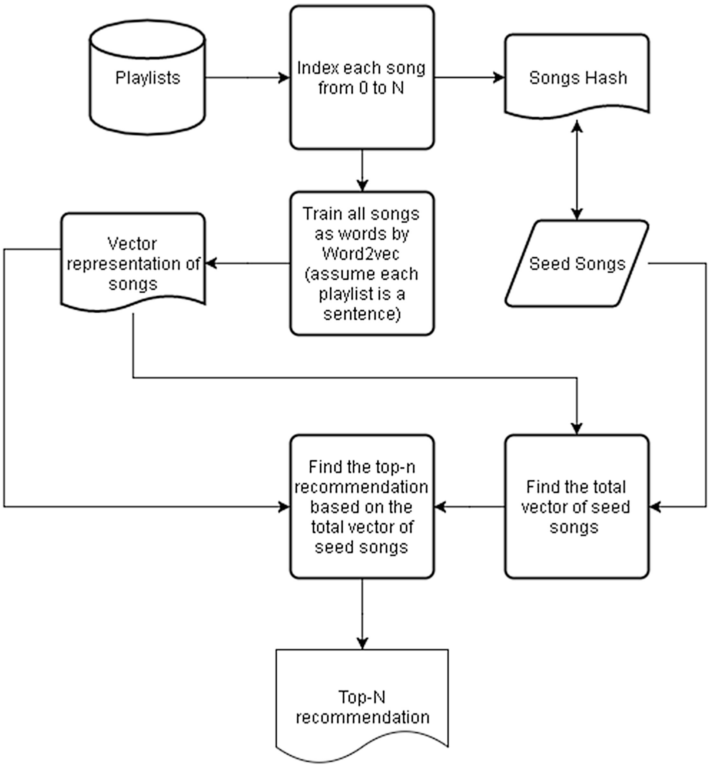

Playlists
Index each song from 0 to N
Songs Hash
Train all songs as words by Word2vec (assume each playlist is a sentence)
Vector representation of songs
Seed Songs
Find the top-n recommendation based on the total vector of seed songs
Find the total vector of seed songs
Top-N recommendation

Fig. 1. Playlist generation architecture based on Word2vec

model is able to calculate the vector that contains users taste. The general model is illustrated in Fig. 1.

4 Evaluation Design and Experiments

This section presents the studies to evaluate the proposed architecture for playlist generation. Here, the proposed system is evaluated by using a real world song dataset. The performance results in terms of hit rates are presented at the end of the section.

As a real world song dataset, we use a dataset which was published by Cornell University and collected by Shuo Chen from Department of Computer Science. The dataset contains playlist sequences and tag data which was obtained from Yes.com and Last.fm. To get datasets of various sizes, researchers pruned the raw data so that only the songs with a number of appearances above a certain threshold are kept. Then the pruned set is divided into a training set and a testing set, making sure that each song has appeared at least once in the training set. This playlist data is divided into three parts then named as *yes_small, yes_big, and yes_complete*. The detailed properties of the dataset are shown in Table 1 [8, 15, 16].

Table 1. Properties of Playlist Dataset

Properties	*Yes_small*	*Yes_big*	*Yes_complete*
Appearance threshold	20	5	0
Number of songs	3,168	9,775	75,262
Number of train transitions	134,431	172,510	1,542,372
Number of test transitions	1,191,279	1,602,079	1,298,181

In our evaluation design, sequences of each playlist in the test set, the first song is set apart as a target song and is dropped from the sequence of the playlist. The principal purpose is to predict the removed song by the model which has the top-N recommendations. If the song is found by the model, it can be acceptable that the song is predicted correctly. The corresponding method (Eq. 3) is known as the hit rate evaluation [17].

$$\text{HitRate(Train, Test)} = \frac{1}{\|\text{Test}\|} \sum_{(\text{h,t}) \in \text{Test}} \delta_{t, R_{\text{Train}}(\text{h})} \tag{3}$$

where $\text{R}_{\text{Train}}(\text{h})$ is a recommendation list which contains top-N recommendation computed by our model based on the training dataset and h is a vector of the set of songs (playlist). According to this equation, $\delta_{t,R_{\text{Train}}(\text{h})} = 1$ if the removed song is in the list of R_{Train} and 0 otherwise.

First, we evaluate our model on *yes_small* with various parameters for Word2vec model and observe hit rate scores (see Figs. 2, 3, 4, 5). Colors in the figures indicate number of vector sizes (50-1000). Then, we choose first five average highest score and evaluate *yes_big* and *yes_complete* with these parameters.

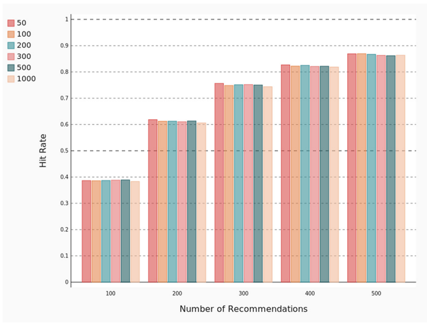


Fig. 2. Hit rates for the window size is 5 on *yes_small*

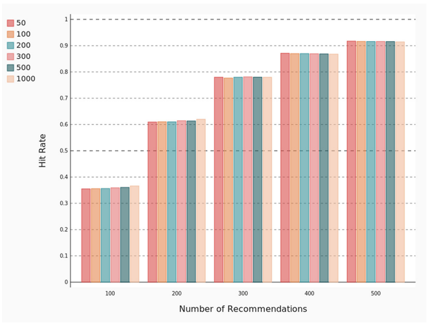


Fig. 3. Hit rates for the window size is 25 on *yes_small*

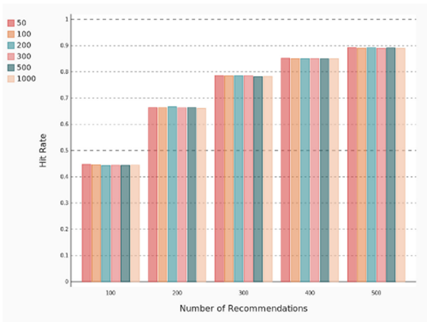


Fig. 4. Hit rates for the window size is 125 on *yes_small*

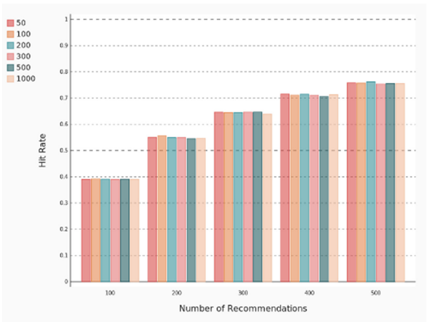


Fig. 5. Hit rates for the window size is 250 on *yes_small*

According to Figs. 2, 3, 4, 5, the best five successful parameters are shown in Table 2.

Table 2. According to *yes_small*, parameters of first five best average hit rate

	Windows size	Vector size	Average hit rate
1	125	50	0.7282
2	125	200	0.7276
3	125	100	0.7270
4	125	300	0.7266
5	125	500	0.7260

We also use parameters described in Table 2 for measuring *yes_big* and *yes_-complete*. Hit rates results for *yes_big* dataset are shown in Fig. 6 and *yes_complete* dataset in Fig. 7.

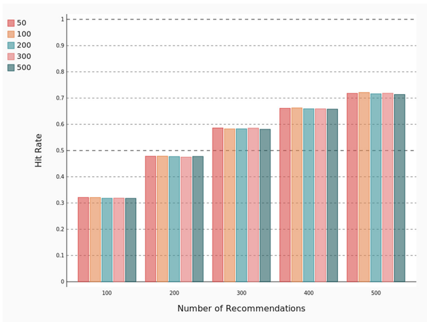


Fig. 6. Hit rates on *yes_big*

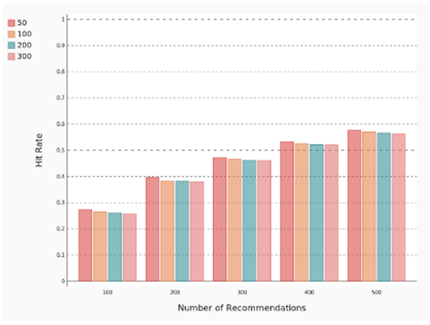


Fig. 7. Hit rates on *yes_complete*

5 Conclusion and Future Works

The study presented in this paper proposed a novel playlist generation method. The evaluations and tests proved that the proposed recommendation system can easily assist music audiences to discover their personalized playlists with an easy and performance efficient way. In this study, we also showed that Word2vec algorithm can be used with Apache Spark big data framework and run on distributed vector representation of songs to produce a playlist reflecting a person's personal tastes.

Recommender systems are broad and challenging area to research, so the elasticity of Word2vec model provides exciting opportunities for future works. However, our model cannot solve the cold start problem, so this model can be combined with traditional models like content-based systems for solving cold start problem or an advanced new approach based on vectors.

References

1. Bobadilla, J., Ortega, F., Hernando, A., Gutiérrez, A.: Recommender systems survey. Knowl. Based Syst. **46**, 109–132 (2013)
2. Celma, O.: Music Recommendation. Springer, Berlin, Heidelberg (2012)
3. Levy, M., Sandler, M.: Music information retrieval using social tags and audio. IEEE Trans. Multimedia **11**, 383–395 (2009)
4. Knees, P., Pohle, T., Schedl, M., Schnitzer, D., Seyerlehner, K.: A document-centered approach to a natural language music search engine. Adv. Inf. Retr. **4956**, 627–631 (2008)
5. Logan, B.: Content-based playlist generation: exploratory experiments. In: Processing of 3rd International Conference on Music Information Retrieval, Paris, France, 13–17 October, pp. 1–2 (2002)
6. Elias, P., Arthur, F., Gerhard, W.: Improvements of audio-based music similarity and genre classification. In: Processing of 6th International Conference on Music Information Retrieval, London, UK, 11–15 September, pp. 634–637 (2005)
7. Paul, Q., Robert, P., Michael, M., Maxwell, W., Scott, J., Richard, W.: Playlist generation, delivery and navigation. U.S. Patent No US20030135513 A1 (2002)

8. Shuo, C., Josh, L.M., Douglas T., Thorsten, J.: Playlist prediction via metric embedding. In: Processing of Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, USA, 12–16 August, pp. 714–722 (2012)
9. Gärtner, D.: User adaptive music similarity with an application to playlist generation. Ph.D. Thesis, University at Karlsruhe (TH), Germany, Carnegie Mellon University, Pittsburgh, PA, USA (2006)
10. Baltrunas, L., Kaminskas, M., Ludwig, B., Moling, O., Ricci, F., Aydin, A., Lüke, K.-H., Schwaiger, R.: InCarMusic: context-aware music recommendations in a car. In: Huemer, C., Setzer, T. (eds.) EC-Web 2011. LNBIP, vol. 85, pp. 89–100. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23014-1_8
11. Masahiro, N., Takaesu, H., Demachi, H., Oono, M., Saito, H.: Development of an automatic music selection system based on runner's step frequency. In: Processing of 9th International Conference on Music Information Retrieval, Pennslyvania, USA, 14–18 September, pp. 193–198 (2008)
12. Wang, X., Wang, Y., Hsu, D., Wang, Y.: Exploration in interactive personalized music recommendation: a reinforcement learning approach. ACM Trans. Multimedia Comput. Commun. Appl. **11**, 1–22 (2014)
13. Bogdanov, D., Herrera, P.: How much metadata do we need in music recommendation? A Subjective evaluation using preference sets. In: Processing of 12th International Conference on Music Information Retrieval, Miami, Florida, USA, 24–28 October, pp. 97–102 (2011)
14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. Adv. Neural Inf. Process. Syst. **26**, 3111–3119 (2013)
15. Moore, J.L., Chen, S., Joachims, T., Turnbull, D.: Learning to embed songs and tags for playlists prediction. In: Processing of 13th International Society for Music Information Retrieval, Porto, Portugal, 8–12 October, pp. 349–354 (2012)
16. Shuo, C., Jiexun, X., Thorsten, J.: Multi-space probabilistic sequence modeling. In: Processing of the 19th ACM Conference on Knowledge Discovery and Data Mining, Chicago, USA, 11–14 August, pp. 865–873 (2013)
17. Bonnin, G., Jannach, D.: A comparison of playlist generation strategies for music recommendation and a new baseline scheme. In: Processing of the Twenty-Seventh AAAI Conference on Artificial Intelligence, Washington, USA, 16–18 July, pp. 16–23 (2013)

A Distributed Framework for Early Trending Topics Detection on Big Social Networks Data Threads

Athena Vakali(✉), Nikolaos Kitmeridis, and Maria Panourgia

Informatics Department, Aristotle University, Thessaloniki, Greece
{avakali,nkitmeri,panomari}@csd.auth.gr

Abstract. Social networks have become big data production engines and their analytics can reveal insightful trending topics, such that hidden knowledge can be utilized in various applications and settings. This paper addresses the problem of popular topics' and trends' early prediction out of social networks data streams which demand distributed software architectures. Under an online time series classification model, which is implemented in a flexible and adaptive distributed framework, trending topics are detected. Emphasis is placed on the early detection process and on the performance of the proposed framework. The implemented framework builds on the lambda architecture design and the experimentation carried out highlights the usefulness of the proposed approach in early trends detection with high rates in performance and with a validation aligned with a popular microblogging service.

1 Introduction

Big data threads are rapidly produced in social networks, with emerging textual data streams, evolving unpredictably, in a non pre-determined manner. Such data threads offer fertile ground for analytics which can reveal trends, phenomena, and knowledge. Already in social networks, such as in Twitter microblogging service[1], data threads unfolding over time are characterized as trending topics when they exhibit viral cascading data patterns.

This work is motivated by the fact that predicting whether a social network's topic will become a trend, prior it is declared as a trend by the social network itself (i.e. Twitter) can be addressed as a classification problem, tackling with evolving big data requirements. Predicting topics which will actually end up as trends in social networks has as major obstacles the real time, often bursty, data production, along with the missing exemplar time series to ignite data processing.

While previous research has analyzed trending topics mostly on a long term, recent efforts have focused on detection of tweets' trend, in an almost real time fashion due to ongoing and continuous social networks events [1, 2]. Setting the appropriate distributed frameworks to support trend detection in social networks, embeds big data demands and real time requirements coverage [3, 4]. Such frameworks must be tunable

[1] https://twitter.com/whatstrending.

© Springer International Publishing AG 2017
P. Angelov et al. (eds.), *Advances in Big Data*, Advances in Intelligent Systems and Computing 529, DOI 10.1007/978-3-319-47898-2_20

to offer the appropriate testbeds for important applications highly related with trend detection, such as fraud and emergencies [5, 6].

This paper addresses the problem of popular topics and trends prediction out of social networks data streams, under an online time series classification model, which is implemented over a flexible and adaptive distributed framework. Trend prediction is employed in a real time fashion, utilizing big data principles and technologies, under a framework designed in Lambda architecture outline. Lambda architecture has been chosen due to its flexibility in processing both evolving and static data threads, based on a tunable latent source model. The proposed model sets the so called latent source "signals" corresponding to an exemplar event of a certain type, and a clustering process is combining with the classification tasks of labeling the data threads over specific categories (either detected as trends or not). Different similarity measures are used to support classification via the latent source model. The proposed work is validated under an experimentation setting with actual Twitter large scale data threads, having the Twitter declared trending topics, as the ground truth for detected viral topics evaluation. Under the experimentation carried out on the proposed distributed architecture, trending topics prediction has reached an accuracy of 78.4 %, while in more than one third of studied cases the prediction is successful in advance of the respective Twitter trends declaration.

The structure of the paper is as follows. Next section outlines the principles of both early trend detection in social networks and of the suitability of the architecture to carry out such a process. Section 3 has the details for the proposed micro-blogging trending topics prediction implementation, while experimentation and results are discussed in Sect. 4. Section 5 has conclusions, indicating future work.

2 A Framework for Early Trend Detection in Social Networks

Trend detection problem as a time series classification problem has been studied in several earlier efforts [7, 8], with a focus on classifying a topic as a trend based either on the Euclidean distance of topic's time series and the time series of trend (or not trend) training sets, respectively or streaming time series into a vector which is then used for clustering over the evolving time series. Social networks (Twitter in particular) data stream emergence in real time, has set the floor for research approaches capable of predicting phenomena, such as disease outbreaks, emergencies, and opinions shifts [9–11]. Popularity alone is not adequate for a topic to break into the trends list since Twitter favours novelty over popularity. It is the velocity of interactions around a specific topic which should be monitored in relation to a baseline interactions level [12], [1]. To resolve such issues earlier efforts have focused on the so called latent source model for time series, which naturally leads to a "weighted majority voting" classification rule, approximated by a nearest-neighbor classifier [7]. The proposed work is inspired by this latent classification model which is advanced here with novel distributed data processing methodologies, with the inclusion of social features related to a process favoring the early detection of a trending topic. To support this approach two distance metrics of cosine and squared Euclidean are utilized advancing earlier efforts which have favored the use of only Euclidean space metrics.

Complex architectures have been designed and tested on evolving big data management use cases extracting useful insights from large scale data collections [13, 14]. Among such distributed frameworks, in this work Lambda Architecture[2] is proposed since it gains ground due to its well defined set of architecture principles allowing both batch and real-time stream data processing [15, 16]. The three Lambda architecture's layers, i.e. the Batch, the Serving, and the Speed layer support the proposed work's framework since they enable processing of high volumes of data in either batch or on a real time mode. Data streams initially enter the system and are dispatched to batch the speed layer.

In our case, since data are collected from social networks, data arrival originates from the Twitter streaming API, and at the batch layer side, a master dataset, proceeds with an immutable, append-only set of raw data, computing arbitrary views. Hadoop[3] and its implementation of MapReduce model has been chosen since it is ideal for this batch layer, leveraging on HDFS. Serving layer gets the output from the batch layer as a set of flat files containing pre-computed views and exposes views for querying. Bridging among these layers and the support of a fast database is handled with specific tools (like Apache Drill[4]) in combination with an in-memory key-value datastore. Speed layer computes data views as well, but it does so incrementally by balancing high latencies of the batch layer with real time views computations. Real time views are of transient nature along with the data propagation rates (from the batch and serving layers), under the so called "complexity isolation" process which pushes into the layer only temporary results [17]. In response to users queries, merging results from batch and real-time views are delivered to the user under an integrated common interface. This architecture is next exploited for the specific social networks trend detection problem, due to its capabilities of handling both static and evolving big data threads.

3 A Micro-blogging Trending Topics Prediction Implementation

The proposed framework builds on Lambda Architecture, with a batch, a speed and a serving layer. However, the key difference between them arises from the framework's execution work flow which is depicted in Fig. 1. Both batch and speed layers are fed with the same incoming data stream, but batch layer executes periodic processing on the whole of the dataset, until the specified time, while speed layer performs the same processing, in real time, in the part of data acquired while the former layer is busy. Finally, the results of the batch layer processing stored in the serving layer are aggregated with the results of the serving layer processing and the outcome are global views of the dataset in response to specific queries. Data processing procedure on batch

[2] http://lambda-architecture.net/.

[3] https://hadoop.apache.org/.

[4] https://drill.apache.org/.

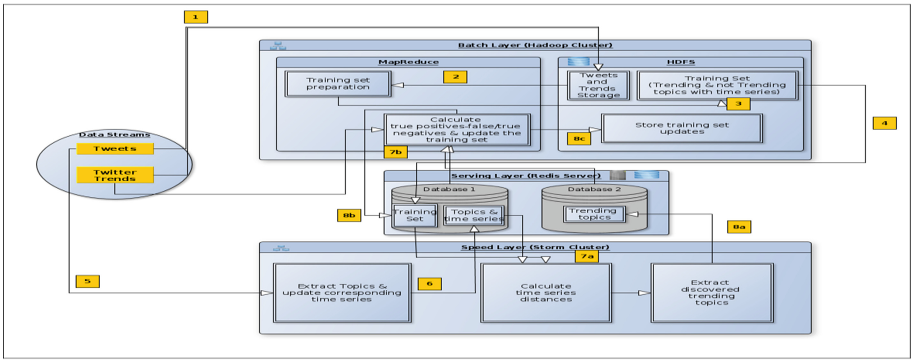

1
Batch Layer (Hadoop Cluster)
MapReduce
HDFS
Training set preparation
2
Tweets and Trends Storage
Training Set (Trending & not Trending topics with time series)
3
Calculate true positives-false/true negatives & update the training set
Store training set updates
8c
7b
4
Data Streams
Tweets
Twitter Trends
Serving Layer (Redis Server)
Database 1
Database 2
Training Set
Topics & time series
Trending topics
8b
8a
Speed Layer (Storm Cluster)
7a
5
Extract Topics & update corresponding time series
6
Calculate time series distances
Extract discovered trending topics

Fig. 1. Twitter trend prediction framework

and speed layers of the implemented framework differ in that batch layer is used for both the creation of the training set and the periodic update of the training set, while speed layer covers the topics classification into trends through a series of execution steps. Big volume of tweets is used to define the training set, in an asynchronous execution at the speed layer, accessing results stored at the service layer.

In this work, early prediction of Twitter's trending topics involves a series comparison process, requiring a sample of the overall tweets dataset over a specific period. This sample is built on the basis of the Twitter declared trending topics and by performing retrospective analysis on the dataset, to generate the time series for each topic. For each topic, the ratio of summarized distances is calculated and once it is above a threshold the topic is classified as a potential trending topic. Proposed execution approach novelty is due to the next key points:

- training set's time series are constructed by a small rate of the overall tweets dataset;
- the time series of topics to be tested are not static, but are generated in real time in a form of a sliding window, maintaining low percentage of tweets for constructing the time series.

Creation of the final form of time series is given in Fig. 2, which visualizes the flow of a time series generation example, in line with the next proposed steps:

- timestamps and exact date of the topic being declared as a trending topic;
- definition of a time range from the moment just before the topic became trended, i.e. the time of the last kept time slot;
- removal of all time slots preceding the one that signifies the start of the aforementioned time range;
- production of the final form of the time series by applying a set of normalization filters.

Time series of non trending topics is generated under a similar procedure. Density of time series to be chosen may range from too sparse to too dense, and the normalization filters proposed here also range from baseline to spike ones, accordingly [7].

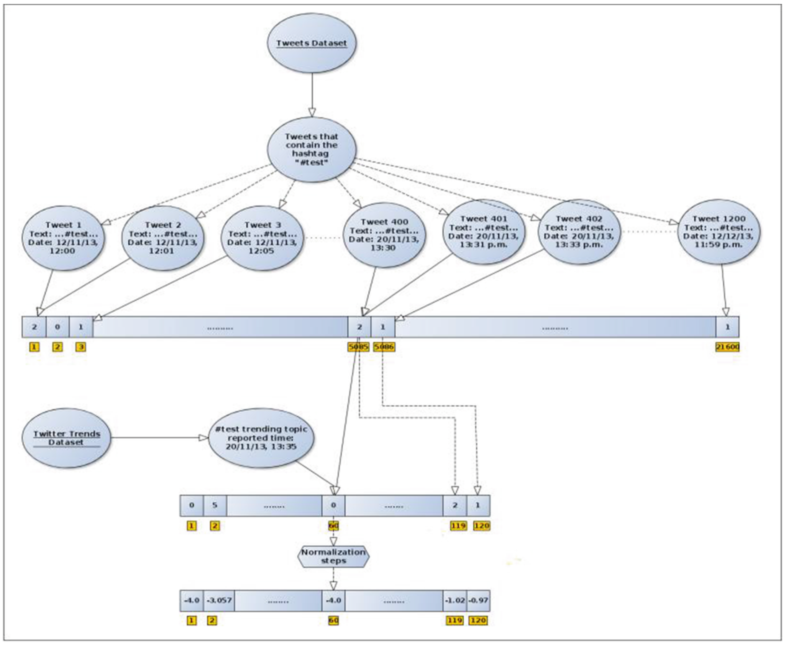

Tweets Dataset
Tweets that contain the hashtag "#test"
Tweet 1 Text: ...#test... Date: 12/11/13, 12:00
Tweet 2 Text: ...#test... Date: 12/11/13, 12:01
Tweet 3 Text: ...#test... Date: 12/11/13, 12:05
Tweet 400 Text: ...#test... Date: 20/11/13, 13:30
Tweet 401 Text: ...#test... Date: 20/11/13, 13:31 p.m.
Tweet 402 Text: ...#test... Date: 20/11/13, 13:33 p.m.
Tweet 1200 Text: ...#test... Date: 12/12/13, 11:59 p.m.
Twitter Trends Dataset
#test trending topic reported time: 20/11/13, 13:35
Normalization steps

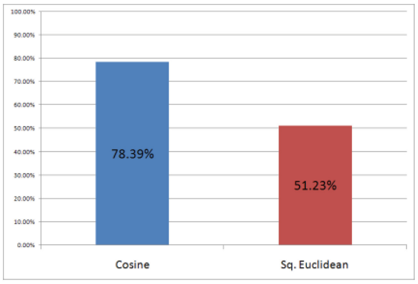
Fig. 2. Trending topic's time series generation example

4 Experimentation and Results

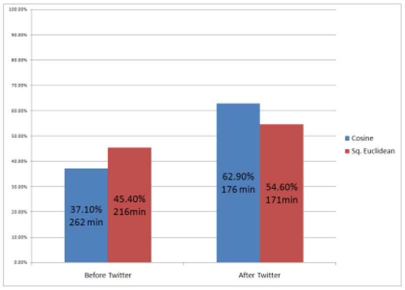
The overall framework (depicted in Fig. 2) has been installed at the GRNET's Cloud service, called Okeanos[5], and it consisted of a Hadoop cluster (batch layer), a Storm cluster (speed layer) and a single node (serving layer). The available resources from Okeanos were 48 CPU cores, 46 GB of RAM and 600 GB of disk storage and in total, 14 virtual machines have been created, using these resources, for the implementation of above clusters and servers.

The proposed framework has been stress tested under various big data tweets threads collected via Twitter streaming API, over several time windows. Due to lack of space, here a sample of 1 month period (Nov-Dec 2013) is discussed, of its total size exceeding 300 GB, along with the Twitter trending topics announced for the same time window. The parallel collection of tweets and the trending topics followed the proposed time series generation, by dividing the time window of the one month in fine grained time slots of 2 min. The evaluation of the proposed methodology implemented on the lambda inspired architecture has reached improved performance since almost 80 % of the actual trending topics were classified as potential trending topics by the method (Fig. 3a). Specifically, 79 % of the correctly classified trending topics, were claimed as trending topics by the method under cosine similarities, and this was

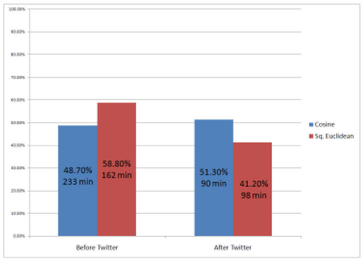
[5] https://okeanos.grnet.gr.



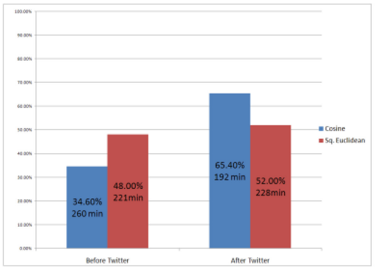
(a) true positives percentages at the end of the 48 hour execution, for Cosine and Squared Euclidean



(c) 36 hours of execution



(b) 24 hours of execution



(d) 48 hours of execution

Fig. 3. Rates of trending detection prior and upon Twitter declaration

actually performed earlier than the Twitter API, with the mean time to be 1.43 h earlier and only 4 % of the potential trending topics were false positive. Figure 3(b–d), also depicts the rates of trending detection ignition for true positives prior and upon Twitter trending topics declaration, with the similarities estimation based on either Euclidean or cosine metrics, in the regular intervals of 24th, 36th, and 48th execution hours.

The percentage of the reports of trending topics by the implemented framework been done before the respective reporting from Twitter follows a downward path throughout the execution (as partially depicted in Fig. 4) which illustrates the whole of the execution flow.

On the contrary, mean times of the true positive's early reporting in comparison to the timing of Twitter's reports, are quite satisfactory and such improved performance is enhanced by the fact the mean times of the later reports, in comparison to Twitter's, are significantly smaller.

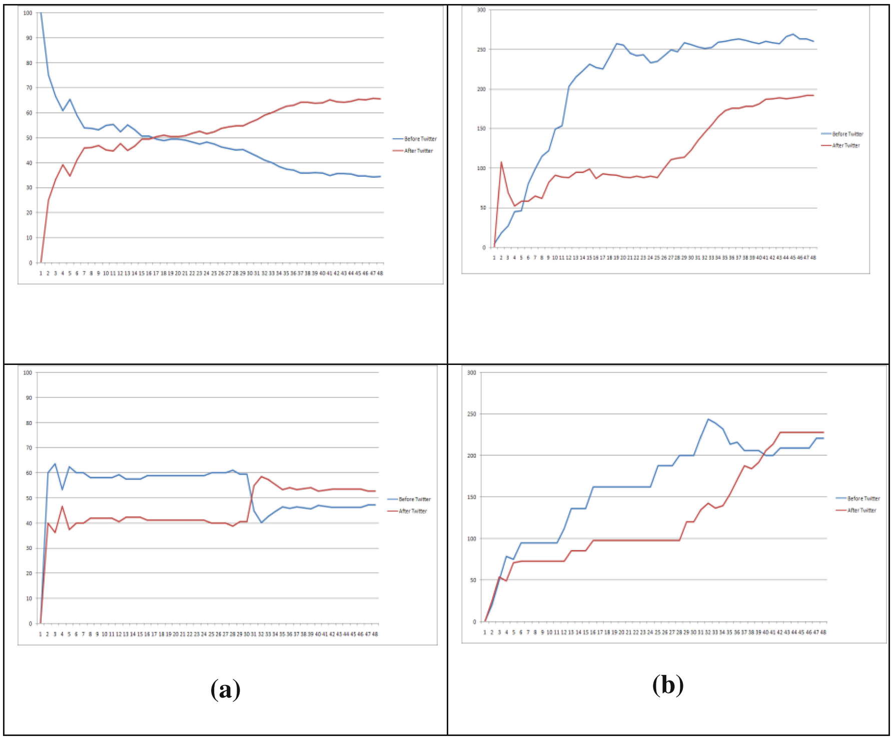


Fig. 4. Hourly Percentages (a) and Mean times (b) fluctuation for trends detection prior and after Twitter declaration

5 Conclusion and Future Work

In summary, the overall percentage of the trending topics of the proposed framework validates its positive performance which flows similarly to Twitter's respective percentages of trending topics detection, its strong contribution lies in its ability to early detect trending topics, and it can further be extended and improved.

Use of different distance metrics for the time series comparison can be beneficial since time series oriented distance metrics like, edit distance with Real Penalty, move-split-merge and sparse dynamic time wrapping are currently under experimentation in the proposed framework. Normalization and filtering of the time series can also be extended and refined while the implemented algorithms execution workflow, can also become more "loyal" to the principles of the Lambda Architecture. In such a case a more powerful infrastructure and the inclusion of a distributed NoSQL database to serve the scope of the serving layer are under consideration. In terms of its applicability, observing and analyzing social networks textual threads as evolving time series, has many future relevant applications, such as recommenders and other personalized services.

Regarding the infrastructure part of the current work, a challenging attempt would be the usage of more recent technologies and tools. While the presented architecture is heavily based on [17], other frameworks could be more effective. Indicatively, Apache Spark[6] could be used as the batch layer and Apache Flink[7] as the serving layer, in Lambda Architecture.

References

1. Arkaitz, Z.: Real-time classification of Twitter trends. J. Assoc. Inf. Sci. Technol. **66**, 462–473 (2015)
2. Salvatore, G., Lo Re, G., Morana, M.: A framework for real-time Twitter data analysis. Comput. Commun. **73**, 236–242 (2016)
3. Li, J.: Bursty event detection from microblog: a distributed and incremental approach. Concurrency Comput. Pract. Exp. **28**, 3115–3130 (2015)
4. Manirupa, D.: Towards methods for systematic research on big data. In: IEEE International Conference on Big Data (Big Data). IEEE (2015)
5. Giatsoglou, M., Chatzakou, D., Shah, N., Faloutsos, C., Vakali, A.: Retweeting activity on Twitter: signs of deception. In: Cao, T., Lim, E.-P., Zhou, Z.-H., Ho, T.-B., Cheung, D., Motoda, H. (eds.) PAKDD 2015. LNCS, vol. 9077, pp. 122–134. Springer, Heidelberg (2015)
6. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes Twitter users: real-time event detection by social sensors. In: Proceedings of the Nineteenth International WWW Conference. ACM (2010)
7. Stanislav, N., Chen, G., Shah, D.: A latent source model for nonparametric time series classification. In: Advances in Neural Information Processing Systems (2013)
8. Kontaki, M., Papadopoulos, A.N., Manolopoulos, Y.: Continuous trend-based classification of streaming time series. In: Eder, J., Haav, H.-M., Kalja, A., Penjam, J. (eds.) ADBIS 2005. LNCS, vol. 3631, pp. 294–308. Springer, Heidelberg (2005)
9. Szomszor, M., Kostkova, P., de Quincey, E.: #Swineflu: Twitter predicts swine flu outbreak in 2009. In: Szomszor, M., Kostkova, P. (eds.) e-Health. LNICST, vol. 69, pp. 18–26. Springer, Heidelberg (2011)
10. Lei, S.: Predicting US primary elections with Twitter. http://snap.stanford.edu/social2012/papers/shi.pdf. Accessed 2012
11. Wang, Y.: To Follow or Not to Follow: Analyzing the Growth Patterns of the Trumpists on Twitter. arXiv:1603.08174 *(*2016)
12. Mathioudakis, M., Koudas, N.: Twittermonitor: trend detection over the twitter stream. In: SIGMOD ACM (2010)
13. Gorton, I., Klein, K.: Distribution, data, deployment: Software architecture convergence in big data systems. IEEE Softw. **32**(3), 78–85 (2015)
14. Tang, B.: A hierarchical distributed fog computing architecture for big data analysis in smart cities. In: Proceedings of the ASE BigData & SocialInformatics. ACM (2015)

[6] http://spark.apache.org/.

[7] https://flink.apache.org/.

15. Mariam, K.: Lambda architecture for cost-effective batch and speed big data processing. In: IEEE Big Data International Conference (2015)
16. Martínez-Prieto, M.: The solid architecture for real-time management of big semantic data. Future Gener. Comput. Syst. **47**, 62–79 (2015)
17. Marz, N.: Big data : principles and best practices of scalable realtime data systems. O'Reilly Media (2013)

Multi-task Deep Neural Networks for Automated Extraction of Primary Site and Laterality Information from Cancer Pathology Reports

Hong-Jun Yoon, Arvind Ramanathan, and Georgia Tourassi(✉)

Oak Ridge National Laboratory, Health Data Sciences Institute,
Oak Ridge, TN, USA
{yoonh, ramanathana, tourassig}@ornl.gov

Abstract. Automated annotation of free-text cancer pathology reports is a critical challenge for cancer registries and the national cancer surveillance program. In this paper, we investigated deep neural networks (DNNs) for automated extraction of the primary cancer site and its laterality, two fundamental targets of cancer reporting. Our experiments showed that single-task DNNs are capable of extracting information with higher precision and recall than traditional classification methods for the more challenging target. Furthermore, a multi-task learning DNN resulted in further performance improvement. This preliminary study, indicate the strong potential for multi-task deep neural networks to extract cancer-relevant information from free-text pathology reports.

Keywords: Multi-task learning · Deep neural network · Cancer pathology report · Natural language processing

1 Introduction

Pathology reports are free-text documents containing information about human tissue specimens. They are a standard component of cancer patient clinical reporting and management. Extraction of information from pathology reports is a critical but largely manual process performed by cancer registries as part of the national cancer surveillance program. In the past few years there has been tremendous interest in developing natural language processing (NLP) methods to automate this time-consuming process and ease the burden of manual abstraction [1, 2]. However, due to the nature of the pathology report, simple text matching is not effective. Some well-known challenges are missing punctuation, clinical findings interspersed with explanations, as well as complex information about multiple specimens mentioned throughout the report.

Due to these challenges, machine learning-based NLP methods have been explored as a promising approach. Martinez and Li applied Naïve Bayes, Support Vector Machine (SVM), and AdaBoost classifiers to identify information from pathology reports including primary cancer site, with feature selections from Bag-of-Words, lemma, and annotations of lexical databases [3]. Jouhet also applied Naïve Bayes and

SVM to classify the topographical and morphological codes using the Term Frequency with Inverse Document Frequency (TF-IDF) of the Bag-of-Words features [4]. Kavuluru proposed n-gram features to identify 14 primary cancer sites using Logistic Regression classifier [5]. Reported results vary widely depending on study, information target, classifier, and feature selection method with F-scores ranging from 0.4 to 0.9.

Recently, there is growing interest in exploring deep learning (DL) methods for NLP tasks [6–8]. Applications range from low-level tasks such as part-of-speech tagging and named entity recognition to high-level tasks such as information retrieval, semantic analysis, and sentence relation modeling. DL methods bypass laborious hand-engineered feature extraction while often boosting NLP task performance by learning high-level abstractions of word-level representations and sentence-level representations. In the biomedical domain, DL-based NLP has been applied for analysis of biomedical articles and electronic medical records [9, 10].

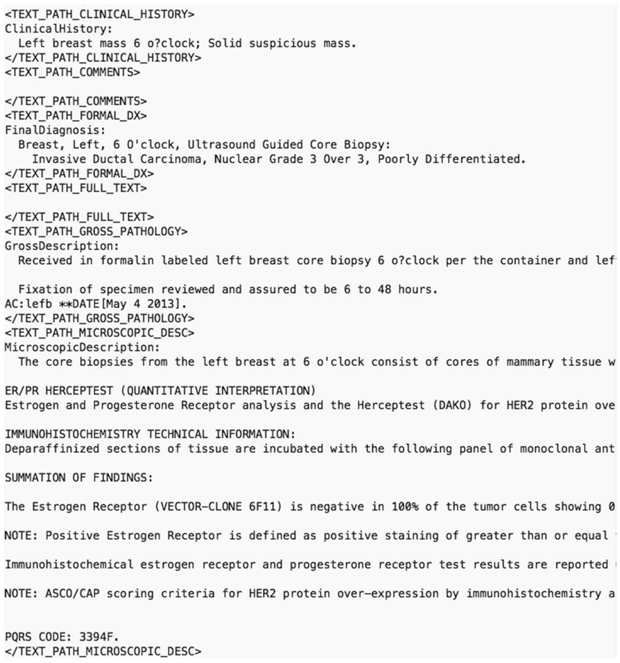
In this study we focused on automated information extraction from cancer pathology reports, and specifically extraction of two key parameters, primary cancer site and its laterality. First, we explored the application of deep neural networks (DNNs), a cascade of hidden representations known to be able to learn complex mappings between the input features and the output. Then, we studied the use of a multi-task learning (MTL) paradigm to improve upon the DNN performance. MTL is a learning framework designed to exploit commonality of different but related classification tasks as a means to achieve higher classification performance [11]. Specifically, we performed comparative analysis between an MTL-based DNN classifier versus single-task DNN classifiers. In addition, we applied Naïve Bayes classifiers and Logistic Regression, two common classification choices.

The paper is organized as follows: Sect. 2 provides a brief description of the pathology report structure; Sect. 3 describes the available dataset, the feature extraction methods, the classifiers, experimental design, and the performance evaluation protocols employed in this study; finally, Sects. 4 and 5 describes results and discussion from our studies.

2 Cancer Pathology Reports

The Surveillance, Epidemiology, and End Results (SEER) program began collecting data on cancer cases since 1973, in seven US states, and has expanded to 18 population-based cancer registries [12]. The SEER program collects incidence and survival of patients with malignant tumors, representing approximate 30% of the US population. Cancer pathology reports are a primary information source for cancer registries.

A sample pathology report is shown Fig. 1. A typical report is divided into sections marked by angle-bracketed tags listed in Table 1. As Fig. 1 shows, not all the sections are filled, therefore simply reviewing few sections to match keywords is not an effective and reliable information extraction strategy.



```
<TEXT_PATH_CLINICAL_HISTORY>
ClinicalHistory:
  Left breast mass 6 o?clock; Solid suspicious mass.
</TEXT_PATH_CLINICAL_HISTORY>
<TEXT_PATH_COMMENTS>

</TEXT_PATH_COMMENTS>
<TEXT_PATH_FORMAL_DX>
FinalDiagnosis:
  Breast, Left, 6 O'clock, Ultrasound Guided Core Biopsy:
    Invasive Ductal Carcinoma, Nuclear Grade 3 Over 3, Poorly Differentiated.
</TEXT_PATH_FORMAL_DX>
<TEXT_PATH_FULL_TEXT>

</TEXT_PATH_FULL_TEXT>
<TEXT_PATH_GROSS_PATHOLOGY>
GrossDescription:
  Received in formalin labeled left breast core biopsy 6 o?clock per the container and lef

  Fixation of specimen reviewed and assured to be 6 to 48 hours.
AC:lefb **DATE[May 4 2013].
</TEXT_PATH_GROSS_PATHOLOGY>
<TEXT_PATH_MICROSCOPIC_DESC>
MicroscopicDescription:
  The core biopsies from the left breast at 6 o'clock consist of cores of mammary tissue w

ER/PR HERCEPTEST (QUANTITATIVE INTERPRETATION)
Estrogen and Progesterone Receptor analysis and the Herceptest (DAKO) for HER2 protein ove

IMMUNOHISTOCHEMISTRY TECHNICAL INFORMATION:
Deparaffinized sections of tissue are incubated with the following panel of monoclonal ant

SUMMATION OF FINDINGS:

The Estrogen Receptor (VECTOR-CLONE 6F11) is negative in 100% of the tumor cells showing 0

NOTE: Positive Estrogen Receptor is defined as positive staining of greater than or equal

Immunohistochemical estrogen receptor and progesterone receptor test results are reported

NOTE: ASCO/CAP scoring criteria for HER2 protein over-expression by immunohistochemistry a


PQRS CODE: 3394F.
</TEXT_PATH_MICROSCOPIC_DESC>
```

Fig. 1. Screenshot of sample free text pathology report. Note that not all sections are filled, e.g., <TEXT_PATH_COMMENTS>.

Table 1. List of tags on pathology reports.

Tag
<PATIENT_DISPLAY_ID>
<TUMOR_RECORD_NUMBER>
<RECORD_DOCUMENT_ID>
<TEXT_PATH_CLINICAL_HISTORY>
<TEXT_PATH_COMMENTS>
<TEXT_PATH_FORMAL_DX>
<TEXT_PATH_FULL_TEXT>
<TEXT_PATH_GROSS_PATHOLOGY>
<TEXT_PATH_MICROSCOPIC_DESC>
<TEXT_PATH_NATURE_OF_SPECIMENS>
<TEXT_PATH_STAGING_PARMS>
<TEXT_PATH_SUPP_REPORTS_ADDENDA>

3 Methods

3.1 Dataset

In this study, we used de-identified data obtained with an IRB-approved protocol from the National Cancer Institute. The acquired data consisted of 1,976 de-identified free text pathology reports of breast and lung cancers from five SEER cancer registries with human-annotated gold standards. Of those, we used 985 reports for which the annotations clearly stated their primary site category (breast, lung) and laterality (left, right). There were 313 cases of left breast, 285 cases of right breast, 156 cases of left lung, and 231 cases of right lung. To mitigate case prevalence bias, we generated a dataset with 300 cases per primary (P) cancer site and laterality (L) grouping by either over-sampling the under-represented or by randomly selecting 300 cases from the over-represented group in the available data. This process resulted in 1,200 cases (=2 primary sites × 2 laterality categories × 300 cases per (P,L) group).

3.2 Feature Extraction

First, we applied a few straightforward pre-processing steps to delete patient display IDs, tumor record numbers, and record document IDs. We also removed angle-bracketed tags, and replaced Unicode escape sequences into appropriated ASCII characters if any.

Next, we converted the reports to lowercase letters and we removed stop words and words of non-alphabet characters (i.e., numbers). Then, we adopted n-grams for feature representation with $n = 1, 2$, namely unigrams and bigrams. N-gram is a popular text representation approach in computational linguistics capturing contiguous sequences of n items (e.g., phonemes, syllables, letters, words) from a given text corpus [13]. The total number of n-grams found in our dataset was 418,346. Last, we collected the 400 most frequently used word-based n-grams for composing features. Table 2 shows the 30 most frequently occurring n-grams. Feature vectors of the pathology reports were

Table 2. Top 30 most frequently occurring n-grams and their number of occurrence in the pathology report data.

n-gram	Frequency	n-gram	Frequency	n-gram	Frequency
Specimen	2128	Carcinoma	1574	Number	1325
Submitted	2062	Description	1479	Adenocarcinoma	1319
Tissue	1931	Biopsy	1477	Grade	1295
Diagnosis	1916	Gross	1455	Patient	1281
cm	1850	Negative	1431	Tan	1277
Labeled	1771	Formalin	1427	Present	1268
Microscopic	1741	Histologic	1403	Identified	1254
Received	1656	Signed	1377	Positive	1221
Tumor	1649	Case	1325	Gross description	1166
Clinical	1591	Performed	1325	Invasion	1166

represented by the term frequency–inverse document frequency (TF-IDF) of the n-grams. TF-IDF measures how important an n-gram is to a document (e.g., a pathology report) relative to whole data corpus [14].

3.3 Deep Neural Networks

A Deep Neural Network (DNN) [15] is a feed-forward artificial neural network that has more than one hidden layers between its input and output layers. On the hidden layers, DNN computes the activations of conditionally independent hidden units h_j given the input vector h_{j-1} as follows

$$h_j = \sigma\left(\sum_i h_{j-1} w_{ij}\right) + b_j$$

where w_{ij} is the weight on a connection to output unit j from input unit i, b_j is the bias input of unit j, and $\sigma(x) = \frac{1}{1+e^{-x}}$. The output layer k produces class probability p_k over all classes N by using the softmax nonlinearity

$$p_k = \frac{\exp(h_k)}{\sum_{n=1}^{N} \exp(h_n)}.$$

DNN is trained using Stochastic Gradient Descent (SGD) [16], where the computation of derivatives is made with random mini-batch of training cases for updating the weights in proportion to the gradient. In the context of our study, two separate single-task DNNs were developed for text analysis of pathology reports, namely one for extracting the primary cancer site and another to extract cancer site laterality.

3.4 Multi-task Learning

Multi-task learning (MTL) is a machine learning paradigm way that learns a series of different but related tasks in parallel using a shared representation [17]. This paradigm is particularly applicable when there is a primary task and one or more secondary tasks. Because MTL exploits the potential commonality among the tasks, it is generally accepted that it can outperform single-task learning models. Several studies reported great results using MTL for automated speech recognition [17] and natural language processing [18]. When MTL is used with a DNN, the main and the secondary tasks share the same hidden representations. In other words, MTL is configured with two Deep Neural Network (DNN) classifiers sharing the same input and hidden neurons.

In this study, the main task is to classify the primary cancer site while the secondary task is to classify the primary site's laterality. Both tasks are solved simultaneously leveraging the same n-gram feature representation extracted from the cancer pathology reports. Conveniently, the MTL framework is trained with the same optimization techniques as a single-task DNN. Therefore, its implementation is straightforward. The first DNN is trained one epoch using the training set of the first task, and then the second DNN is trained one epoch using the training set of the second task.

3.5 Performance Evaluation

To compare the performance of the various DNN classifiers we performed 10-fold cross-validation experiments, where the sample data is randomly partitioned into 10 equal sized subsamples, so that each subsample was retained as the validation for testing the model while the remaining 9 subsamples were used as training the model.

Both micro and macro F-scores were used for evaluation of the methods, as typical done with biomedical NLP applications. For given precision, recall, and F-score of the $class_i$,

$$Precision_{class_i} = \frac{TP_{class_i}}{TP_{class_i} + FP_{class_i}}, Recall_{class_i} = \frac{TP_{class_i}}{TP_{class_i} + FN_{class_i}}$$

$$F_{class_i} = \frac{2 \cdot Precision_{class_i} \cdot Recall_{class_i}}{Precision_{class_i} + Recall_{class_i}}$$

where TP_{class_i}, FP_{class_i}, and FN_{class_i} are true positives, false positives, and false negatives of the $class_i$, macro F-score is defined as

$$\text{Macro-F} = \frac{1}{N} \cdot \sum_{i=1}^{N} F_{class_i}.$$

On the other hand, micro precision, recall, and F-scores are defined as follows:

$$Precision' = \frac{\sum_{i=1}^{N} TP_{class_i}}{\sum_{i=1}^{N} (TP_{class_i} + FP_{class_i})}, Recall' = \frac{\sum_{i=1}^{N} TP_{class_i}}{\sum_{i=1}^{N} (TP_{class_i} + FN_{class_i})}$$

$$\text{Micro-F} = \frac{2 \cdot Precision' \cdot Recall'}{Precision' + Recall'}.$$

4 Results

The MTL-DNN algorithm was implemented with the PDNN [17] software package running on Theano 0.8.2 [19] deep learning library on Python 2.7.11 environment. The network architecture shared across tasks was 400 × 400 × 100 nodes, and task-specific individual networks were specified by 100 × 50 × 20 × 2 nodes. The sigmoid activation function was selected. The learning rate was 0.08.

For comparison purposes, we divided the primary and secondary tasks and repeated the study using individual single-task Deep Neural Networks with the same network configuration and learning rule used to train the MTL-DNN. In addition, we implemented single task classifiers using Logistic Regression and Naïve Bayes classifiers using the Weka [20] software package.

Results are shown in Table 3.

The classification performance for the primary cancer site classification task was almost identical and nearly perfect for all classifiers. Notable performance differences

Table 3. Macro and micro F-scores of the two classification tasks, primary site category and laterality, based on Naïve Bayes, Logistic Regression, Deep Neural Networks, and Multi-Task Learning of DNN.

Task	Classifier	Macro F-score	Micro F-score
Primary site	Naïve Bayes	0.987	0.988
	Logistic Regression	0.997	0.998
	DNN	0.997	0.998
	MTL	**0.998**	**0.998**
Laterality	Naïve Bayes	0.648	0.654
	Logistic Regression	0.899	0.899
	DNN	0.929	0.929
	MTL	**0.948**	**0.948**

were observed for the secondary, laterality classification task. The MTL classifier achieved substantially higher macro and micro F-scores than the other classification methods suggesting that MTL can effectively leverage the commonality between the two tasks using a shared representation.

Further analysis of the laterality classification was performed using the area under the Receiver Operating Characteristic (ROC) curve as the performance metric (Fig. 2). ROC curve fitting and calculation of the Area Under ROC Curve (AUC) were based on Wilcoxon statistics and obtained by JROCFIT [21] software.

AUC of MTL was 0.9762 (±0.0045), which is dramatically higher than the AUCs of Naïve Bayes, 0.7061 (±0.0149), and Logistic Regression, 0.9102 (±0.0087). However, it was not significantly higher than the AUC of DNN, 0.9731 (±0.0048).

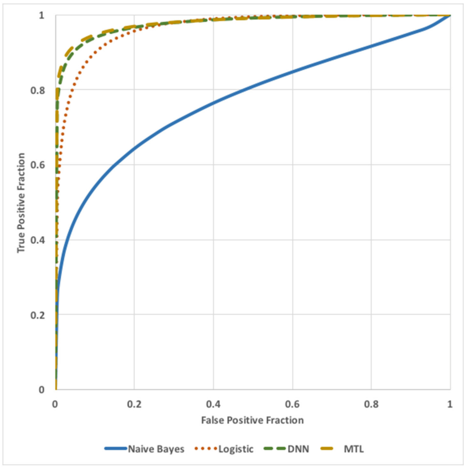


Fig. 2. Receiver Operating Characteristic (ROC) curves of the laterality classification task. Area Under ROC Curves (AUC) of the classifiers based on Wilcoxon statistics are 0.7061 for Naïve Bayes, 0.9102 for Logistic Regression, 0.9731 for DNN, and 0.9762 for MTL.

To evaluate scalability, we performed an additional experiment by augmenting the original pathology dataset. By oversampling the set, we generated an augmented dataset of 100,000 pathology reports. Then, we evaluated DNN training scalability for the primary site laterality task. First, we selected 2,000 n-grams for feature representation as described in the previous section. Then, DNN training experiments were performed on the TITAN supercomputer at the Oak Ridge Leadership Computing Facility. TITAN features 18,688 compute nodes, where each node contains an advanced 16-core 2.2 GHz AMD Opteron processor and an NVIDIA Tesla K20 accelerator. The DNN was implemented by Torch [22]. We tested Downpour SGD [23], an asynchronous stochastic gradient descent algorithm. We analyzed scalability by arbitrarily choosing a fixed test set accuracy (92%), and measured the training time each method took to reach the pre-selected test accuracy. Scalability test results are illustrated in Fig. 3.

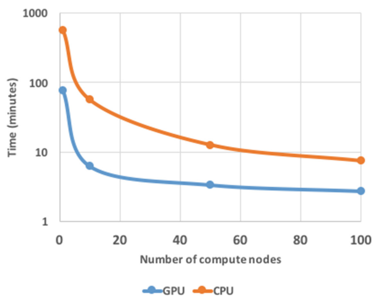


Fig. 3. Time to reach a fixed classification accuracy (92 %) for different architecture and number of computing nodes.

5 Discussion

This study presents a novel application of deep learning for information extraction from cancer pathology reports. Using two different but closely related information extraction tasks, the study showed that deep learning outperforms traditional classifiers for the more challenging task. Furthermore, the study demonstrated the value of multi-task learning for boosting further classification performance. Although the improvement was not statistically significant (mainly due to the relatively small size of the available dataset), MTL appears to be a promising approach for analysis of pathology reports, which tend to include several related pieces of clinical information. Compared to other published studies on the topic, our information extraction algorithms perform competitively, with F-scores on the high end of the reported spectrum (0.58 to 0.97). However, no further conclusions can be drawn since in depth analysis requires implementation on the same clinical text corpus. Our future studies will focus on expanding the dataset size as well as extending the scope of the information extraction task to multiple cancer types and multiple clinical attributes.

Acknowledgements. This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

The study was supported by the Laboratory Directed Research and Development (LDRD) program of Oak Ridge National Laboratory, under LDRD projects No. 7417 and No. 8231.

References

1. Greenhalgh, T., Hurwitz, B.: Narrative based medicine: why study narrative. Br. Med. J. **318** (7175), 48–50 (1999)
2. Stein, H.D., Nadkarni, P., Erdos, J., Miller, P.L.: Exploring the degree of concordance of coded and textual data in answering clinical queries from a clinical data repository. JAMIA **7** (1), 42–54 (2000)
3. Martinez, D., Li, Y.: Information extraction from pathology reports in a hospital setting. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, pp. 1877–1882 (2011)
4. Jouhet, V., Defossez, G., Burgun, A., Le Beux, P., Levillain, P., Ingrand, P., Claveau, V.: Automated classification of free-text pathology reports for registration of incident cases of cancer. Methods Inf. Med. **51**(3), 242 (2012)
5. Kavuluru, R., Hands, I., Durbin, E.B., Witt, L.: Automatic extraction of ICD-O-3 primary sites from cancer pathology reports. In: Clinical Research Informatics AMIA Symposium (2013, forthcoming)
6. Bengio, Y., Ducharme, R., Vincent, P., Janvin, C.: A neural probabilistic language model. The. J. Mach. Learn. Res. **3**, 1137–1155 (2003)
7. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning, pp. 160–167. ACM (2008)
8. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
9. Choi, E., Schuetz, A., Stewart, W.F., Sun, J.: Medical concept representation learning from electronic health records and its application on heart failure prediction. arXiv preprint arXiv:1602.03686. (2016)
10. Miotto, R., Li, L., Dudley, J.T.: Deep Learning to predict patient future diseases from the electronic health records. In: Advances in Information Retrieval, pp. 768–774. Springer International Publishing (2016)
11. Caruana, R.: Multitask Learning. Springer, New York (1998)
12. Surveillance, Epidemiology, and End Results (SEER) Program (www.seer.cancer.gov) Research Data (1973–2013), National Cancer Institute, DCCPS, Surveillance Research Program, Surveillance Systems Branch, released April 2016
13. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw-Hill, Inc., New York (1986)

14. Aizawa, A.: An information-theoretic perspective of TF–IDF measures. Inf. Process. Manage. **39**(1), 45–65 (2003)
15. Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.R., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., Kingsbury, B.: Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Sig. Process. Mag. **29**(6), 82–97 (2012)
16. Bottou, L.: Online algorithms and stochastic approximations. In: Saad, D. (ed.) Online Learning and Neural Networks. Cambridge University Press, Cambridge. (1998)
17. Miao, Y.: Kaldi+PDNN: building DNN-based ASR systems with Kaldi and PDNN. arXiv preprint arXiv:1401.6984. (2014)
18. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning, pp. 160–167. ACM (2008)
19. Team, T.T.D., et al.: Theano: A Python framework for fast computation of mathematical expressions. arXiv preprint arXiv:1605.02688 (2016)
20. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. ACM SIGKDD Explor. Newslett. **11**(1), 10–18 (2009)
21. ROC Analysis: Web-based Calculator for ROC Curves. http://www.rad.jhmi.edu/jeng/javarad/roc/JROCFITi.html
22. Collobert, R., Kavukcuoglu, K., Farabet, C.: Torch7: A matlab-like environment for machine learning. In: BigLearn, NIPS Workshop, No. EPFL-CONF-192376 (2011)
23. Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., Ranzato, M., Senior, A., Tucker, P., Yang, K., Ng, A.Y.: Large scale distributed deep networks. In: Advances in Neural Information Processing Systems, pp. 1223–1231 (2012)

A Graph-Based Big Data Model for Wireless Multimedia Sensor Networks

Cihan Küçükkeçeci[1(✉)] and Adnan Yazıcı[2(✉)]

[1] AYESAŞ, ODTÜ Teknokent, Ankara, Turkey
cihank@ayesas.com
[2] Department of Computer Engineering,
Middle East Technical University, Ankara, Turkey
yazici@ceng.metu.edu.tr

Abstract. Wireless multimedia sensor networks are of interest to researchers from different disciplines and many studies have been proposed in a wide variety of application domains, such as military surveillance systems, environmental monitoring, fault monitoring and distributed smart cameras in the last decade. In a wireless sensor network, a large number of sensors can be deployed to monitor target areas and autonomously collect sensor data. This produces a large amount of raw data that needs to be stored, processed, and analyzed.

In this paper, we propose a graph-based big data model for simulating multimedia wireless sensor networks. The big sensor data is stored in a graph database for the purpose of advanced analytics like statistics, data mining, and prediction. A prototype implementation of the proposed model has been developed and a number of experiments have been done for measuring the accuracy and efficiency of our solution. In addition, we present a case study using the military surveillance domain with a number of complex experimental queries by using our prototype. The experimental results show that our proposed multimedia wireless sensor network model is efficient and applicable in large-scale real life applications.

1 Introduction

A wireless multimedia sensor network (WMSN) is a distributed wireless network that consists of a set of multimedia sensor nodes which are connected to each other or connected to leading gateways. Nowadays, smart devices such as mobile phones, smart televisions and smart watches are equipped with sensors and network connections. Hence, with the advances in wireless communication technologies, multimedia sensor networks are expected to be one of the major components in the Internet of things (IoT).

A typical application for a WMSN would be a surveillance system or a monitoring system. Smart city surveillance cameras with 7/24 recording, or one million sensor nodes reporting meteorological data produce data in various formats as video, audio, and text [4]. All those huge amount of structured

© Springer International Publishing AG 2017
P. Angelov et al. (eds.), *Advances in Big Data*, Advances in Intelligent Systems and Computing 529, DOI 10.1007/978-3-319-47898-2_22

or unstructured data is considered as big data, which is defined by a number of Vs; *Volume, Velocity, Variety, Veracity, and Value.* Min et al. [3] present a comprehensive survey of big data and they identify "defining the structural model of big data" as a fundamental problem. Fusing and analyzing big data are challenging tasks and there are many research studies that are related to big data from different points of view in recent years. Many researchers state that relational database management systems (RDBMS) are inadequate for big data and as a solution, NoSQL databases are mostly utilized [1,2,16]. There are four main types of NoSQL databases, which are key-value store (e.g. Amazon's Simple DB), big table (e.g. Apache Cassandra), document-oriented (e.g. MongoDB) store and graph based model (e.g. Neo4j) [15].

Graph databases consist of nodes and edges (relations between nodes) which store data as properties. Graph databases are very efficient and convenient to handle social networks, fraud detection, graph-based operations, real-time recommendations and hierarchical relations. While storage of the big data is an important task, processing the streaming data and taking action for mission-critical applications are also crucial. Arkady et al. [6] conclude that analyzing and extracting the valuable data from dirty raw data is an important research topic. In order to process that kind of data, we need to identify the data flow.

In this paper, we propose a graph based big data model in a generic manner for handling multimedia wireless sensor network data. Graph based data model suits well for advanced analytics like graph mining, and prediction using the complex relationships between data [19]. For that purpose, big sensor data is stored in a graph database in such a way that proposed graph model represents both the sensor network topology and the data flow among the nodes. The applicability of our solution is illustrated with a prototype implementation and a case study in the military surveillance domain. A number of experiments have been done for measuring the accuracy and efficiency of our solution. Simulation results show that our proposed multimedia wireless sensor network model is applicable in large-scale real life scenarios.

The contribution of this work is to store the multimedia sensor network data in a well-defined graph-based big data model for analyzing, filtering, aggregating and correlating by using an open-source graph database, and simulating multimedia wireless sensor networks to run a number of complex experimental queries. Although there have been some related studies in literature about surveillance systems in the big data context, to the best of our knowledge, there has not been an applicable graph database on a graph-based big data model for WMSNs yet.

This paper is organized as follows: next section provides related work. Then we propose our graph-based big model in Sect. 3. Implementation of our model is presented in Sect. 4. Section 5 illustrates a case study in the military surveillance domain and Sect. 6 presents the experimental results and evaluations. Finally, conclusions are drawn in Sect. 7.

2 Related Work

Over the years, various methods are used for wireless sensor network data representation and management [12–14,17]. Yang et al. [10] propose a hybrid data model to store their wireless sensor network data. NoSQL part of the hybrid model is stored with a key-value structure to provide higher scalability and better performance. Christine et al. [18] discuss big data with spatial data received from wireless sensors using real life scenarios.

Renzo et al. [9] present a survey paper on graph database models. They compare graph database models to other database models like a relational model and then compare the representative graph database models. Mark et al. [8] introduce a graph based model which is called HyperNode. They also define a language HNQL (HyperNode Query Language) to query and update their model. Arati et al. focus on the information retrieval from sensor networks and propose hybrid protocol which is called APTEEN [11]. They have noticed that achieving an efficient model needs to replace the conventional flat topology with a graph based model.

PipeNet [20] is multi layered wireless sensor network application on pipeline scenario which is the similar multi layer architecture as we propose but in another domain. They have developed the system to analyze the collected multi-modal data for detection of the leaks. Another research about WSN related to surveillance domain is a survey paper written by Felemban [21]. His survey research enlists the literature for experimenting work done in border surveillance and intrusion detection using the technology of WSN. Our research differs from the existing works by employing a graph based approach for surveillance domain and focusing on the simulation of the big data.

3 The Graph-Based Big Data Model

Our graph-based big data model is built on the multimedia sensor networks topology. There is a sink node in the base station and there are a number of clusters connected to the sink. Each cluster consists of a gateway, which can be called the cluster head, and a set of sensor nodes.

The data flow occurs from the sensor nodes to gateways and from gateways to the gateways and sink. Each sensor node holds a set of data sensed by the sensors and camera of the node. Sensor nodes apply algorithms like correlation and transformation on the raw data, which is called first level fusion. The sensor fused data are reported to the leading gateway by all of its connected sensor nodes.

The gateway waits for all sensor nodes to report. When all reports are ready, the gateway applies an aggregation or filtering on the received data. The second level fusion is done at this point and the output of the fusion is like a summary of that cluster. The gateway fused data are forwarded to the Sink node for a final decision. Similar to the second-level fusion, the sink waits for all gateways' fused data. By applying some patterns to detect anomalies or other kinds of analysis are done at the third level fusion. The output of the last level fusion is an action like triggering an alarm or a notification message to another system.

A graph database is our storage environment and data model is designed as a graph-based big data model in parallel to our NoSQL database selection. Data stored in graph database is used for further analytical processes like tracking and event detection.

4 Model Implementation over Graph Database

Our research includes applying the currently available databases to our graph-based big data model. Salem et al. [7] compare a set of databases like Cougar and TinyDB. Li-Yung Ho et al. [5] propose a distributed graph database based on an open-source graph database which is called Neo4j. The options are limited if you are looking for a graph database. Neo4j, Titan and OrientDB are featured open-source graph databases.

Neo4j is a well-known graph database and used by many researchers. In addition, Neo4j is relatively easier to be used rapidly by developing some small pieces of code. Spring Framework support is really helpful to put things together very fast. Titan is another open-source option for a graph database but its development is stopped and discontinued in early 2015. Therefore we did not prefer to utilize Titan. OrientDB is another open-source graph database which is not as popular as Neo4j for now but has many advantages over it. OrientDB's official website includes the comparison of OrientDB and Neo4j databases. From all those compared features, Multi-Master Replication and SQL and operational database are the important capabilities for us.

Nodes (vertices) and relations (edges) are defined in graph databases to store data. Compared to the traditional RDBMS approach, every row in a table is replaced with a node and its properties. There are edges to represent cross-table references. Node types are; Sink, Gateway, SensorNode, SensorRawData, SensorFusedData, GatewayFusedData and SinkFusedData. Edge types are; Lead,

Algorithm 1. Sample First Level Fusion Algorithm

```
 1: procedure FIRSTLEVELFUSION($pir, seismic, acoustic$)    ▷ Sensed raw data values
 2:     if $pir$ = true and $seismic \geq$ THRESHOLD and $acoustic \geq$ THRESHOLD
          then
 3:         $video \leftarrow startVideoRecording()$.
 4:         $frame \leftarrow selectFrames(video)$.
 5:         $lowLevelFeaturesFrame \leftarrow findLowLevelFeatures(frame)$.
 6:         $foreground \leftarrow selectForeground(frame)$.
 7:         $lowLevelFeaturesForeground \leftarrow findLowLevelFeatures(foreground)$.
 8:         $silhouette \leftarrow extractSilhouette(foreground)$.
 9:         $weight \leftarrow calculateWeight()$
10:     end if
11: end procedure
```

Collect, LastCollected, Next, Fusion, FusedBy, LastFusion, Reported and Forwarded. The edge types are defined for the usage between specific nodes.

Sample algorithms for all three levels of fusion are given in Algorithms 1, 2 and 3 which provides the proof of concept execution of all phases of the simulated environment.

Algorithm 2. Sample Second Level Fusion Algorithm

```
1: procedure SECONDLEVELFUSION        ▷ all connected sensor nodes reported fusion data
2:     $fusedData[] \leftarrow \{\}$.
3:     $normalizedData[] \leftarrow \{\}$.
4:     for all concepts reported by sensor nodes do
5:         $diff \leftarrow current.acoustic - previous.acoustic$.
6:         $diffRate \leftarrow current.acoustic *$ THRESHOLD-DUPLICATE/100.
7:         if $diff < diffRate$ then
8:             mark concept as duplicate and drop.
9:         else
10:            $fusedData[] \leftarrow concept$.
11:        end if
12:    end for
13:    for all concepts $fusedData$ do
14:        $diff \leftarrow current.acoustic - previous.acoustic$.
15:        $diffRate \leftarrow current.acoustic *$ THRESHOLD-NORMALIZE/100.
16:        if $diff < diffRate$ then
17:            mark concept as normalized.
18:        else
19:            $normalizedData[] \leftarrow concept$.
20:        end if
21:    end for
22: end procedure
```

Algorithm 3. Sample Third Level Fusion Algorithm

```
1: procedure THIRDLEVELFUSION        ▷ all gateways forwarded detected concepts
2:     $normalizedData[] \leftarrow \{\}$.
3:     for all forwarded concepts do
4:         $diff \leftarrow current.acoustic - previous.acoustic$.
5:         $diffRate \leftarrow current.acoustic *$ THRESHOLD-NORMALIZE/100.
6:         if $diff < diffRate$ then
7:             mark concept as normalized.
8:         else
9:             $normalizedData[] \leftarrow concept$.
10:        end if
11:    end for
12:    $notifyOperator(maxWeightedConcept)$.
13: end procedure
```

5 A Case Study in the Military Surveillance Domain

Surveillance systems need robust and scalable infrastructure. To achieve that, all data flow and data itself are needed to be analyzed and modeled. Data flow is provided by a set of sensors and video cameras. Assume that, we cluster the sensors and cameras according to the districts and each cluster forwards sensed data to the HQ (Head Quarter) which is the operation center. At the HQ, an alarm is triggered, or a notification is sent to the officers to early detection of violence.

Sensor types can be seismic, acoustic and PIR (Passive Infrared) which are scalar sensors. In addition to them, video cameras or thermal cameras can be added to critical locations. As the default, cameras are switched off. According to the sensed information from scalar sensors, predefined conditions can be extracted using rule based approaches. The motion or environmental change may be detected and interpreted to activate the camera by providing a rough prediction of the moving object.

We categorize the moving object as; Animal, Human, and Vehicle. The data collected from scalar sensors are analyzed to guess the category of the objects according to predefined thresholds (Table 1).

Table 1. Sample thresholds to identify objects

Object type	PIR	Seismic (Hz)	Acoustic (dB)
Animal	True	5 - 20	5 - 30
Human	True	21 - 55	31 - 50
Vehicle	True	>35	>50

6 Experimental Work

We setup a test environment to make some experiments on our graph model. Test environment hardware specifications are;

- Intel i7-4710HQ Quad Core CPU
- 16 GB DDR3 RAM
- 240 GB SSD storage
- 4 GB NVIDIA 860GTX GPU

Test environment has three database systems which are OrientDB v2.1.2 (Graph database), Neo4j v2.3.2 (Graph database) and MySQL v5.7.1 (Relational database). We have simulated a sensor network with synthetic raw data for all three databases. Sensor nodes are placed in a square shaped area. Gateways are located in the center of each group of sensor nodes. The sink node is placed in the center of the whole area. Total count for specific node types are;

- 1 Sink
- 2,500 Gateway (Each gateway leads 10,000 sensor nodes)
- 25,000,000 Sensor Node

6.1 Comparison to Relational Data Model

A number of experiments are done on all three databases installed in our test environment. We have run many queries on both our graph data model and relational data model. Below some of the queries are randomly selected as sample queries and Table 2 shows the performance test results of the experiment.

1. **Detected Concepts Query:** This query finds the specific type of objects with the highest probability of its detection time and location.
   OrientDB Query:
```
SELECT concept, weight, fusionDate, out("fusedby").indexX, out("fusedby").
    indexY FROM fuseddata WHERE weight>0.90 AND concept = "Vehicle" ORDER
    BY fusionDate
```
   Neo4j Query:
```
MATCH (b:fuseddata)-[:fusedby]->(sd:sensornode) WHERE b.concept = "Human"
    AND b.weight>0.90 RETURN b.concept, b.weight, b.fusionDate, sd.indexX,
    sd.indexY
```
   Relational SQL Query:
```
SELECT b.concept, b.weight, a.fusion_Date, sd.indexx, sd.indexy FROM
    sinkfuseddata a, gatewayfuseddata g, sensorfuseddata b, sensornode sd
    WHERE a.concept = 'Vehicle' AND a.weight>0.90 AND a.id = g.fusion AND
    g.id = b.fusion AND b.fusedby = sd.id ORDER BY a.fusion_Date ASC
```

2. **Explosion Videos Query:** This query finds the possible explosions by identifying continued high volume around the surveillance area with their recorded video paths and video duration. The value bigger than 15 is assumed to be a high volume sound.
   OrientDB Query:
```
SELECT in("collect").name[0], in("collect").indexX[0], in("collect").indexY
    [0], acoustic, out("video").videoPath[0], out("video").videoDurationSec
    [0] FROM sensorrawdata WHERE acoustic>15 AND out("next").acoustic[0]>15
     AND out("next").out("next").acoustic[0]>15 ORDER BY name
```
   Neo4j Query:
```
MATCH (sn:sensornode)-[:collect]->(sa:sensorrawdata)-[:next]->(sb:
    sensorrawdata)-[:next]->(sc:sensorrawdata)-[:video]->(sv:
    sensorrawvideodata) WHERE sa.acoustic>15 AND sb.acoustic>15  AND sc.
    acoustic>15 RETURN sn.name, sa.acoustic, sn.indexX, sn.indexY, sv.
    videoPath, sv.videoDurationSec ORDER BY sn.name
```
   Relational SQL Query:
```
SELECT s.name, s.indexx, s.indexy, ra.collectDate, ra.acoustic, v.
    video_path, v.duration FROM sensornode s, sensorrawdata ra,
    sensorrawdata rb, sensorrawdata rc, sensorrawvideodata v WHERE ra.
    acoustic>15 AND rb.acoustic>15 AND rc.acoustic>15 AND s.id = ra.
    sensornode_id and ra.id = rb.next_id and rb.id = rc.next_id and rc.
    video_id = v.id ORDER BY s.name ASC
```

3. **Distance of Nodes Recursive Query:** This query finds the detected "Human" typed objects with high accuracy and calculates the distance of the sensor node to the sink node.
   OrientDB Query:
```
SELECT $nodeId, out("fusedby").name[0], fusionDate, $deep.count FROM
    fuseddata LET $nodeId = out("fusedby").@rid, $deep = (SELECT COUNT(*)
    FROM (TRAVERSE in('lead') FROM $nodeId)) WHERE concept = "Human" AND
    weight>0.9
```

Neo4j Query:

```
MATCH p=(a:sink)-[:lead*]->(b:gateway) WITH b.name as gname, length(p) AS
    depth MATCH (sfd:fuseddata)-[:fusedby]->(sn:sensornode)<-[:lead]-(g:
    gateway) WHERE sfd.concept = "Human" AND sfd.weight>0.9 AND
    g.name = gname RETURN gname, sn.name, sfd.fusionDate, depth
```

Relational SQL Query:

```
WITH RECURSIVE search_graph(id, name, lead, depth) AS (SELECT g.id, g.name,
     g.lead, 0 FROM gateway g WHERE g.lead is null UNION ALL SELECT g.id, g
    .name, g.lead, sg.depth + 1 FROM gateway g, search_graph sg WHERE g.
    lead = sg.id) SELECT s.name, s2.name, s2.indexX, s2.indexY, s.depth
    FROM search_graph s inner join (select distinct sn.id, sn.name, sn.
    indexX, sn.indexY, sn.lead FROM sinkfuseddata sfd, gatewayfuseddata gfd
    , sensorfuseddata srfd, sensornode sn WHERE srfd.fusion = gfd.id AND
    gfd.fusion = sfd.id AND srfd.fusedby = sn.id AND sfd.concept = 'Human'
    AND sfd.weight>0.9) s2 ON s2.lead = s.id
```

Table 2 shows the performance results of our example queries. For the first query, OrientDB beats Neo4j and graph model is better than the relational model. For the second query, Neo4j performs better than OrientDB and the graph model is again better than the relational one. The last query is to test the recursive SQL like query. The graph based model is much better than the relational model. But Neo4j fails for this query, maybe there is other functions of the Cypher (the query language of Neo4j).

Table 2. Test results

Query	OrientDB (ms)	Neo4j (ms)	MySQL (ms)
Detected concepts	209	618	938
Explosion videos	355	145	422
Distance of nodes	4,293	79,812	36,469

6.2 Doubling Sensed Raw Data Size

An OrientDB graph database is selected for the execution of experiments. The previous experiments were applied on generated synthetic data of one month where each sensor node can sense data with 5 min of period. Now we increase the simulation duration from 1 month to 5 months step by step and diagnosed the query performance.

Figure 1 shows the performance of queries in terms of “query execution time”. The results are not bad as the increase ratio of the “query execution time” is better than linear time which is fairly well compared to the doubled data size.

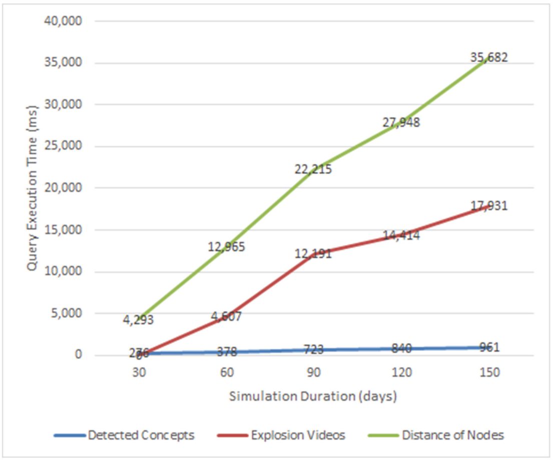


Fig. 1. OrientDB query execution time/simulation duration chart

7 Conclusions

We have developed a graph-based big data model to represent the multimedia sensor networks. We have implemented our proposed graph data model and simulated multimedia wireless sensor networks with synthetic data to generate big multimedia sensor data. As an application area, we have selected the surveillance systems. The sensor data retrieved from the network and the video data streamed from cameras are treated like big data. The storage environment of aforesaid big data is selected as a graph database which suits well to our graph model.

As our performance test results show that the graph model performs much better than the relational model. In order to decide on the graph-based big database for our implementation, we have compared two very well-known graph databases, Neo4j and OrientDB. OrientDB generally performs better than Neo4j in our experiments. On the other hand, Neo4j's master-slave approach is not scalable because only one single node is active at a time. From the point of big data view, it is not suitable for us.

Our graph-based big data model successfully survives even with millions of data nodes. We have tested many complex query scenarios on our synthetic data and millions of data can be efficiently queried, in a few seconds. As our ongoing research work, our big data graph database will be used for advanced analytics, more complex database queries, object tracking, and early detection of events.

Acknowledgements. This work is supported in part by a research Grant from TÜBİTAK with Grant No. 114R082. We thank to each of the researchers of CEng Multimedia Database Lab. for their very valuable contributions. The first author would also like to thank AYESAŞ for providing financial support.

References

1. Moniruzzaman, A.B., Hossain, S.A.: Nosql database: New era of databases for big data analytics-classification, characteristics and comparison. arXiv preprint arXiv:1307.0191 (2013)
2. Chad, V., Macias, M., Zhao, Z., Nan, X., Chen, Y., Wilkins, D.: A comparison of a graph database, a relational database: a data provenance perspective. In: Proceedings of the 48th Annual Southeast Regional Conference, p. 42. ACM (2010)
3. Chen, M., Man, S., Liu, Y.: Big data: a survey. Mobile Netw. Appl. **19**, 171–209 (2014)
4. Perera, C., Zaslavsky, A., Christen, P., Georgakopoulos, D.: Sensing as a service model for smart cities supported by Internet of Things. Trans. Emerg. Telecommun. Technol. **25**(1), 81–93 (2014)
5. Ho, L-Y., Wu, J.-J., Liu, P.: Distributed graph database for large-scale social computing. In: 2012 IEEE 5th International Conference on Cloud Computing (CLOUD), pp. 455–462. IEEE (2012)
6. Zaslavsky, A., Perera, C., Georgakopoulos, D.: Sensing as a service, big data, arXiv preprint arXiv:1301.0159 (2013)
7. Hadim, S., Nader, M.: Middleware: Middleware challenges and approaches for wireless sensor networks. IEEE Distrib. Syst. Online **3**, (2006)
8. Levene, M., Loizou, G.: A graph-based data model, its ramifications. IEEE Trans. Knowl. Data Eng. **7**(5), 809–823 (2011)
9. Angles, R., Gutierrez, C.: Survey of graph database models. ACM Comput. Surv. (CSUR) **40**(1), 1–39 (2008)
10. Li, Y., Wu, C., Guo, L., Lee, C.-H., Guo, Y.: Wiki-health: a big data platform for health. In: Cloud Computing Applications for Quality Health Care Delivery, p. 59 (2014)
11. Manjeshwar, A., Agrawal, D.P.: APTEEN: a hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. In: International Parallel and Distributed Processing Symposium, vol. 2. IEEE Computer Society (2002)
12. Hu, C., Liu, Y., Chen, L.: Semantic link network based model for organizing multimedia big data. IEEE Trans. Emerg. Topics Comput., 1 (2011)
13. Korpeoglu, B., Yazici, A., Korpeoglu, I., George, R.: A new approach for information processing in wireless sensor network. In: Proceedings of the 22nd International Conference on Data Engineering Workshops, pp. 34–34. IEEE (2006)
14. Zhang, P., Yan, Z., Sun, H.: A novel architecture based on cloud computing for wireless sensor network. In: Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013), pp. 472–475 (2013)
15. Jing, H., Haihong, E., Le, G., Du, J.: Survey on NoSQL database. In: 2011 6th International Conference on Pervasive Computing and Applications (ICPCA), pp. 363–366. IEEE (2011)
16. Xu, Z., Liu, Y., Mei, L., Hu, C., Chen, L.: Semantic based representing and organizing surveillance big data using video structural description technology. J. Syst. Softw. **102**, 217–225 (2015)
17. Diallo, O., Rodrigues, J.J., Sene, M.: Real-time data management on wireless sensor networks: a survey. J. Netw. Comput. Appl. **35**(3), 1013–1021 (2012)
18. Jardak, C., Mahonen, P., Riihijärvi, J.: Spatial big data and wireless networks: experiences, applications, and research challenges. IEEE Netw. **28**(4), 26–31 (2014)

19. Simmen, D., Schnaitter, K., Davis, J., He, Y., Lohariwala, S., Mysore, A., Shenoi, V., Tan, M., Xiao, Y.: Large-scale graph analytics in Aster 6: bringing context to big data discovery. Proc. VLDB Endowment **7**(13), 1405–1416 (2014)
20. Stoianov, I., Nachman, L., Madden, S., Tokmouline, T.: PIPENET: a wireless sensor network for pipeline monitoring. In: 2007 6th International Symposium on Information Processing in Sensor Networks, pp. 264–273. IEEE (2007)
21. Felemban, E.: Advanced border intrusion detection and surveillance using wireless sensor network technology. Int. J. Commun. Netw. Syst. Sci. **6**(5), 251 (2013)

CCM: Controlling the Change Magnitude in High Dimensional Data

Cesare Alippi[1,2], Giacomo Boracchi[1], and Diego Carrera[1(✉)]

[1] Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy
{cesare.alippi,giacomo.boracchi,diego.carrera}@polimi.it
[2] Università della Svizzera Italiana, Lugano, Switzerland

Abstract. The effectiveness of change-detection algorithms is often assessed on real-world datasets by injecting synthetically generated changes. Typically, the magnitude of the introduced changes is not controlled, and most of experimental practices lead to results that are difficult to reproduce and compare with. This problem becomes particularly relevant when the data-dimension scales, as it happens in big data applications.

To enable a fair comparison among change-detection algorithms, we have designed "Controlling Change Magnitude" (CCM), a rigorous method to introduce changes in multivariate datasets. In particular, we measure the change magnitude as the symmetric Kullback-Leibler divergence between the pre- and post-change distributions, and introduce changes by applying a roto-translation directly to the data. We present an algorithm to identify the parameters yielding the desired change magnitude, and analytically prove its convergence. Our experiments show the effectiveness of the proposed method and the limitations of tests run on high-dimensional datasets when changes are injected following traditional approaches. The MATLAB framework implementing the proposed method is made publicly available for download.

1 Introduction

Change detection is an important problem in machine learning and data mining. Change-detection tests [5,12,13,17] and outlier/anomaly detectors [9,16] provide precious information for adaptation and data-analysis purposes, like revealing unforeseen evolutions of the process generating the data or anomalous events. One of the issues to be considered when using change-detection algorithms in big data applications concerns the assessment of detection performance, which requires special care when the data dimension scales.

To get stable performance measures, change-detection algorithms need to be executed on a large number of datasets, and to study the detection performance when data dimension scales, datasets having different number of components are needed. Moreover, all these datasets should be affected by changes at known locations. Unfortunately, there are not many real-world datasets exhibiting real changes that satisfy these requirements. As such, experiments typically involve few real-world datasets, and results have often to be interpreted by visual inspection, as in the tests on financial time series in [17]. While these experiments indicate whether the algorithm is successful or not, visual inspection does not provide a sound quantitative assessment of the performance, and is not viable when dimension increases.

In practice, experiments are often performed on either synthetically generated data or on real-world datasets that have been manipulated introducing changes at known locations. The first option consists in choosing two different distributions, ϕ_0 and ϕ_1, to generate the pre- and post-change data, respectively (see e.g. [3,17]). In online classification problems, changes (concept drifts [11]) can be also introduced by modifying the label assignment-criteria, rather than the distribution of the raw data [4,10,19]. Generating synthetic datasets is often the easiest option to arbitrarily increase the number of datasets to be tested, thus achieve the desired accuracy in the performance measures. However, synthetic data rarely represent realistic monitoring scenarios, as they follow quite simplistic distributions.

Manipulating real-world data is certainly more appealing, since this allows testing change-detection algorithms in more realistic monitoring conditions. However, special care should be taken when introducing changes, to make sure that their magnitude is in a reasonable range. Controlling the change magnitude is important to make experiments reproducible, and to fairly compare the results among different datasets. In particular, controlling the change magnitude is necessary to study how change-detection performance vary when data dimension scales [2]. Most of the papers in the literature resort to introducing arbitrary shifts/scaling [3], swapping few components of the original dataset [13], or mixing different datasets [7]. Unfortunately, these practices yield changes having magnitude that cannot be easily controlled as it depends on the data dimension and distribution.

We here describe "Controlling Change Magnitude" (CCM), a rigorous method to introduce changes having a controlled magnitude. The method is flexible and allows to operate on real-world datasets having arbitrary dimensions. In particular, we approximate ϕ_0 by a Gaussian mixture (GM), and measure the change magnitude by the symmetric Kullback-Leibler (sKL) divergence between ϕ_0 and ϕ_1. We introduce changes by roto-transalating the data, and the transformation parameters are estimated through an iterative algorithm (Sect. 3) which yields constant change magnitude.

This method was briefly introduced in [2], where it was used to generate datastreams for investigating the detectability-loss, a problem that affects change-detection algorithms monitoring the log-likelihood w.r.t. ϕ_0. We here provide a detailed description of the method and proof the convergence of algorithms used for estimating the roto-translation parameters (Sect. 4). We also illustrate the main functionalities of our software framework[1] (Sect. 5) that can be used to prepare datastreams for change-detection purposes from datsets of

[1] publicly available for download at: http://home.deib.polimi.it/carrerad.

popular machine learning repositories like the UCI one [14]. Finally (Sect. 6), we demonstrate the effectiveness of the proposed method, showing that changes introduced by other methods can yield considerably different magnitude where data dimension scales, as in big data scenario.

2 Problem Formulation

We consider a dataset S of stationary data, i.e. containing independent and identically distributed (i.i.d.) random vectors $\mathbf{s} \in \mathbb{R}^d$. We assume that S is generated by a probability density function (pdf) ϕ_0 that is possibly unknown. We want to generate a datastream $X = \{\mathbf{x}(t),\ t = 1, \dots\}$ affected by a change at time $t = \tau$ as

$$\mathbf{x}(t) \sim \begin{cases} \phi_0 & t < \tau \\ \phi_1 & t \geq \tau \end{cases}, \text{ where } \phi_1(\mathbf{x}) := \phi_0(Q\mathbf{x} + \mathbf{v}), \tag{1}$$

where $\mathbf{v} \in \mathbb{R}^d$ and $Q \in \mathbb{R}^{d \times d}$ is an orthogonal matrix, i.e. $Q'Q = QQ' = I$. This change model allows us to assemble X by simply selecting $\mathbf{x}(t)$ from S when $t < \tau$, and roto-translating the remaining $\mathbf{s} \in S$ after the change point, i.e. $\mathbf{x}(t) = Q'(\mathbf{s} - \mathbf{v})$ for $t \geq \tau$. This is quite a general model which includes changes in the mean as well in the correlation among the data components.

Our goal is to define Q and $\mathbf{v}$ such that the change in (1) features a specific magnitude, which we measure by the symmetric Kullback-Leibler divergence (also known as Jeffreys' divergence) between ϕ_0 and ϕ_1

$$\begin{aligned} \mathrm{sKL}(\phi_0, \phi_1) &:= \mathrm{KL}(\phi_0, \phi_1) + \mathrm{KL}(\phi_1, \phi_0) \\ &= \int_{\mathbb{R}^d} \log \frac{\phi_0(\mathbf{x})}{\phi_1(\mathbf{x})} \phi_0(\mathbf{x}) d\mathbf{x} + \int_{\mathbb{R}^d} \log \frac{\phi_1(\mathbf{x})}{\phi_0(\mathbf{x})} \phi_1(\mathbf{x}) d\mathbf{x}. \end{aligned} \tag{2}$$

In practice, given ϕ_0 and the desired magnitude κ, we have to estimate $\mathbf{v} \in \mathbb{R}^d$ and $Q \in \mathbb{R}^{d \times d}$ such that

$$\mathrm{sKL}\left(\phi_0, \phi_1\right) = \mathrm{sKL}\left(\phi_0, \phi_0(Q \cdot + \mathbf{v})\right) = \kappa. \tag{3}$$

The symmetric Kullback-Leibler divergence is quite a natural choice in the change-detection scenario, as discussed in [2]. We observe that (2) is well posed if and only if the supports of ϕ_0 and ϕ_1 coincide. For this reason, we here assume that ϕ_0 is strictly positive on $\mathbb{R}^d$, and so ϕ_1.

3 Method Description

In this section we describe two algorithms that iteratively compute the rotation matrix Q and the translation vector $\mathbf{v}$ yielding $\mathrm{sKL}(\phi_0, \phi_1) \approx \kappa$. To compute $\mathrm{sKL}(\phi_0, \phi_1)$ in (2), both ϕ_0 and ϕ_1 are necessary; however, since ϕ_0 is typically unknown, we replace it with a GM estimate $\widetilde{\phi}_0$, which is computed on the whole

```
1.  Input: $\widetilde{\phi}_0$ and $\kappa$, the target value of $\mathrm{sKL}(\widetilde{\phi}_0, \phi_1)$
2.  Output: Initial roto-translation parameters $\boldsymbol{\theta}^{(0)}$, $P$, $\rho^{(0)}$, $\mathbf{u}$
3.  Set $\rho^{(0)} = 1$.
4.  repeat
5.      Randomly generate $m$ angles $\boldsymbol{\theta}^{(0)}$ in $[-\pi/2, \pi/2]^m$ and a unitary vector $\mathbf{u}$.
6.      Generate a matrix $A \in \mathbb{R}^{d\times d}$ with Gaussian entries.
7.      Set $P$ as the orthogonal matrix of the QR decomposition of $A$.
8.      Set $Q^{(0)}(\boldsymbol{\theta}^{(0)}, P) = PS(\boldsymbol{\theta}^{(0)})P'$ and $\mathbf{v}(\rho^{(0)}, \mathbf{u}) = \rho^{(0)}\mathbf{u}$.
9.      Compute $s^{(0)} = \mathrm{sKL}(\widetilde{\phi}_0, \phi_1)$, where $\phi_1 = \widetilde{\phi}_0(Q^{(0)} \cdot + \mathbf{v}^{(0)})$
10.     $\rho^{(0)} = 2\rho^{(0)}$.
11. until $s^{(0)} > \kappa$;
```

Algorithm 1. Definition of initialization parameters.

dataset S. We also adopt a suitable parametrization for Q and $\mathbf{v}$, that are randomly initialized as in Algorithm 1. These parameters are then adjusted using a bisection method, described in Algorithm 2, such that $\mathrm{sKL}(\phi_0, \phi_1)$ approaches the target value κ up to a tolerance ε.

Fitting Pre-Change Distribution: To define ϕ_1 satisfying (3) we need to know ϕ_0, which is typically unknown. Therefore, we compute its estimate $\widetilde{\phi}_0$ by fitting a GM on the whole dataset of stationary data S. We adopt GM since these are flexible models that can well approximate skewed and multimodal distributions even in high dimensions [13].

The pdf $\widetilde{\phi}_0$ of a GM is a convex combination of k Gaussian functions

$$\widetilde{\phi}_0(\mathbf{x}) = \sum_{i=1}^{k} \frac{\lambda_i}{(2\pi)^{d/2}\det(\Sigma_i)^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu_i)^T \Sigma_i^{-1}(\mathbf{x}-\mu_i)}, \tag{4}$$

where $\sum_{i=1}^{k} \lambda_i = 1$ and $\lambda_i \geq 0$, $i \in 1, \ldots, k$ are the weights of each Gaussian having mean μ_1 and covariance Σ_i. We estimate the weights λ_i and the parameters of the Gaussians using an Expectation-Maximization (EM) algorithm [15], and select the best value of k via cross validation.

Parametrization: To ease calculations, we express Q with respect to its angles of rotation. We stack $m := \lfloor d/2 \rfloor$ angles $\theta_1, \ldots, \theta_m$ in a vector $\boldsymbol{\theta}$, and define the matrix $S(\boldsymbol{\theta}) \in \mathbb{R}^{d\times d}$ as

$$S(\boldsymbol{\theta}) = \begin{bmatrix} R(\theta_1) & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & R(\theta_m) & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix}, \qquad R(\theta_i) = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix}. \tag{5}$$

where $R(\theta_i) \in \mathbb{R}^{2\times 2}$ defines a counterclockwise rotation of angle θ_i around the origin in $\mathbb{R}^2$. When d is even, the last column and the last row of $S(\boldsymbol{\theta})$ are dropped. The matrix $S(\boldsymbol{\theta})$ defines a rotation in $\mathbb{R}^d$ whose planes of rotation are

```
1. Input: $\boldsymbol{\theta}^{(0)}$, $P$ ,$\rho^{(0)}$, $\mathbf{u}$ from Algorithm 1, $\widetilde{\phi}_0$, $\kappa$ and tolerance $\varepsilon$
2. Output: $Q$ and $\mathbf{v}$ defining the roto-translation yielding $\text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_0(Q\cdot+\mathbf{v})) \approx \kappa$
3. Set the lower bounds parameters $\boldsymbol{\theta}_l^{(1)} = 0$, $\rho_l^{(1)} = 0$.
4. Set the upper bounds parameters $\boldsymbol{\theta}_u^{(1)} = \boldsymbol{\theta}^{(0)}$, $\rho_u^{(1)} = \rho^{(0)}$.
5. Set $j = 1$.
6. repeat
7.     Compute $\boldsymbol{\theta}^{(j)} = (\boldsymbol{\theta}_l^{(j)} + \boldsymbol{\theta}_u^{(j)})/2$, and $Q^{(j)}(\boldsymbol{\theta}^{(j)}, P)$ as in (6).
8.     Compute $\rho^{(j)} = (\rho_l^{(j)} + \rho_u^{(j)})/2$, and $\mathbf{v}^{(j)}(\rho^{(j)}, \mathbf{u})$ as in (7).
9.     Compute $s^{(j)} = \text{sKL}(\widetilde{\phi}_0, \phi_1^{(j)})$, where $\phi_1^{(j)}(\cdot) = \widetilde{\phi}_0(Q^{(j)}\cdot+\mathbf{v}^{(j)})$.
10.    if $s^{(j)} < \kappa$ then
11.        $\boldsymbol{\theta}_l^{(j+1)} = \boldsymbol{\theta}^{(j)}$, $\rho_l^{(j+1)} = \rho^{(j)}$.
12.    else
13.        $\boldsymbol{\theta}_u^{(j+1)} = \boldsymbol{\theta}^{(j)}$, $\rho_u^{(j+1)} = \rho^{(j)}$.
14.    end
15.    $j = j + 1$;
16. until $\left|s^{(j)} - \kappa\right| < \varepsilon$;
17. Set $Q = Q^{(j)}$, $\mathbf{v} = \mathbf{v}^{(j)}$.
```

Algorithm 2. Computation of the roto-translation yielding sKL $\approx \kappa$.

generated by the coordinate axes: rotation matrices Q referring to different coordinate axes can be obtained by multiplying $S(\boldsymbol{\theta})$ against an orthogonal matrix $P \in \mathbb{R}^{d\times d}$. Thus, the rotation matrix is parametrized as

$$Q(\boldsymbol{\theta}, P) = PS(\boldsymbol{\theta})P'. \tag{6}$$

The translation vector $\mathbf{v}$ is parameterized by its norm and direction:

$$\mathbf{v}(\rho, \mathbf{u}) = \rho\mathbf{u}, \qquad \text{where} \quad \|\mathbf{u}\|_2 = 1,\ \rho = \|\mathbf{v}\|_2\,. \tag{7}$$

Initialization: Algorithm 1 is meant to define $Q^{(0)}$ and $\mathbf{v}^{(0)}$ satisfying $\text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_0(Q^{(0)}\cdot+\mathbf{v}^{(0)})) > \kappa$. We initially set $\rho^{(0)} = 1$ ([LINE 3]) and randomly generate a vector $\mathbf{u}$ having unitary norm ([LINE 5]), as in [1], Algorithm 11. Moreover, we randomly define the angles of rotation $\boldsymbol{\theta}^{(0)}$ in $[-\pi/2, \pi/2]^m$ ([LINE 5]) and randomly select an orthogonal matrix P ([LINES 5-7]). We then compute the rotation matrix $Q^{(0)}(\boldsymbol{\theta}^{(0)}, P)$ and translation vector $\mathbf{v}^{(0)}(\rho^{(0)}, \mathbf{u})$ as in (6) and (7), respectively ([LINE 8]). If the corresponding $\text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_0(Q^{(0)}\cdot+\mathbf{v}^{(0)}))$ is larger than κ, the algorithm terminates and we use $Q^{(0)}$ and $\mathbf{v}^{(0)}$ to initialize Algorithm 2. Otherwise, we double $\rho^{(0)}$ and repeat this procedure until the condition is satisfied. Convergence of Algorithm 1 is proved in Sect. 4.

Iterations: Algorithm 2 implements a bisection method to compute $\boldsymbol{\theta}$ and ρ yielding the desired sKL value. The algorithm is initialized from $\boldsymbol{\theta}^{(0)}$, P, $\rho^{(0)}$, and $\mathbf{u}$ provided by Algorithm 1, and iteratively adjusts $\boldsymbol{\theta}$ and ρ, while P and $\mathbf{u}$ are fixed. We use the subscripts l and u to denote the parameters yielding $\text{sKL}(\widetilde{\phi}_0, \phi_1)$ values that are smaller and larger than κ, respectively.

Initially, $\boldsymbol{\theta}_l$ and ρ_l are set to 0 ([LINE 3]), while $\boldsymbol{\theta}_u = \boldsymbol{\theta}^{(0)}$ and $\rho_u = \rho^{(0)}$, ([LINE 4]). As in any bisection method, we set $\boldsymbol{\theta}^{(j)}$ to the average of $\boldsymbol{\theta}_l^{(j)}$ and $\boldsymbol{\theta}_u^{(j)}$ ([LINE 7]), and $\rho^{(j)}$ to the average of $\rho_l^{(j)}$ and $\rho_u^{(j)}$ ([LINE 8]). Then, if the corresponding value of sKL, namely $s^{(j)}$ ([LINE 10]), is smaller than the target value κ, we update $\boldsymbol{\theta}_l$ and ρ_l ([LINE 11]), otherwise we update $\boldsymbol{\theta}_u$ and ρ_u ([LINE 13]).

4 Proofs of Convergence

To proof the convergence of the proposed method we demonstrate first that Algorithm 1 terminates after a finite number of iterations (Theorem 1), and then that Algorithm 2 actually converges (Theorem 2).

Theorem 1. *Let $\widetilde{\phi}_0$ be a Gaussian mixture. Then, for any $\kappa > 0$, Algorithm 1 converges in a finite number of iterations.*

Proof. It is enough to show that $\text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_0(Q^{(0)} \cdot + \mathbf{v})) \to \infty$ as $\|\mathbf{v}\|_2 \to \infty$, or that it admits a lower bound that diverges as $\|\mathbf{v}\|_2 \to \infty$. The lower bound follows from the definition of sKL and the Jensen's inequality:

$$\begin{aligned} \text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_0(Q^{(0)} \cdot + \mathbf{v})) &\geq \int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{x}) \log\left(\frac{\widetilde{\phi}_0(\mathbf{x})}{\widetilde{\phi}_0(Q^{(0)}\mathbf{x} + \mathbf{v})}\right) d\mathbf{x} \\ &\geq \int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{x}) \log(\widetilde{\phi}_0(\mathbf{x})) dx - \log\left(\int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{x}) \widetilde{\phi}_0(Q^{(0)}\mathbf{x} + \mathbf{v}) d\mathbf{x}\right). \end{aligned} \tag{8}$$

We now define $f(\mathbf{v}) := \int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{x})\widetilde{\phi}_0(Q^{(0)}\mathbf{x} + \mathbf{v})d\mathbf{x}$, which we observe is a continuous function (see Lemma 16.1 of [6]). We prove the theorem demonstrating that $f(\mathbf{v}) \in L^1(\mathbb{R}^d)$, as this implies that $f(\mathbf{v}) \to 0$ when $\|\mathbf{v}\|_2 \to \infty$, thus that the lower bound (8) diverges. Thus, it follows:

$$\begin{aligned} \int_{\mathbb{R}^d} |f(\mathbf{v})| \, d\mathbf{v} = \int_{\mathbb{R}^d} f(\mathbf{v}) d\mathbf{v} &= \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{x}) \widetilde{\phi}_0(Q^{(0)}\mathbf{x} + \mathbf{v}) d\mathbf{x} d\mathbf{v} \\ &= \int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{x}) \left[\int_{\mathbb{R}^d} \widetilde{\phi}_0(Q^{(0)}\mathbf{x} + \mathbf{v}) d\mathbf{v}\right] d\mathbf{x} = 1. \end{aligned} \tag{9}$$

□

Theorem 2. *Let $\widetilde{\phi}_0$ be a Gaussian mixture. Then, for any $\kappa > 0$, Algorithm 2 converges in a finite number of iterations.*

Proof. The thesis follows from the Intermediate Value Theorem [18] (Theorem 4.23) applied to the function used in the bisection procedure, i.e.

$$sKL(a) := \text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_0(Q(\boldsymbol{\theta}(a), P) \cdot + \mathbf{v}(a))), \tag{10}$$

where $\boldsymbol{\theta}(a) = (1-a)\boldsymbol{\theta}_l^{(1)} + a\boldsymbol{\theta}_u^{(1)}$ and $\rho(a) = (1-a)\rho_l^{(1)} + a\rho_u^{(1)}$ are convex combinations of the initialization parameters (Algorithm 2, lines 2–3). We observe that sKL(0) $< \kappa$ (since $sKL(0) = 0$) and $sKL(1) > \kappa$, thus it is enough to show that $sKL(\cdot)$ is continuous in $[0, 1]$ to prove that there exist a value of $\bar{s}$ such that $sKL(\bar{s}) = \kappa$, thus that $\boldsymbol{\theta}(\bar{s})$ and $\rho(\bar{s})$ yield sKL equal to κ, thus, that bisection method converges (see [8], Theorem 2.1).

To show that $sKL(\cdot)$ in (10) is continuous, we show that $KL(\cdot) = \int_{\mathbb{R}^d} g(\cdot, \mathbf{x})d\mathbf{x}$ is continuous, where

$$g(a, \mathbf{x}) := \widetilde{\phi}_0(\mathbf{x}) \log \left(\frac{\widetilde{\phi}_0(\mathbf{x})}{\widetilde{\phi}_0(Q(\boldsymbol{\theta}(a), P)\mathbf{x} + \mathbf{v}(\rho(a), \mathbf{u}))} \right), \tag{11}$$

It is clear that g is continuous in $[0, 1] \times \mathbb{R}^d$, thus $g(\cdot, \mathbf{x})$ is continuous in $[0, 1]$: to show that $KL(\cdot)$ is also continuous, we leverage Lemma 2.1 in [6] and prove that $|g(a, \mathbf{x})|$ admits a dominant integrable function that does not depend on a. To this purpose, we exploit the following upperbound[2]

$$\left| \log(\widetilde{\phi}_0(\mathbf{x})) \right| \leq c_1 + c_2 \left\| \mathbf{x} \right\|_2^2, \qquad \forall \mathbf{x} \text{ s.t. } \left\| \mathbf{x} \right\|_2 > r. \tag{12}$$

that holds for some constants $c_1, c_2, r > 0$. Then, for a sufficiently large r

$$\begin{aligned} |g(a, \mathbf{x})| &= \widetilde{\phi}_0(\mathbf{x}) \left(\log(\widetilde{\phi}_0(\mathbf{x})) - \log(\widetilde{\phi}_0(Q(\boldsymbol{\theta}(a), P)\mathbf{x} + \mathbf{v}(\rho(a), \mathbf{u}))) \right) \\ &\leq \widetilde{\phi}_0(\mathbf{x})(c_1 + c_2 \left\| \mathbf{x} \right\|_2^2)(c_1 + c_2 \left\| Q(\boldsymbol{\theta}(a), P)\mathbf{x} + \mathbf{v}(\rho(a), \mathbf{u}) \right\|_2^2). \end{aligned} \tag{13}$$

The exponential decay of GM implies that

$$\sqrt{\widetilde{\phi}_0(\mathbf{x})}(c_1 + c_2 \left\| \mathbf{x} \right\|_2^2)(c_1 + c_2 \left\| Q(a\boldsymbol{\theta}^{(0)}, P)\mathbf{x} + \mathbf{v}(a\rho^{(0)}, \mathbf{u}) \right\|_2^2) < c$$

for some $c > 0$. Thus, the function that dominates $|g(\cdot, \mathbf{x})|$ for $\left\| \mathbf{x} \right\|_2 > r$ is $c\sqrt{\widetilde{\phi}_0(\mathbf{x})}$ which obviously belongs[3] to $L^1(\mathbb{R}^d)$. □

5 The Framework

We implement the proposed method in a MATLAB framework which is publicly available for download at http://home.deib.polimi.it/carrerad. The main function of the framework is `generateDatastreams`, which manipulates a dataset S containing i.i.d. data and generates multiple datastreams affected by a change as in (1). At first, this function performs a random shuffling of the dataset to obtain different datastreams everytime the function is invoked. Then it estimates the pdf $\widetilde{\phi}_0$ over the whole S and generates a change having magnitude κ at time τ by performing a roto-translation of the data implementing Algorithms 1 and 2.

The pdf $\widetilde{\phi}_0$ is implemented as the object `gmDistr`, which includes `fit` and `symmetricKullbackLeibler` among its methods. The method `fit` estimates

[2] This bound is trivial for Gaussian pdfs and follows from basic algebra in case of GM.
[3] There is no need to find a dominant function for $\left\| \mathbf{x} \right\|_2 \leq r$ since there $g(\cdot, \mathbf{x})$ is bounded.

$\widetilde{\phi}_0$ over the dataset S and is based on the EM algorithm [15] in MATLAB. The method `symmetricKullbackLeibler` takes as input the ϕ_1 and estimates sKL$(\widetilde{\phi}_0, \phi_1)$ via a Monte Carlo simulation which involves the computation of the log-likelihood with respect to $\widetilde{\phi}_0$ and ϕ_1 on synthetically generated samples. To prevent severe numerical issues raising when computing the log-likelihood of a GM via (4) (in particular when $d \gg 1$), we adopt the following upper-bound that approximates $\widetilde{\phi}_0(\mathbf{x})$ as in [2]

$$\widetilde{\phi}_0(\mathbf{x}) \leq -\frac{k\lambda_{i^*}}{2}\Big(\log\left((2\pi)^d \det(\Sigma_{i^*})\right) + (\mathbf{x}-\mu_{i^*})'\Sigma_{i^*}^{-1}(\mathbf{x}-\mu_{i^*})\Big) \tag{14}$$

where $i^* = \underset{i=1,\ldots,k}{\operatorname{argmax}}\left(\frac{\lambda_i}{(2\pi)^{d/2}\det(\Sigma_i)^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu_i)'\Sigma_i^{-1}(\mathbf{x}-\mu_i)}\right)$. The matrix Q and the vector $\mathbf{v}$ are computed by `computeRotoTranslation`, which implements both Algorithms 1 and 2, as in Sect. 3.

6 Experiments

Experiments are meant to demonstrate the effectiveness of CCM showing the limitations of methods typically used in the literature which do not control the change magnitude. In our experiments we use the *MiniBooNE Particle Dataset* from the UCI repository [14], which contains measurements from a physical experiment designed to distinguish electron neutrinos from muon neutrinos. We consider only the data from the muon class, obtaining a dataset of dimension 50 with 93108 samples. We fit a mixture $\widetilde{\phi}_0$ of $k = 2$ Gaussians, where k is selected by 5-fold cross-validation. To avoid numerical issues, we preprocess the dataset by standardizing each component (i.e. subtracting the sample mean and dividing by the sample standard deviation). We generate 1000 datasets for each value of $d = 1, \ldots, 50$, by randomly choosing d components out of the 50 available.

We consider three methods to synthetically introduce changes:

CCM: it was configured to yield constant sKL$(\widetilde{\phi}_0, \phi_1) = 1\, \forall d$.
Offset: an offset $\nu = 1$ is added to each component of the data (after preprocessing ν equals the standard deviation of each component). With respect to (1) this change model has $Q = I_d$ and $\mathbf{v} = \nu \mathbf{1}_d$.
Swap: two components, randomly chosen, are swapped. This change can be described as in (1), where Q is the corresponding permutation matrix and $\mathbf{v} = 0$.

We measure the magnitude of the introduced changes by computing sKL$(\widetilde{\phi}_0, \phi_1)$, where $\widetilde{\phi}_0$ ($k = 2$) is fitted on each d-dimensional dataset and ϕ_1 is set as in (1) using the parameters defined by the change model. Figure 1(a) shows the distribution of sKL$(\widetilde{\phi}_0, \phi_1)$ for several d. While CCM clearly preserves sKL $= 1$, the Offset and Swap methods do not, and sKL increases with d. Moreover, the dispersion of sKL also increases, indicating that arbitrarily introducing changes using Offset and Swap methods can lead to considerably different magnitudes when d increases.

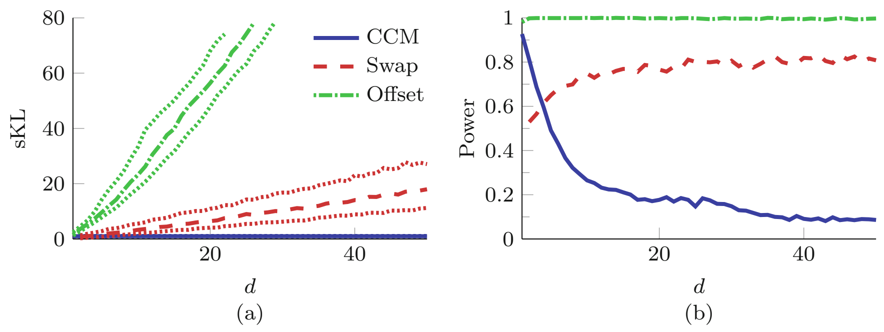


Fig. 1. (a) Values of sKL($\widetilde{\phi}_0, \phi_1$) obtained using CCM, Swap and Offset methods to introduce changes when $d \in \{1, \ldots, 50\}$. Solid, dashed and dash-dotted lines represent the median values, while the dotted lines represent the first and the third quartiles. (b) The estimated powers of t-tests on the log-likelihood values.

Controlling the change-magnitude is important as this determines to a large extent the change-detection performance. As an example, we perform t-test on the mean value of the log-likelihood computed over two windows W_P and W_R containing 200 pre- and post-change samples, respectively. The log-likelihood is computed with respect to a GM estimated on a training set of pre-change data having $200d$ samples. Figure 1(b) reports the estimated power of the t-test when changes are introduced by the different methods and for different values of d. While changes introduced by CCM report a decreasing power, coherently with the study in [2] that demonstrates the detectability-loss problem. The power in case of changes introduced by Swap and Offset increases with d, and this fact indeed prevents to correctly study how d influences the change-detection performance.

7 Conclusions

We have presented CCM, a rigorous method to control the change magnitude in multivariate datasets. The algorithms at the core of CCM are sound and a proof of their convergence is given. Our experiments remark the importance of generating datasets by using CCM to control the change magnitude, rather than more heuristic experimental practices that are commonly adopted. Most importantly, CCM enables to fairly assess the detection performance and investigate the change-detection challenges raising in high dimensional data, as it typically happens in big data scenarios. Future works concern extending CCM to include also change models affecting the data dispersion.

References

1. Alippi, C.: Intelligence for Embedded Systems, A Methodological Approach. Springer, Switzerland (2014)

2. Alippi, C., Boracchi, G., Carrera, D., Roveri, M.: Change detection in multivariate datastreams: Likelihood and detectability loss. In: Proceedings of IJCAI (2016)
3. Alippi, C., Boracchi, G., Roveri, M.: A Just-In-Time adaptive classification system based on the Intersection of Confidence Intervals rule. Neural Netw. **24**(8), 791–800 (2011)
4. Alippi, C., Boracchi, G., Roveri, M.: Just-in-time classifiers for recurrent concepts. IEEE Trans. Neural Netw. Learn. Syst. **24**(4) (2013)
5. Alippi, C., Boracchi, G., Roveri, M.: Hierarchical change-detection tests. IEEE Trans. Neural Netw. Learn. Syst. **PP**(99), 1–13 (2016)
6. Bauer, H.: Measure and Integration Theory. Walter de Gruyter, Berlin (2001)
7. Boracchi, G., Roveri, M.: Exploiting self-similarity for change detection. In: Proceedings of IEEE International Joint Conference on Neural Networks (IJCNN) (2014)
8. Burden, R.L., Faires, J.D.: Numerical Analysis. Brooks/Cole, USA (2001)
9. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3), 15:1–15:58 (2009)
10. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with drift detection. In: Proceedings of Brazilian Symposium on Artificial Intelligence (SBIA) (2004)
11. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM Comput. Surv. (CSUR) **46**(4) (2014)
12. Harel, M., Mannor, S., El-yaniv, R., Crammer, K.: Concept drift detection through resampling. In: Proceedings of ICML, pp. 1009–1017 (2014)
13. Kuncheva, L.I.: Change detection in streaming multivariate data using likelihood detectors. IEEE Trans. Knowl. Data Eng. **25**(5) (2013)
14. Lichman, M.: UCI machine learning repository. http://archive.ics.uci.edu/ml
15. McLachlan, G., Peel, D.: Finite Mixture Models. Wiley, New York (2004)
16. Pimentel, M.A., Clifton, D.A., Clifton, L., Tarassenko, L.: A review of novelty detection. Sig. Process. **99**, 215–249 (2014)
17. Ross, G.J., Tasoulis, D.K., Adams, N.M.: Nonparametric monitoring of data streams for changes in location and scale. Technometrics **53**(4) (2011)
18. Rudin, W.: Principles of Mathematical Analysis. McGraw-Hill, New York (1964)
19. Street, W.N., Kim, Y.: A streaming ensemble algorithm (sea) for large-scale classification. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2001)

Spark Parameter Tuning via Trial-and-Error

Panagiotis Petridis[1], Anastasios Gounaris[1(✉)], and Jordi Torres[2]

[1] Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece
{ppetridis,gounaria}@csd.auth.gr

[2] Computer Architecture Department, Technical University of Catalonia, Barcelona, Spain
torres@ac.upc.edu

Abstract. Spark has been established as an attractive platform for big data analysis, since it manages to hide most of the complexities related to parallelism, fault tolerance and cluster setting from developers. However, this comes at the expense of having over 150 configurable parameters, the impact of which cannot be exhaustively examined due to the exponential amount of their combinations. In this work, we investigate the impact of the most important of the tunable Spark parameters on the application performance and guide developers on how to proceed to changes to the default values. We conduct a series of experiments and we offer a trial-and-error methodology for tuning parameters in arbitrary applications based on evidence from a very small number of experimental runs. We test our methodology in three case studies, where we manage to achieve speedups of more than 10 times.

1 Introduction

Spark [9,10] has emerged as one of the most widely used frameworks for massively parallel data analytics. In summary, it improves upon Hadoop MapReduce in terms of flexibility in the programming model and performance [6], especially for iterative applications. It can accommodate both batch and streaming applications, while providing interfaces to other established big data technologies, especially regarding storage, such as HDFS and NoSQL databases. Its key feature is that it manages to hide the complexities related to parallelism, fault-tolerance and cluster setting from end users and application developers. To support all these, Spark execution engine has been evolved to an efficient albeit complex system with more than 150 configurable parameters. The default values are usually sufficient for a Spark program to run, e.g., not to run out of memory without having the option to spill data on the disk and thus crash. But this gives rise to the following research question: *"Can the default configuration be improved?"*

The aim of this work is to answer the above question in an efficient manner. Clearly, it is practically impossible to check all the different combinations of parameter values for all tunable parameters. Therefore, tuning arbitrary Spark applications by inexpensively navigating through the vast search space of all

© Springer International Publishing AG 2017
P. Angelov et al. (eds.), *Advances in Big Data*, Advances in Intelligent Systems and Computing 529, DOI 10.1007/978-3-319-47898-2_24

possible configurations in a principled manner is a challenging task. Very few research endeavors focus on issues related to understanding the performance of Spark applications and the role of tunable parameters [1,4,7]. For the latter, Spark's official configuration guides and tutorial book [3] provide a valuable asset in understanding the role of every single parameter.

Understanding the role of a parameter does not necessarily mean that the impact of each parameter on the performance of arbitrary applications is understood as well. Moreover, such an understanding does not imply that tuning is straightforward. An added complexity stems from the fact that most parameters are correlated and the impact of parameters may vary from application to application and it will also vary from cluster to cluster. In this work, we experiment with the MareNostrum petascale supercomputer at the Barcelona Supercomputing Center. After configuring the cluster in an application-independent way according to the results in [7], we examine the impact of configurable parameters on a range of applications and derive a simple trial-and-error tuning methodology that can be applied to each application separately. We test our methodology using three case studies with particularly encouraging results.

The summary of our contributions is as follows. (i) We provide an overview of the known results to date on configuring Spark applications. (ii) We identify the most important parameters in terms of their potential impact on performance and we test them on the Marenostrum petascale supercomputer. The number of these parameters is 12. (iii) Based on our results, we propose a novel tuning methodology to be applied on an individual application basis (summarized in Fig. 1). The methodology treats applications as black boxes, follows an efficient trial-and-error approach that involves a low number of experimental runs for just 10 different configurations at most, and takes into account the correlation between different parameters. (iv) We evaluate our methodology in practice using three case studies and we show that we can achieve significant speedups (up to more than 10 times). A more extended version is in [5].

2 Overview of Existing Results for Spark Configuration

In this work, we target parameters belonging to the *Shuffle Behavior, Compression and Serialization*, and *Memory Management* categories. Note that there are several other parameters belonging to categories such as *Application Properties, Execution Behavior* and *Networking* that may affect the performance, but these parameters are typically set at the cluster level, i.e., they are common to all applications running on the same cluster of machines, e.g., as shown in [7]. Next, we summarize the known results to date with regards to Spark configuration.

Optimization of Spark on MareNostrum. The work in [7] sheds lights onto the impact of configurations related to parallelism. The main results, which are reused in our work, are summarized as follows. First, the number of cores allocated to each Spark executor has a big impact on performance and should be configured in an application-independent manner. Second, the level of parallelism, i.e., the number of partitions per participating core, plays a significant role. A range of additional aspects, e.g., using Ethernet instead of Infiniband, have been investigated, but their significance was shown to be small. Finally, the type of the underlying file system affects the performance, however, this is a property of the infrastructure rather than a tunable parameter.

Guides from Spark documentation. Spark official documentation presents a summary of tuning guidelines that can be summarized as follows. (i) The type of the serializer is an important configuration parameter. The default option uses Java's framework, but if `Kryo` library is applicable, it may reduce running times significantly. (ii) The memory allocated to main computations and shuffling and the memory allocated to caching is important. It is stated that *"although there are two relevant configurations, the typical user should not need to adjust them as the default values are applicable to most workloads"*. Memory-related configurations are also related to the performance overhead of garbage collection (GC). (iii) The level of parallelism, i.e., the number of tasks in which each RDD is split needs to be set in a way that the the cluster resources are fully utilized. (iv) Data locality, i.e., enforcing the processing to take place where data resides, is important for distributed applications. Apart from the tuning guidelines above, certain other tips are provided by the Spark documentation, such as preferring arrays to hashmaps. Also, similar guidelines are discussed more briefly in [3]. In our work, we explicitly consider the impact of serialization and memory allocation. Tuning the parallelism degree is out of our scope, but we follow the guidelines in [7]. Also, we do not deal with GC and locality-related thresholds, because they have not seemed to play a big role in our experiments, where memory was not a scarce resource. Based on these guidelines, Alpine Data has published online a so-called *cheat-sheet*, which is a tuning guide for system administrators (available from http://techsuppdiva.github.io/spark1.6.html). The guide is tailored to the YARN cluster manager. Compared to our tuning model, it is more complex, and contains checks from logs and tips to modify the application code and setting of configuration parameters per application. By contrast, we focus only on the latter aspect considering each application as a black box requiring significantly fewer comparisons and considering more tuning parameters.

Other sources. In [1], the impact of the input data volume on Spark's applications is investigated. The key parameters identified were related to memory and compression, although their exact impact is not analyzed. By contrast, we examine a superset of these parameters. The work in [8] is similar to [7] in that it discusses the deployment of Spark on a high-performance computing (HPC) system. In the described its evaluation, it identifies also four key Spark parameters along with the application-dependent level of parallelism. All these parameters are included in our discussion.

Profiling Spark applications and other related work. A thorough investigation of bottlenecks in Spark is presented in [4]. The findings are also very interesting, since it is claimed that many applications are CPU-bound and memory plays a key role. In our experiments, we perform memory fine-tuning, since we also identify memory management as one of the most important parameters.

The work in [2] deals with the issue of parameter optimization in workflows, but may involve far too many experimental runs, whereas we advocate a limited number of configuration runs independently from the application size.

3 Parameters of Interest

Navigating through the vast search space is one the biggest challenges in parameter testing and tuning, due to the exponential increase of different configurations in the number of properties and their valid values. Based on evidence from (i) the documentation and the earlier works presented in Sect. 2 and (ii) our own runs (for which we do not present results due to space constraints), we narrow our focus on 12 parameters, the configuration of which needs to be investigated according to each application instance separately. As already explained, the application-independent parameters that need to be specific for a specific data-center and those related to data parallelism are out of our scope. Finally, we do not consider parameters related to YARN or MESOS.

1. `spark.reducer.maxSizeInFlight:` If this value is increased, reducers would request bigger output chunks. This would increase the overall performance but may aggravate the memory requirements. So, in clusters that there is adequate memory and where the application is not very memory-demanding, increasing this parameter could yield better results. On the other hand, if a cluster does not have adequate memory available, reducing the parameter should yield performance improvements.
2. `spark.shuffle.compress:` In general, compressing data before they are transferred over the network is a good idea, provided that the time it takes to compress the data and transfer them is less than the time it takes to transfer them uncompressed. But if transfer times are faster than the CPU processing times, the main bottleneck of the application is shifted to the CPU and the process is not stalled by the amount of data that are transferred over the network but from the time it takes for the system to compress the data. Clearly, the amount of data transmitted during shuffling is application-dependent, and thus this parameter must not be configured to a single value for all applications.
3. `spark.shuffle.file.buffer:` The role of this parameter bears similarities to the `spark.shuffle.maxSizeInFlight` parameter, i.e., if a cluster has adequate memory, then this value could be increased in order to get higher performance. If not, there might be performance degradation, since too much memory would allocated to buffers.
4. `spark.shuffle.manager:` The available implementations are three: *sort, hash*, and *tungsten-sort. Hash* creates too many open files for certain inputs and aggravates memory limitations. However, combining the enabling of the `spark.shuffle.consolidateFiles` parameter with the *Hash* manager, may mitigate this problem. *Tungsten-sort* is reported to yield the highest performance in general provided that certain requirements are met. Overall, there is no clear winner among the shuffle manager options.

5. `spark.io.compression.codec`: Three options are available, namely *snappy, lz4,* and *lzf*. Although there are many tests conducted by various authors for the generic case, the best performing codec is application-dependent.
6. `spark.shuffle.io.preferDirectBufs`: In environments where off-heap memory is not tightly limited, this parameter may play a role in performance.
7. `spark.rdd.compress`: The trade-offs with regards to this parameter are similar to those for shuffle compress. However, in this case, the trade-off lies between CPU time and memory.
8. `spark.serializer`: In Spark's documentation, it is stated that *KryoSerializer* is thought to perform much better than the default Java Serializer when speed is the main goal. However, the Java serializer is still the default choice, so this parameter needs to be considered.
9. `spark.shuffle.memoryFraction`: If, during shuffling, spills are often, then this value should be increased from its default. Since this parameter is directly linked to the amount of memory that is going to be utilized, it may have a high performance impact. However, any increase is at the expense of the next parameter.
10. `spark.storage.memoryFraction`: Since this parameter is directly linked to the amount of memory that is going to be utilized, it affects the performance.
11. `spark.shuffle.consolidateFiles`: This parameter provides the option of consolidating intermediate files created during a shuffle, so that fewer files are created and performance is increased. It is stated however that, depending on the filesystem, it may cause performance degradation.
12. `spark.shuffle.spill.compress`: As for the previous compression options, a trade-off is involved. If transferring the uncompressed data in an I/O operation is faster than compressing and transferring compressed data, then this option should be set to false. Provided that there is a high amount of spills, this parameter may have an impact on performance.

4 Sensitivity Analysis

We employ three benchmark applications: (i) sort-by-key; (ii) shuffling and (iii) k-means. *K-means* and *Sort-by-key* are also part of the HiBench benchmark and where selected because they can be considered as representative of a variety of applications. The *shuffling* application generates the data according to the terasort benchmark, but does not perform any sorting; it just shuffles all the data in order to stress the shuffling component of the system, given that shuffling in known to play a big role in the performance of Spark applications. To avoid the interference of the underlying file system, in all applications, the dataset was generated at the beginning of each run on the fly. The MareNostrum hardware specifications are described in [7]. 20 16-core machines are used, and the average allocated memory per core is 1.5 GB. The version of Spark at the time of the experiments was 1.5.2.

For each of the selected parameters, we perform a separate set of tests, and in each test we examine a different value. Then, the performance is compared with the performance of the default configuration after modifying the serializer, as argued below. If there is a big difference between the results, then the parameter can be considered as having an impact on the overall performance.

The parameter values are selected as follows. If the parameter takes a binary value, for instance a parameter that specifies whether to use a feature or not, then the non-default value is tested. For parameters that have a variety of different values that are distinct, for instance the compression codec that will be used (*snappy, lzf, lz4*), all the different values are tested. Finally, for parameters that take numeric values in a wide range, e.g., `spark.io.file.buffer`, the values close to the default are tested. Each experiment was conducted five times (at least) and the median value is reported.

Sort-by-Key experiments. The setup of the *sort-by-key* experiments is as follows. 1 billion key-value pairs are used, and each key and value have a length of 10 and 90 bytes, respectively. Also there are both 1000000 unique values for both keys and values. The degree of partitioned is set to 640, which is the optimal configuration according to the results in [7].

We first assess the impact of `spark.serializer`. *KryoSerializer* performs significantly better than the default *Java Serializer*, yielding approximately 25 % lower times. Since this gap is big, and in order to be able to extract insights regarding the performance of the rest of the parameters, the experiments that follow were conducted with the *KryoSerializer*. The impact of all the remaining parameters is summarized in Table 1 (2nd column). As baseline, we use the *KryoSerializer* performance, which is approximately 150 s.

We see that both the non-default shuffle managers perform better than the default. *Hash* performs at 127 s and *Tungsten-sort* at 131 s, nearly 30 s faster than the default. Regarding the memory fraction parameters, the values for `shuffle.memoryFraction` and `storage.memoryFraction` both set to 0.4 provide a less significant speedup (139 s). One would expect these parameters to have a bigger impact, given that *sort-by-key* is a shuffling-intensive application. Interestingly, the second test for these parameters, with values of 0.1 and 0.7, respectively, led to application crash. `spark.reducer.maxSizeInFlight` does not appear to be have a significant impact on performance. Increasing this parameter's value at 96 mb yields the nearly same performance with the default (167 s) but decreasing it to 24 mb gives a small performance improvement (149 s). A similar observation can be drawn for `shuffle.file.buf-fer`; here, increasing the value yields slightly better results (140 s). The biggest impact on performance however can be seen in the `shuffle.compression` test runs. Disabling the compression degrades the performance by more than 100 %. It is worth noting that this may not always be the case, since the behavior of this parameter heavily relies on network and hardware details. Regarding the compression codecs, there is not any noteworthy impact, since both *lzf* and *lz4* seem to be performing similarly to the default codec, *snappy*. Also, the file consolidation implementation

Table 1. Impact of parameters (in seconds)

Configuration	Sort-by-key	Shuffling	k-means (100 M)	k-means (200 M)
`spark.serializer = KryoSerializer`	160	890	22	36
`spark.shuffle.manager = Hash`	127	994	21	39
`spark.shuffle.manager = Tungsten-sort`	131	737	23	36
`shuffle.memoryFraction = 0.4, storage.memoryFraction = 0.4`	139	912	21	42
`shuffle.memoryFraction = 0.1, storage.memoryFraction = 0.7`	N/A	N/A	21	39
`spark.reducer.maxSizeInFlight = 96 mb`	167	880	20	40
`spark.reducer.maxSizeInFlight = 24 mb`	149	845	21	44
`spark.shuffle.file.buffer = 64 k`	140	870	21	37
`spark.shuffle.file.buffer = 16 k`	160	950	23	42
`spark.shuffle.compress = false`	380	2300	21	36
`spark.io.compress.codec = lzf`	159	882	20	37
`spark.io.compress.codec = lz4`	152	1057	23	39
`ppark.shuffle.consolidateFiles = true`	139	905	21	40
`spark.rdd.compress = false`	167	810	21	38
`spark.shuffle.io.preferDirectBufs = false`	169	734	21	36
`spark.shuffle.spill.compress = true`	154	896	22	38

does not appear to provide any significant improvement either. This could be attributed to a variety of reasons, one being the fact that the *sort-by-key* application does not generate a very high number of files during shuffling. The last three parameters, `shuffle.spill.compress`, `shuffle.io.preferDirectBufs` and `rdd.compress` do not seem to significantly affect the performance too. For the former, this can be attributed to the fact that the spills conducted are few. For `Rdd.compress`, there is a small performance degradation as expected, since the RDD can fit into the main memory and CPU time is unnecessarily spent for the compression.

Shuffling Experiments. In this set of experiments, the cluster setting was kept as in the previous one. The raw dataset is 400 GB. Since the RDDs occupy more space, and the overall available memory in the executors is approximately 400 GB for all RDDs and shuffling, accesses to the local storage take place.

Again, we first test the impact of `spark.serializer` and compare it with the default performance. *KryoSerializer* performs approx. 10 % better than the default Java serializer. In the experiments that follow, we use the performance of the *KryoSerializer* as the baseline performance, which is 815 s. In Table 1 (3rd column), the results for the performance impact of the rest of the parameters are presented. Contrary to the previous experiments, the *Hash* shuffle manager performs worse resulting in performance degradation of 200 s. This could probably be attributed to the fact that because the input much larger than the available memory for shuffling. On the other hand, the *Tungsten-sort* manager yields a speedup of 90 s. Similarly to *sort-by-key*, increasing the memory available for shuffling at the expense of the `storage.memoryFraction` does not benefit the performance. Also, doing the opposite does not leave enough memory for shuffling and the application crashes. Regarding the `reducer.maxSizeInFlight` parameter, there seems to be no significant impact on the application. Changing this parameter seems to affect systems that have very small amounts of memory available. The same applies to `shuffle.file.buffer` parameter. The only difference however is that, when reducing the buffer size from 32 KB to 15 KB, the performance degrades by about 135 s, which is more than 10 % from the initial execution. This might happen for the following reason. The buffers reduce the number of disk seeks and system calls made during the creation of intermediate shuffle files. When the input is big enough and the buffers are relatively small, the number of system calls and disk seeks increases thus leading to performance degradation. In addition, disabling the shuffle compression offers no improvement and greatly increases completion time. Regarding the compression codec parameter, the *lzf* codec does not seem to have any impact on the overall performance. However, the *lz4* codec increases the application completion time by about 200 s, i.e., incurring 25 % extra overhead. Finally, the last three parameters configuration produce runtimes near the default value as was the case in the *sort-by-key* experiments. As such, they considered as not having any significant impact on the performance. As stated before though, this may be specific to the infrastructure used, and in other environments, they may behave differently.

K-means experiments. The final series of experiments tests the impact of the parameters on the performance of *k-means*. We experimented with two data inputs with 100 M and 200 M 100-dimensional records, respectively. The number of centers was 10 and the amount of iterations was fixed to 10 to allow for fair time measurements. In these cases, the impact of the *KryoSerializer* (chosen as default) is very small. The impact of all the other parameters is shown in Table 1. An initial observation is that the impact of all parameters is less significant than in the previous two applications. From the fourth column we can see that, in absolute times, the differences are at most 2 s, which is less than 10 %. This also applies, to a slightly less extent, to the fifth column as well, where the differences are up to 3 s. However, although we conclude that the parameters selected do not affect the performance of the *k-means* benchmark significantly, we can note the fact that the parameter `shuffle.compress` that dramatically

Table 2. Average parameter impact

	Sort-by-key	Shuffling	K-means	Average
`spark.serializer`	26.6 %	9.2 %	<5 %	12.6 %
`shuffle/storage.memoryFraction`	13.1 %	11.9 %	8.3 %	11.3 %
`spark.reducer.maxSizeInFlight`	5.5 %	5.7 %	11.5 %	7.5 %
`spark.shuffle.file.buffer`	6.3 %	11.6 %	6.9 %	8.2 %
`spark.shuffle.compress`	137.5 %	182 %	<5 %	107.2 %
`spark.io.compress.codec`	<5 %	18 %	6.1 %	8.9 %
`spark.shuffle.consolidateFiles`	13 %	11 %	7.7 %	10.5 %
`spark.rdd.compress`	<5 %	<5 %	5 %	<5 %
`spark.shuffle.io.preferDirectBufs`	5.6 %	9.9 %	<5 %	5.9 %
`spark.shuffle.spill.compress`	<5 %	6.1 %	<5 %	<5 %

degraded the performance in the previous experiments, has no overall impact on the test *k-means* instances. This is actually expected, since it mostly affects the data shuffling part of the application, which plays a small, non-dominant role in *k-means*. The same reasoning, i.e., data shuffling is not prominent in *k-means*, explains the less impact of the other parameters, too.

Summary Statistics. In Table 2, the average impact of each parameter can be seen for each benchmark. All percentages refer to the mean deviation from the default runtime, regardless of whether the deviation is for the better or worse performance. The default is enabling the *KryoSerializer*, apart from when testing the impact of this serializer itself. The lowest quartile of the parameters in terms of the magnitude of their incurred average impact are disregarded in the following tuning methodology, so that the latter remains as simple as possible. We make an exception for `spark.shuffle.spill.compress`, as explained later.

5 The Proposed Tuning Methodology

Based on (i) the results of the previous section, (ii) the expert knowledge as summarized in Sect. 2 and (iii) our overall experience from running hundreds of experiments (not all are shown in this work), we derive an easily applicable tuning methodology. This methodology is presented in Fig. 1 in the form of a block diagram. In the figure, each node represents a *test run* with one or two different configurations. Test runs that are higher in the figure are expected to have a bigger impact on performance and, as a result, a higher priority. As such, runs start from the top and, if an individual configuration improves the performance, the configuration is kept and passed to its children replacing the default value for all the test runs on the same path branch. If an individual configuration does not improve the performance, then the configuration is not added

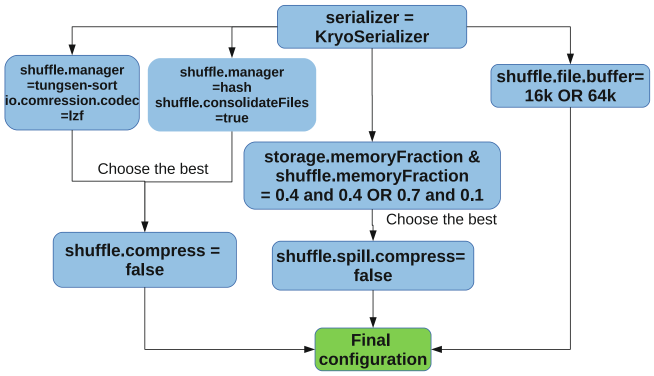

serializer = KryoSerializer
shuffle.manager =tungsten-sort io.comression.codec =lzf
shuffle.manager =hash shuffle.consolidateFiles =true
shuffle.file.buffer= 16k OR 64k
Choose the best
storage.memoryFraction & shuffle.memoryFraction = 0.4 and 0.4 OR 0.7 and 0.1
Choose the best
shuffle.compress = false
shuffle.spill.compress= false
Final configuration

Fig. 1. Our proposed spark parameter tuning methodology

and the default is kept. In other words, each parameter configuration is propagated downstream up to the final configuration as long as it yields performance improvements. Contrary to the previous experiments, the methodology test runs investigate the combined effect of tuning.

Overall, as shown in the figure, at most ten configurations need to be evaluated referring to nine of the parameters in Sect. 3. Note that, even if each parameter took only two values, exhaustively checking all combinations would result in $2^9 = 512$ runs. Finally, the methodology can be employed in a less restrictive manner, where a configuration is chosen not only if it improves the performance, but if the improvement exceeds a threshold, e.g., 5 % or 10 %.

The rationale of the diagram blocks, itemized by the main parameter they target, is as follows:

– `spark.serializer`: This parameter had the highest impact in our series of experiments, and using the *KryoSerializer* was the default baseline for all the other parameters. We keep the same rationale in our methodology, so we perform this test first.
– `spark.shuffle.manager`: As shown in the results of the previous section, the shuffle manager has a high impact on performance, so it should be included in the methodology. Since, based on documentation, *tungsten-sort* works better with the *lzf* compression codec, we combine the test of these two settings. Also, the test run for the other option of this parameter, the *hash* shuffling manager, is conducted in combination with the implementation of consolidating files during a shuffle, to avoid problems from the creation of too many intermediate files.
– `spark.shuffle.compress`: In our parameters, disabling it led to serious performance degradation (by default it is enabled). This means that it has an impact. Interestingly, the best results presented by Spark's developers for the *terasort* benchmark are produced when this is disabled, which further supports our choice to include it in our methodology.

- `storage/shuffle.memoryFraction:` The memory fraction allocated is inherently important in Spark, due to its main memory-oriented execution model. However, this parameter is also tightly connected to the hardware characteristics of the cluster infrastructure.
- `spark.shuffle.spill.compress:` While this parameter appears not to have any significant impact on the performance in the experiments that we conducted, it is closely linked to the shuffling memory fraction. Since the latter is taken into consideration, we also include this one.
- `spark.shuffle.file.buffer:` In our experiments, the impact of this parameters is rather small. However, it is included for completeness. A shorter version of our methodology with two required runs less, would omit it.

We test the effectiveness of our methodology using three case studies. More specifically, the methodology is applied to the *sort-by-key*, *k-means* and *aggregate-by-key* benchmark applications, respectively. For the former, we keep the same input as previously. For *k-means*, we use a different data input, which leads to radically different performance than previously, i.e., this instance of *k-means* cannot be deemed as used for constructing our methodology. *Aggregate-by-key* is a new application.

Sort-by-key. For *Sort-by-key* over 1 billion 100-byte records, the default performance is 218 s. We set a performance improvement threshold at 10 % of this value, i.e., 21.8 s. The final configuration: advocated by our methodology is `spark.serializer=` *KryoSerializer* and `shuffle.manager=` *hash* and `shuffle.consolidateFiles=` *true* and `shuffle/storage.memoryFraction` = 0.4/0.4: 120 s. Overall, the running time in this case study, decreased from 218 s down to 120 s (44 % performance improvement).

K-Means. Next, we apply the methodology on *K-Means* for 10 centers, 10 iterations, 100 million points and 500 columns. The runtime with the default configuration is 654 s. The final configuration does not include the *KryoSerializer* and is as follows: `shuffle/storage.memoryFraction` = 0.1/0.7 and `shuffle.spill.compress`=false. Overall the running time dropped by more than 91 %, from 654 to 54 s. An interesting note is that in the profiling experiments in Sect. 4, *k-means* did not exhibit such high speedups. This is due to the fact that the performance of *k-means* is sensitive to its input dataset.

Aggregate-by-key. The running time for the default configuration is 77.5 s. As input, we use 2 billion key-value pairs with length of 10 and 90, respectively. In this case study, we set the threshold to a lower values, at 5 %. For *aggregate-by-key*, the overall performance improvement is about 21 %, and the configuration is `shuffle.manager=` *hash* and `shuffle.consolidateFiles=` *true* and `shuffle/storage.memoryFraction` = 0.1/0.7.

6 Conclusions

This work deals with configuring Spark applications in an efficient manner. We focus on 12 key application instance-specific configurable parameters and assess their impact using real runs on a petaflop supercomputer. Based on the results and the knowledge about the role of these parameters, we derive a trial-and-error methodology, which requires a very small number of experimental runs. We evaluate the effectiveness of our methodology using three case studies, and the results show that we can achieve up to more than a 10-fold speedup. Although our results are significant, further research is required to investigate additional infrastructures, benchmark applications, parameters and combinations.

References

1. Awan, A.J., Brorsson, M., Vlassov, V., Ayguade, E.: How data volume affects spark based data analytics on a scale-up server (2015). arXiv:1507.08340
2. Holl, S., Zimmermann, O., Palmblad, M., Mohammed, Y., Hofmann-Apitius, M.: A new optimization phase for scientific workflow management systems. Future Gener. Comput. Syst. **36**, 352–362 (2014)
3. Karau, H., Konwinski, A., Wendell, P., Zaharia, M.: Learning Spark: Lightning-Fast Data Analysis. O'Reilly Media, Sebastopol (2015)
4. Ousterhout, K., Rasti, R., Ratnasamy, S., Shenker, S., Chun, B.G.: Making sense of performance in data analytics frameworks. In: 12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2015), pp. 293–307 (2015)
5. Petridis, P., Gounaris, A., Torres, J.: Spark parameter tuning via trial-and-error. arXiv:1607.07348
6. Shi, J., Qiu, Y., Minhas, U.F., Jiao, L., Wang, C., Reinwald, B., Özcan, F.: Clash of the titans: mapreduce vs. spark for large scale data analytics. PVLDB **8**(13), 2110–2121 (2015)
7. Tous, R., Gounaris, A., Tripiana, C., Torres, J., Girona, S., Ayguadé, E., Labarta, J., Becerra, Y., Carrera, D., Valero, M.: Spark deployment and performance evaluation on the marenostrum supercomputer. In: IEEE International Conference on Big Data (Big Data), pp. 299–306 (2015)
8. Wang, Y., Goldstone, R., Yu, W., Wang, T.: Characterization and optimization of memory-resident mapreduce on HPC systems. In: 28th International Parallel and Distributed Processing Symposium, pp. 799–808 (2014)
9. Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauly, M., Franklin, M.J., Shenker, S., Stoica, I.: Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. In: NSDI 2012 (2012)
10. Zaharia, M., Chowdhury, M., Franklin, M.J., Shenker, S., Stoica, I.: Spark: cluster computing with working sets. In: HotCloud 2010 (2010)

Parallel Computing TEDA for High Frequency Streaming Data Clustering

Xiaowei Gu[1], Plamen P. Angelov[1](✉), German Gutierrez[2], Jose Antonio Iglesias[2], and Araceli Sanchis[2]

[1] Data Science Group, School of Computing and Communications, Lancaster University, Lancaster LA1 4WA, UK
{x.gu3,p.angelov}@lancaster.ac.uk

[2] CAOS Research Group, Computer Science Department, Carlos III University of Madrid, 28911 Leganes, Spain
{ggutierr,jiglesia,masm}@inf.uc3m.es

Abstract. In this paper, a novel online clustering approach called *Parallel_TEDA* is introduced for processing high frequency streaming data. This newly proposed approach is developed within the recently introduced TEDA theory and inherits all advantages from it. In the proposed approach, a number of data stream processors are involved, which collaborate with each other efficiently to achieve parallel computation as well as a much higher processing speed. A fusion center is involved to gather the key information from the processors which work on chunks of the whole data stream and generate the overall output. The quality of the generated clusters is being monitored within the data processors all the time and stale clusters are being removed to ensure the correctness and timeliness of the overall clustering results. This, in turn, gives the proposed approach a stronger ability of handling shifts/drifts that may take place in live data streams. The numerical experiments performed with the proposed new approach *Parallel_TEDA* on benchmark datasets present higher performance and faster processing speed when compared with the alternative well-known approaches. The processing speed has been demonstrated to fall exponentially with more data processors involved. This new online clustering approach is very suitable and promising for real-time high frequency streaming processing and data analytics.

Keywords: High frequency streaming data · TEDA · Parallel computation · Clustering · Real time

1 Introduction

The amount and scale of the data streams grow rapidly because of the more mature technologies and lower prices of various electronic devices as well as wider distributed sensor networks. As the world has already entered the Era of Big Data, data-intensive technologies have been extensively used in the developed economies and numerous international organizations and companies are now much more frequently getting involved in Internet-based activities. All of these have largely increased the demand for developing advanced intensive data processing/analysis methodologies and technologies.

Traditional clustering methods more often share several common problems [1–8], for example, they require the number of clusters [2,3] or thresholds [4–6] or radii [7]. Most of the existing methods are offline [9], require iterative calculations. The minority of online, incremental, one pass approaches [10] still require user-defined parameters like radii, thresholds which makes them not fully autonomous and unsupervised. Very recently, completely autonomous approaches were proposed [11,12], however, they do not consider the big data, high frequency, parallel computation aspects. In this paper, we propose *Parallel_TEDA* approach which is addressing precisely these aspects. It is memory- and computation-efficient. Indeed, a single data processor might also have difficulties in handling data samples which arrive in a very high speed.

Some published algorithms [13–16] try to improve computation efficiency by dividing the streaming data chunk by chunk and processing them separately. Then the algorithms fuse the individual clustering results together. However, such algorithms [13–16] lack the ability to detect the evolutions and transitions of the clusters, which sometimes lead to the fusion of non-related clusters [17].

TEDA (Typicality and Eccentricity Data Analysis) was introduced in [18–20] as a fully data-driven and prior-assumptions-free framework for data analysis. Moreover, TEDA supports the recursive form of calculation and is parameter-free. Therefore, TEDA is a suitable algorithm for real-time streaming data processing.

In this paper, we propose a novel online clustering approach for big data streams called *Parallel_TEDA*, which is developed on the basis of TEDA [18–20]. As a result, the proposed approach inherits all its advantages, including no requirement of user input or assumptions, regardless the shape of clusters, recursive computation, self-adaptation to the data pattern. Meanwhile, the idea of parallel processing [13–16] is involved to make it suitable for high frequency data streams processing. Within the *Parallel_TEDA* framework, a number of data streams processors and a fusion center are involved to collaborate with each other to process the high frequency data stream. As a result, the processing speed is largely improved because of its parallel computation structure. Within the proposed approach, the high frequency data stream is split into data chunks, which are passed to different processors. Once the data samples are being processed by the processors, they are discarded to improve the memory-efficiency and only the key information is stored. The fusion center automatically gathers all the key information from the individual processors and fuses together into the final clustering results. Because of the computationally lean recursive expression, the calculations are very fast and no iterations or search is involved which makes possible for real time processing of high frequency data streams. Moreover, compared with the traditional parallel processing techniques [13–16], the proposed *Parallel_TEDA* is able to follow the ever changing data pattern and successfully avoids the problem of meaningless fusion because of advantages of TEDA.

Additionally, we use the time tag, age [21] to monitor the quality of the existing clusters stored in the data stream processors and enhance its ability to self-evolve. Thus, *Parallel_TEDA* is able to follow the changes of the data pattern and avoid the accumulation of error. Because of the time tag, when the *Parallel_TEDA* approach is dealing with multi-data streams, there is no need to wait for processing the data streams sequentially one by one. Once, there is any free processor, the next data stream can be processed seamlessly. In addition, several data streams can be processed together at the same time because of the multi-processors structure.

The remainder of this paper is organized as follows: Sect. 2 introduces the theoretical basis of the TEDA framework. The details of the proposed *Parallel_TEDA* approach are described in Sect. 3. In Sect. 4, the algorithm of the proposed *Parallel_TEDA* approach is presented. Section 5 is for numerical experiments as well as analysis and the conclusions are presented in Sect. 6.

2 Theoretical Basis of the TEDA Framework

Typicality and Eccentricity based Data Analytics (TEDA) was introduced recently [18–20] as a non-parametric, assumption-free methodology for extracting information from data. In its nature, it is close to statistical learning; however, traditional statistical methods assume random variables and try to approximate these *prior* assumptions with the experimental data. TEDA is free from the range of assumptions made in traditional probability theory and statistical learning and is entirely data-driven. No assumption is made for data (in)dependence, number, nature (determinative or stochastic). The main quantities (measures) of mutual (ensemble) properties of the data, namely, typicality can be seen as similar to probability and membership function in fuzzy sets, but it is objective and assumptions and parameters free.

2.1 Cumulative Proximity

Cumulative proximity [18,19,22] is the sum of distances from a particular data sample to all the available data samples. The cumulated proximity of $\boldsymbol{x}_i$ to all other data samples is expressed as:

$$\pi_k(\boldsymbol{x}_i) = \sum_{l=1}^{k} d^2(\boldsymbol{x}_i, \boldsymbol{x}_l); \quad i = 1, 2, \ldots, k; \quad k > 1 \tag{1}$$

where $d(\boldsymbol{x}_i, \boldsymbol{x}_l)$ denotes any distance measure between $\boldsymbol{x}_i$ and $\boldsymbol{x}_l$. For clarity, in this paper, we use the most commonly used Euclidean type of distance, namely $d(\boldsymbol{x}_i, \boldsymbol{x}_l) = \|\boldsymbol{x}_i - \boldsymbol{x}_l\|$.

2.2 Standardized Eccentricity

Eccentricity and its standardised form [18–20] are derived by normalising the cumulative proximity and are very useful for anomaly detection, image processing, fault detection, etc.

The eccentricity of $\boldsymbol{x}_i$ is calculated as [18,19].

$$\xi_k(\boldsymbol{x}_i) = \frac{2\pi_k(\boldsymbol{x}_i)}{\sum_{l=1}^{k} \pi_k^i(\boldsymbol{x}_l)};\ i = 1, 2, \ldots, k;\ k > 1 \tag{2}$$

Standardized eccentricity is derived directly from the eccentricity [18,19].

$$\varepsilon_k(\boldsymbol{x}_i) = k\xi_k^i(\boldsymbol{x}_i) = \frac{2k\pi_k(\boldsymbol{x}_i)}{\sum_{l=1}^{k} \pi_k(\boldsymbol{x}_l)};\ i = 1, 2, \ldots, k;\ k > 1 \tag{3}$$

It is obvious that the sums of the eccentricity and standardized eccentricity are 2 and $2k$, respectively. Compared with eccentricity, standardized eccentricity is a more convenient measure because when k tends to be infinity, its mean value does not drop fast, namely more stable than eccentricity. Using it, the well-known *Chebyshev inequality* [24] obtains a very elegant form.

2.3 Typicality

Typicality is the main measure introduced within TEDA [18–20]. It can be considered as a form of centrality or the normalized inverse cumulative proximity [20]. The typicality of $\boldsymbol{x}_i$ $(j = 1, 2, \ldots, k;\ k > 1)$ is [20]:

$$\tau_k(\boldsymbol{x}_i) = \frac{\frac{1}{\pi_k(\boldsymbol{x}_i)}}{\sum_{l=1}^{k} \frac{1}{\pi_k(\boldsymbol{x}_l)}}) = [\pi_k(\boldsymbol{x}_i)[\sum_{l=1}^{k}(\pi_k(\boldsymbol{x}_l))^{-1}]]^{-1} \tag{4}$$

Typicality represents spatial distribution pattern of the data and resembles the probability distribution function [19].

2.4 Recursive Form of Calculation

It is very important in the context of working with high frequency live data streams to have one pass type of algorithms and recursive calculations. In this case, all quantities considered within the TEDA framework, namely, the cumulative proximity, standardized eccentricity and typicality can all be updated recursively. With Euclidean distance, the cumulated proximity of new coming data sample $\boldsymbol{x}_{k+1}$ at the $(k+1)^{th}$ time instance is recursively calculated as follows [18,22]:

$$\pi_{k+1}(\boldsymbol{x}_{k+1}) = (k+1)(\left\|\boldsymbol{x}_{k+1} - \boldsymbol{\mu}_{k+1}\right\|^2 + X_{k+1} - \left\|\boldsymbol{\mu}_{k+1}\right\|^2)$$

$$\text{where } \boldsymbol{\mu}_{k+1} = \tfrac{k}{k+1}\boldsymbol{\mu}_k + \tfrac{1}{k+1}\boldsymbol{x}_{k+1}; \boldsymbol{\mu}_1 = \boldsymbol{x}_1 \tag{5}$$

$$X_{k+1} = \tfrac{k}{k+1}X_k + \tfrac{1}{k+1}\left\|\boldsymbol{x}_{k+1}\right\|^2;\ X_1 = \left\|\boldsymbol{x}_1\right\|^2$$

The corresponding sum of cumulated proximity can be updated as follows [20]:

$$\sum_{j=1}^{k+1} \pi_{k+1}(\boldsymbol{x}_j) = \sum_{j=1}^{k+1}\sum_{l=1}^{k+1}(\boldsymbol{x}_j - \boldsymbol{x}_l)^T(\boldsymbol{x}_j - \boldsymbol{x}_l) = 2(k+1)^2(X_{k+1} - \left\|\boldsymbol{\mu}_{k+1}\right\|^2) \tag{6}$$

Using Eqs. (5) and (6), the recursive form of the standardized eccentricity can be obtained as:

$$\varepsilon_{k+1}(\boldsymbol{x}_{k+1}) = \frac{\left\|\boldsymbol{x}_{k+1} - \boldsymbol{\mu}_{k+1}\right\| + X_{k+1} - \left\|\boldsymbol{\mu}_{k+1}\right\|^2}{X_{k+1} - \left\|\boldsymbol{\mu}_{k+1}\right\|^2} = 1 + \frac{\left\|\boldsymbol{x}_{k+1} - \boldsymbol{\mu}_{k+1}\right\|}{X_{k+1} - \left\|\boldsymbol{\mu}_{k+1}\right\|^2} \quad (7)$$

The recursive update of the typicality $\tau_k(\boldsymbol{x}_j)$ can obviously be achieved using Eq. (5).

Before and independently of TEDA, the clustering quality and monitoring parameter, *age*, was introduced in 2005 [21].

2.5 Cluster Age

Since the proposed approach is for live streaming data, the data pattern of the stream will potentially change along with time elapse, as a result, the old existing clusters may not be able to represent well the ensemble properties of the data samples following a possible shift or drift [22,23]. Cluster age [22] allows to decide whether a cluster is outdated. Cluster age [21] is an accumulated relative time tag which allows to decide whether a cluster is outdated and is expressed as follows:

$$A_k^c = k - \frac{\sum_{l=1}^{S_k^c} I_l^c}{S_k^c};\ c = 1, 2, ...C;\ S_k^c \geq 1; k > 1 \quad (8)$$

where the sub-label c denotes the c^{th} cluster and C is the number of existing clusters in the clustering result; S_k^c is the support of the c^{th} cluster at the time instance k (number of data samples associated with it).

2.6 Chebyshev Inequality

The well-known *Chebyshev inequality* [24] describes the probability of the distance between a certain data sample $\boldsymbol{x}_i$ and the mean value to be larger than n times the standard deviation, namely $n\sigma$. For Euclidean type of distance it has the following form:

$$P(\|\boldsymbol{x}_i - \boldsymbol{\mu}_k\| \leq n^2\sigma^2) \geq 1 - \frac{1}{n^2} \quad (9)$$

In TEDA, this condition is expressed regarding standardized eccentricity and in a more elegant form [19]:

$$P(\varepsilon_k(\boldsymbol{x}_i) \leq 1 + n^2) \geq 1 - \frac{1}{n^2} \quad (10)$$

That means, we can directly check the value of the standardized eccentricity, ε_k and see if it is less than 10 for the "3σ" case.

3 The Proposed Approach Parallel_TEDA

First of all, let us denote the data stream in the Hilbert space R^d as $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k, \ldots\}$, where the index k indicates the time instance at which the k^{th} data sample arriving. The continuously arriving data samples are divided into chunks and sent to different data stream processors (assuming there are processors and the chunk size is K) according to the time instance of arrival, namely:

$$\mathbf{X} = \begin{bmatrix} \underbrace{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_K}_{chunk1 \rightarrow processor1} \\ \underbrace{\boldsymbol{x}_{K+1}, \boldsymbol{x}_{K+2}, \ldots \boldsymbol{x}_{2K}}_{chunk2 \rightarrow processor2} \\ \vdots \\ \underbrace{\boldsymbol{x}_{(N-1)K+1}, \boldsymbol{x}_{(N-1)K+2}, \ldots \boldsymbol{x}_{NK}}_{chunkN \rightarrow processorN} \\ \underbrace{\boldsymbol{x}_{NK+1}, \boldsymbol{x}_{NK+2}, \ldots \boldsymbol{x}_{(N+1)K}}_{chunk(N+1) \rightarrow processor1} \\ \underbrace{\boldsymbol{x}_{(N+1)K+1}, \boldsymbol{x}_{(N+1)K+2}, \ldots \boldsymbol{x}_{(N+2)K}}_{chunk(N+2) \rightarrow processor2} \\ \vdots \end{bmatrix} \tag{11}$$

For each data chunk, once it is sent to the data stream processor, it will be processed separately and the current processing result will only be influenced by the previous data chunks processed within the same data stream processor and will also influence the future output of the processor. Therefore, for the remainder of this section, all the basic elements of *Parallel_TEDA* approach introduced are considered to be within the same particular data stream processor.

The collection of data samples entered the i^{th} data processor are denoted as $\mathbf{X}^i = \{\boldsymbol{x}_1^i, \boldsymbol{x}_2^i, \ldots, \boldsymbol{x}_K^i\}$ $(i = 1, 2, \ldots, N)$, and the number of samples, which can also be viewed as time instance, is k and will keep growing with time. The number of data samples processed by each processor is considered to be the same.

Parallel_TEDA approach is divided into two stages. The first stage is the clusters and parameters updating within the individual data stream separately. We consider this stage to be a separate processing. The second stage is for merging the separate clustering results of all existing data streams together, called clusters fusion. The architecture of the proposed approach is presented in Fig. 1.

In the remainder of this section, the proposed approach will be described in detail.

3.1 Separate Processing Stage

In this stage, all the data chunks separated from the data stream are processed separately in their corresponding processors. For the i^{th} data stream processor, at each time instance a new data sample arrives, denoted as $\boldsymbol{x}_{k+1}^i$,

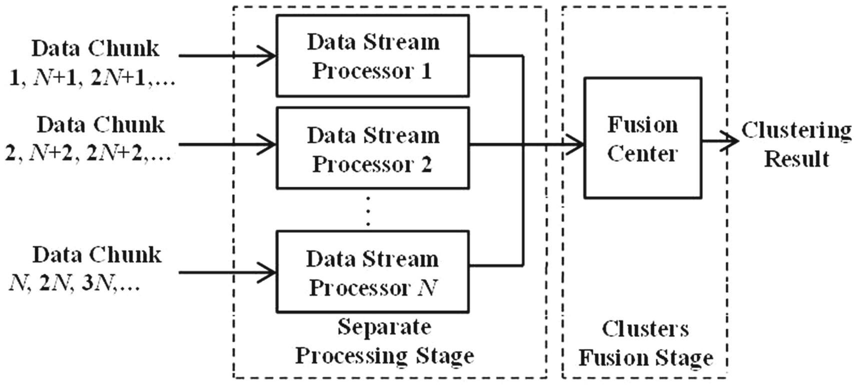


Fig. 1. The architecture of *Parallel_TEDA*

their standardized eccentricity per cluster is calculated using Eq. (7). Then, we monitor $\varepsilon^{i,c}_{k+1}(\boldsymbol{x}^i_{k+1})$; $c = 1, 2, \ldots, C^i$, here C^i denotes the number of existing clusters in the i^{th} processors.

For every cluster, the standardized eccentricities of all its members sum to $2S^{i,c}_k$, and the average standardized eccentricity is $\varepsilon_{average} = 2$. Combining the Chebyshev conditions in terms of the standardised eccentricity (Eq. (10)) [19], we use $n = 2$ to achieve a balance between sensitivity to anomalous data and tolerance to the intra-cluster variance, namely, $\varepsilon_o = 5$. The condition for associating a data sample with a particular cluster can be expressed as follows:

$$\text{IF } (\varepsilon^{i,c}_{k+1}(\boldsymbol{x}^i_{k+1}) \leq \varepsilon_o) \quad \text{THEN } (\boldsymbol{x}^i_{k+1} \in c) \tag{12}$$

For each new data sample $\boldsymbol{x}^i_{k+1}$, if the condition to a particular existing cluster is met, it is associated with this cluster. If χ^i_{k+1} is associated with two or more clusters, it is associated with the cluster that satisfies the following condition:

$$c_{selected} = argmin^{C_i}_{c=1}\varepsilon^{i,c}_{k+1}(\boldsymbol{x}^i_{k+1}) \tag{13}$$

Once, the cluster that $\boldsymbol{x}^i_{k+1}$ should be assigned to is decided, the corresponding mean (center) $\boldsymbol{\mu}^{i,c_{selected}}_{k+1}$, the mean of the scalar product $X^{i,c_{selected}}_{k+1}$, age $A^{i,c_{selected}}_{k+1}$ and support $S^{i,c_{selected}}_{k+1}$ of the cluster need to be updated accordingly:

$$\begin{aligned}
\boldsymbol{\mu}^{i,c_{selected}}_{k+1} &\leftarrow \left(\frac{S^{i,c_{selected}}_k}{S^{i,c_{selected}}_k+1}\boldsymbol{\mu}^{i,c_{selected}}_k + \frac{1}{S^{i,c_{selected}}_k+1}\boldsymbol{x}^i_{k+1}\right) \\
X^{i,c_{selected}}_{k+1} &\leftarrow \left(\frac{S^{i,c_{selected}}_k}{S^{i,c_{selected}}_k+1}X^{i,c_{selected}}_k + \frac{1}{S^{i,c_{selected}}_k+1}\left\|\boldsymbol{x}^i_{k+1}\right\|^2\right) \\
A^{i,c_{selected}}_{k+1} &\leftarrow \left(k+1-\frac{S^{i,c_{selected}}_k(k-A^{i,c_{selected}}_k)+k+1}{S^{i,c_{selected}}_k+1}\right) \\
S^{i,c_{selected}}_{k+1} &\leftarrow (S^{i,c_{selected}}_k+1)
\end{aligned} \tag{14}$$

In contrast, if there is no cluster that meets the condition (Eq. (12)) for a particular data sample, $\boldsymbol{x}_{k+1}^{i}$, the new data sample $\boldsymbol{x}_{k+1}^{i}$ forms a new cluster. The parameters of the new cluster are then initialised as follows:

$$C_i \leftarrow (C_i+1);\quad \boldsymbol{\mu}_{k+1}^{i,C_i} \leftarrow \quad \boldsymbol{x}_{k+1}^{i};\quad X_{k+1}^{i,C_i} \leftarrow \left\|\boldsymbol{x}_{k+1}^{i}\right\|^2;\quad A_{k+1}^{i,C_i} \leftarrow 0;\quad S_{k+1}^{i,C_i} \leftarrow 1 \tag{15}$$

For the existing clusters which do not have new member at the $(k+1)^{th}$ time instance, their age should be updated using Eq. (16) All other parameters do not change.

$$A_{k+1}^{i,C_i} \leftarrow (A_{k}^{i,C_i}+1), c \in Other \tag{16}$$

After the parameters updating, before the processor handles the next input data sample, every existing cluster needs to be checked whether it is already out of date according to its age using the following ageing condition:

$$\text{IF } (A_{k+1}^{i,C_i} > \mu_A^i + \sigma_A^i) \text{ AND } (A_{k+1}^{i,C_i} > K) \text{ THEN (The } c^{th} \text{ cluster is out of date)} \tag{17}$$

where μ_A^i is the average value of the ages of all clusters within the processor, σ_A^i is standard deviation of all clusters'ages within the i^{th} processors.

Once, the processor detects a stale cluster, the stale cluster is removed automatically because it may have adverse influence on the future clustering results. After the stale clusters cleaning operation, the data stream processor will be ready for the next data sample and begin a new round of data processing and parameters updating. Based on the needs of users, the clustering results can be viewed and checked at any time. Once requested by users, all the processors will send the existing clusters and the corresponding parameters stored in their memories to the fusion center, and the fusion center will fuse all the clustering results together and give the final output. However, this user inference is entirely optional. The algorithm is designed to work fully autonomously and will perform fusion anyway unless specifically prompted not to.

3.2 Clusters Fusion Stage

In this stage, the clustering results from all the data stream processors will be fused together to generate the final output of the proposed *Parallel_TEDA* approach.

Once we get all the clusters from all the processors and re-denote their parameters as: centers $\boldsymbol{\mu}_j$, means of scalar product X_j and supports S_j, $j = 1, 2, \ldots, C_0 (C_0 = \sum_{i=1}^{N} C_i)$, the fusion process will start. The clusters fusion stage begins from the cluster with the smallest support and ends with the largest one. For each cluster, starting from its closest neighboring cluster to the farthest cluster away from it, we check the condition to decide whether this cluster should be merged with the neighboring cluster:

$$\begin{array}{c}\text{IF } (\varepsilon_l(\boldsymbol{\mu}_j) \leq \varepsilon_o) \text{ AND } (\varepsilon_j(\boldsymbol{\mu}_l) \leq \varepsilon_o) \\ \text{THEN (The } j^{th} \text{ and } l^{th} \text{ clusters should merge together)}\end{array} \tag{18}$$

where $\boldsymbol{\mu}_j$ and $\boldsymbol{\mu}_l$ are the centers of the j^{th} and l^{th} clusters, ε_i and ε_j are calculated using the following equations:

$$\begin{aligned}\varepsilon_l(\boldsymbol{\mu}_j) &= \frac{S_l^2\left\|(\boldsymbol{\mu}_j-\boldsymbol{\mu}_l)\right\|^2+(S_l+1)(S_lX_l+\left\|\boldsymbol{\mu}_j\right\|^2)-\left\|\boldsymbol{\mu}_j+S_l\boldsymbol{\mu}_l\right\|^2}{(S_l+1)(S_lX_l+\left\|\boldsymbol{\mu}_j\right\|^2)-\left\|\boldsymbol{\mu}_j+S_l\boldsymbol{\mu}_l\right\|^2}\\ \varepsilon_j(\boldsymbol{\mu}_l) &= \frac{S_j^2\left\|(\boldsymbol{\mu}_l-\boldsymbol{\mu}_j)\right\|^2+(S_j+1)(S_jX_j+\|\boldsymbol{\mu}_l\|^2)-\left\|\boldsymbol{\mu}_l+S_j\boldsymbol{\mu}_j\right\|^2}{(S_j+1)(S_jX_j+\|\boldsymbol{\mu}_l\|^2)-\left\|\boldsymbol{\mu}_l+S_j\boldsymbol{\mu}_j\right\|^2}\end{aligned} \tag{19}$$

If the merging condition is met, the two clusters merge together:

$$\begin{gathered}\boldsymbol{\mu}_{new} \leftarrow (\tfrac{S_j}{S_j+S_l}\boldsymbol{\mu}_j + \tfrac{S_l}{S_j+S_l}\boldsymbol{\mu}_l); \quad X_{new} \leftarrow (\tfrac{S_j}{S_j+S_l}X_j + \tfrac{S_l}{S_j+S_l}X_l);\\ S_{new} \leftarrow (S_j + S_l); \quad C_0 \leftarrow (C_0 - 1)\end{gathered} \tag{20}$$

After the merging, there may be some trivial clusters (with small support) left satisfying the following condition:

$$\text{IF } (S_i < \frac{\sum_{j=1}^{C_o} S_j}{5C_o}) \text{ THEN (Merge the } i^{th}\text{cluster with the nearest larger cluster)} \tag{21}$$

where the nearest larger cluster is determined as follows:

$$c_{merge} = \text{argmin}_{l=1}^{C_o}(\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_l\|); \quad l \neq i; \quad S_l \geq \frac{\sum_{j=1}^{C_o} S_j}{5C_o} \tag{22}$$

Then, Eq. (20) is used to merge the minor cluster with the larger one.

4 Parallel_TEDA Procedure

In this section, the proposed *Parallel_TEDA* approach is summarized in a form of pseudo-code in two parts according to the structure given in Fig. 1.

The first part of this approach is executed in every processor independently, namely the separate processing stage. The second part of the proposed algorithm is executed in the fusion center, namely clusters fusion stage.

1. Parallel_TEDA Algorithm Part 1:

```
– While the data sample of the $i^{th}$ data stream is available (or until nor interrupted):
  • If ($k = 1$) Then:
    $C_i \leftarrow 1$;  $\boldsymbol{\mu}_k^{i,C_i} \leftarrow \boldsymbol{x}_k^i$;  $X_k^{i,C_i} \leftarrow \left\|\boldsymbol{x}_k^i\right\|^2$;  $A_k^{i,C_i} \leftarrow 0$;  $S_k^{i,C_i} \leftarrow 1$;
  • Else:
    1. Calculate $\varepsilon_{k+1}^{i,c}(\boldsymbol{x}_k^i)$, $c = 1, 2, \ldots, C_i$ using Eqs. (5) and (7);
    2. If (Association Condition is met (Eq. (12)) Then:
       * Find the target cluster $c_{selected}$ using Eqs. (12) and (13);
       * Update $\boldsymbol{\mu}_{k+1}^{i,c_{selected}}$, $X_{k+1}^{i,c_{selected}}$, $S_{k+1}^{i,c_{selected}}$, $A_{k+1}^{i,c_{selected}}$ using Eq. (14);
       * Update the ages of other clusters using Eq. (16);
```

```
        3. Else
            * Add a new cluster and its corresponding parameters using Eq. (15);
            * Update the ages of other clusters using Eq. (16);
        4. End If
    • End If
– End While
```

2. Parallel_TEDA Algorithm Part 2:

```
– While the existing clusters exhibit the potential of merging
    • Calculate $\varepsilon_l(\boldsymbol{\mu}_j)$ and $\varepsilon_j(\boldsymbol{\mu}_l)$ using Eq. (19).
    • If (Merging Condition is met) Then:
        * Merge the two clusters using Eq. (20);
– End While
– Merge the minor clusters with the nearest larger clusters using Eqs. (21), (22)
  and (20)
```

5 Numerical Experiments

In this section, several numerical experiments using benchmark datasets from [25] are conducted to study the performance of the newly proposed *Parallel_TEDA* approach. The details of the benchmark datasets are given in Table 1.

Table 1. Benchmark dataset description

DataSet	Details				
	Number of data samples	Number of clusters	Number of attributes	Maximum cluster size	Minimum cluster size
A1	3000	20	2	150	150
A2	5250	35	2	150	150
A3	7500	50	2	150	150
S1	5000	15	2	350	300
S2	5000	15	2	350	300

The *Parallel_TEDA* approach was developed into a software which run within MATLAB R2015a. The performance was evaluated on a PC with dual core processor with clock frequency 3.6 GHz each and 8 GB RAM. The first experiment is conducted to verify the correctness and effectiveness of the *Parallel_TEDA* approach. Datasets A1 and A2 are used together to testify the ability of the parallel computation as well as handling multi-data streams. In this experiment, 11 data stream processors are involved and the data chunk size is set to be 250. The processing procedure of the data chunks of the three datasets is given in Fig. 2. The time-varying clustering results of each process cycle are presented in Fig. 3.

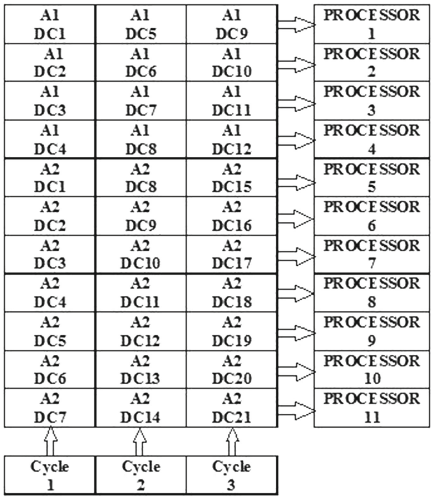
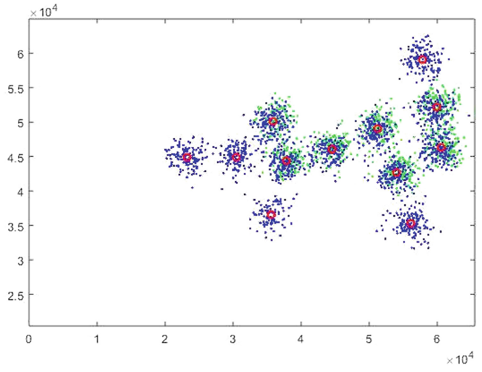


Fig. 2. The processing procedure of first experiment (DC is short for data chunk)

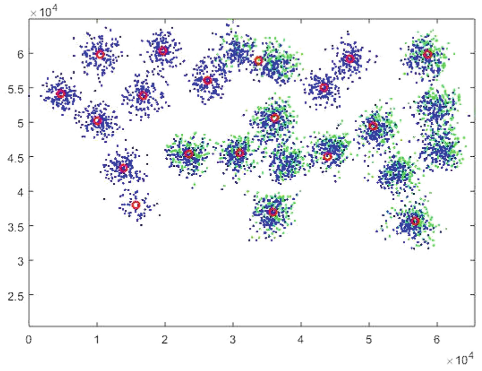
As it is clearly shown in Fig. 3, the proposed approach can successfully follow the changes of the data pattern including the shift leading to creation of new clusters (evolving the structure). Parallel processors automatically discard the stale clusters which cannot represent adequately the latest ensemble properties of the data. Moreover, it also shows that the proposed approach is capable to seamlessly handle multiple data streams.

The dataset A3 is used to investigate the relationships between execution time and the number of processors as well as the chunk size. The amount of time consumed (in seconds) for processing the dataset A3 are listed in Table 2 with different number of processors and chunk sizes. The amount of the time consumed by the two processing stages are presented separately for easier analysis. The relationship between average processing time and the number of processors is additionally depicted in Fig. 4 for visual clarity.

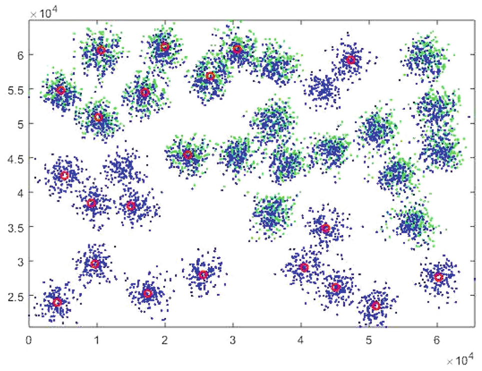
Table 2 and Fig. 4 show that, the processing speed becomes higher and the time required, respectively, becomes lower when more processors are involved. It is also interesting to notice that, when more processors are involved in the task, there will be an approximately exponential decrease of the execution time used by the first stage of the proposed approach. However, the fusion center needs time to fuse all the clusters together as there will be more clusters generated by individual processors. Correspondingly, the execution time of stage 2 of the proposed approach will increase with the larger chunk size because the larger chunk size will lead to more clusters generated in the processors in each process cycle, which in turn, also cost more time in processing the future data chunk as well as in fusing clustering results together. Nonetheless, the number of processors has the most significant influence on the overall processing speed. Combining the



(a) Process cycle 1



(b) Process cycle 2



(c) Process cycle 3

Fig. 3. The time-varying clustering result (The green dots are the data samples from A1 dataset, blue dots are the ones from A2 dataset and red circles are centers of the clusters).

Table 2. Execution time study (in seconds)

Chunk size	Stages	Number of processors						
		2	3	4	5	6	7	8
100	1	0.5746	0.4093	0.3137	0.2517	0.2097	0.1911	0.1715
	2	0.0088	0.0111	0.0105	0.0237	0.0115	0.0289	0.0132
200	1	0.6018	0.4276	0.3141	0.2698	0.2213	0.2018	0.1773
	2	0.0091	0.0118	0.0125	0.0149	0.0135	0.0158	0.0248
300	1	0.6359	0.4468	0.3317	0.2900	0.2308	0.2018	0.1829
	2	0.0100	0.0140	0.0166	0.0127	0.0162	0.0175	0.0215
400	1	0.6764	0.4739	0.3447	0.291	0.2389	0.2159	0.1916
	2	0.0222	0.0148	0.0140	0.0186	0.0166	0.0308	0.0389
500	1	0.7438	0.5316	0.3721	0.3047	0.2602	0.2196	0.1996
	2	0.0114	0.0133	0.0178	0.019	0.0418	0.0523	0.0550

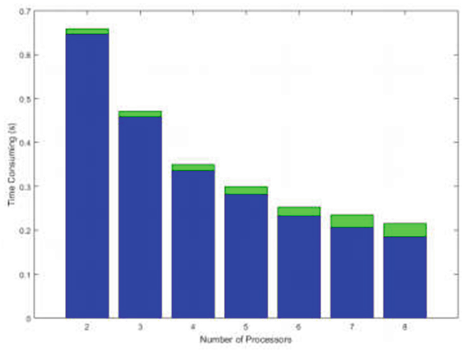


Fig. 4. The relationship between processing time and number of processors (The blue bars represent the time consumed in stage 1 and the green ones represent the time consumed in stage 2)

two stages of the approach, there is still an approximately exponential decrease of the processing time (see Fig. 4).

In order to further study the performance of the proposed approach, the well-known subtractive [6, 7] and ELM clustering approaches [10] were used for comparison purpose.

Since *Parallel_TEDA* approach is for live data streams, no processed data samples will be kept in memory, we consider the following alternative measures as indicators of clustering performances: (1) Number of clusters, C_0; (2) Maximum support of the clusters, S_{max}; (3) Minimum support of the clusters, S_{min}; and (4) Execution time, t_{exe}.

For a better comparison, in the following experiments, *Parallel_TEDA* approach will not discard old clusters, and each time these approaches only handle

Table 3. Performance comparison (PTE denotes *Parallel_TEDA*; SuC denotes subtractive clustering)

Clustering approaches	User input	Data set	Measures			
			C_0	S_{max}	S_{min}	t_{exe}
PTE	**None**	A1	**20**	**166**	**135**	**0.1388**
SuC	R=0.3		9	560	152	0.3728
	R=0.1		31	153	27	0.3445
ELM	R=600		4	2532	2	0.8820
	R=200		63	914	1	5.6551
PTE	**None**	A2	**35**	**176**	**125**	**0.2741**
SuC	R=0.3		11	712	166	0.6811
	R=0.1		35	157	143	0.6954
ELM	R=600		9	3583	1	1.7233
	R=200		105	914	1	14.8939
PTE	**None**	A3	**50**	**176**	**125**	**0.4956**
SuC	R=0.3		13	884	298	1.1141
	R=0.1		50	165	140	1.1323
ELM	R=600		4	3583	1	2.7453
	R=200		132	818	1	28.0227
PTE	**None**	S1	**15**	**359**	**294**	**0.3372**
SuC	R=0.3		10	670	323	0.6281
	R=0.1		15	355	296	0.6114
ELM	R=10000		4	2532	2	0.8820
	R=3000		63	914	1	5.6551
PTE	**None**	S2	**15**	**359**	**286**	**0.3288**
SuC	R=0.3		10	833	319	0.6313
	R=0.1		15	351	289	0.6215
ELM	R=10000		13	1700	1	2.4259
	R=3000		102	358	1	14.9862

one data stream. We use 4 processors in the proposed *Parallel_TEDA* approach in comparative experiments and the chunk size is set to 200 data samples/points. The comparison results are presented in Table 3. As it is shown in Table 3, compared with the two comparative clustering approaches, *Parallel_TEDA* approach exhibits more accurate clustering results. Note that, both subtractive and ELM clustering approaches require the radius (r or R, respectively) to be pre-defined. Moreover, its choice heavily influences the result (see Table 3). The proposed *Parallel_TEDA* approach does not require such parameter to be pre-defined. Yet, it is able to identify the correct number of clusters and is the fastest of them. The subtractive clustering approach can be comparable with the proposed approach in terms of clustering accuracy, but its performance heavily relies on the proper user input. In addition, it is offline and iterative. Moreover, even though only 4 processors are involved, *Parallel_TEDA* algorithm is already the fastest among the three. Apparently, the proposed *Parallel_TEDA* approach has clear advantages because of the high performance and the potential for even higher processing speed.

6 Conclusions

In this paper, we proposed a novel real-time clustering approach for high frequency streaming data processing, called *Parallel_TEDA*. This approach inherits the advantages of the recently introduced TEDA theoretical framework and has the ability of parallel computation due to its multi-processors structure. In addition, it can successfully follow the drifts and/or shifts in the data pattern. Within the TEDA framework, the streaming data samples are divided into a number of data chunks and sequentially sent to the data processors for clustering. Each generated cluster is additionally assigned a time tag for online cluster quality monitoring. The fusion center gathers the clustering results from each data processor and fuses them together to obtain the overall output. Numerical experiments show the superior performance of the proposed approach as well as the potential for an even higher processing speed. This approach will be a promising tool for further applications in online high frequency data processing and analysis.

Acknowledgements. The second author would like to acknowledge the partial support through The Royal Society grant IE141329/2014 *Novel Machine Learning Paradigms to Address Big Data Streams*. The third, fourth, and fifth authors would like to acknowledge the support by the Spanish Goverment under the project TRA2013-48314-C3-1-R and the project TRA2015-63708-R.

References

1. Fukunaga, K., Hostetler, L.: The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Trans. Inf. Theor. **21**(1), 32–40 (1975)
2. MacQueen, J.: Some methods for classification and analysis of multi-variate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Statistics, vol. 1, Berkeley, pp. 281–297 (1967)
3. Bezdek, J.C., Ehrlich, R., Full, W.: FCM: the fuzzy c-means clustering algorithm. Comput. Geosci. **10**(2), 191–203 (1984)
4. Johnson, S.: Hierarchical clustering schemes. Psychometrika **32**(3), 241–254 (1967)
5. de Oliveira, J.V., Pedrycz, W. (eds.): Advances in Fuzzy Clustering and Its Applications. Wiley, New York (2007)
6. Yager, R., Filev, D.: Generation of fuzzy rules by mountain clustering. J. Intell. Fuzzy Syst. **2**(3), 209–219 (1994)
7. Chiu, S.L.: Fuzzy model identification based on cluster estimation. J. Intell. Fuzzy Syst. **2**(3), 1064–1246 (1994)

8. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: 2nd International Conference on Knowledge Discovery and Data Mining, vol. 96(34), Portland, Oregan, pp. 226–231 (1996)
9. Wang, C., Lai, J., Huang, D., Zheng, W.: SVStream: a support vector-based algorithm for clustering data streams. IEEE Trans. Knowl. Data Eng. **25**(6), 1410–1424 (2013)
10. Baruah, R., Angelov, P.: Evolving local means method for clustering of streaming data. In: IEEE Congress on Computational Intelligence, Brisbane, Australia, pp. 2161–2168 (2012)
11. Hyde, R., Angelov, P.: A fully autonomous data density based clustering technique. In: IEEE Symposium on Evolving and Autonomous Learning Systems, Orlando, USA, pp. 116–123 (2014)
12. Angelov, P., Gu, X., Gutierrez, G., Iglesias, J.A., Sanchis, A.: Autonomous data density based clustering method. In: 2016 IEEEWorld Congress on Computational Intelligence, Vancouver, Canada, pp. 2405-2413 (2016)
13. Guha, S., Mishra, N., Motwani, R., O'Callaghan, L.: Clustering data streams. In: Proceedings of the Annual Symposium on Foundations of Computer Scuebce (FOCS), Redondo Beach, CA, pp. 359–366 (2000)
14. Guha, S., Meyerson, A., Mishra, N., Motwani, R., O'Callaghan, L.: Clustering data streams: theory and practice. IEEE Trans. Knowl. Data Eng. **15**(3), 515–528 (2003)
15. Aggarwal, C., Han, J., Wang, J., Yu, S.: A framework for clustering evolving data streams. In: Proceedings of the International Conference on Very Large Data Bases, Berlin, Germany, pp. 81–92 (2003)
16. Comode, G., Muthukrishnan, S., Zhang, W.: Conquering the divide: continuous clustering of distributed data streams. In: Proceedings of the International Conference on Data Engineering, Istanbul, pp. 1036–1045 (2007)
17. Gama, J., Rodrigues, P., Sebastio, R.: Evaluating algorithms that learn from data streams. In: Proceedings of the ACM Symposium on Applied Computing, Hawaii, pp. 1496–1500 (2009)
18. Angelov, P.: Outside the box: an alternative data analytics framework. J. Autom. Mob. Rob. Intell. Syst. **8**(2), 53–59 (2014)
19. Angelov, P.: Typicality distribution function - a new density-based data analytics tool. In: IEEE International Joint Conference on Neural Networks (IJCNN), Killarney, pp. 1–8 (2015)
20. Angelov, P., Gu, X., Kangin, D., Principe, J.: Empirical data analysis: a new tool for data analytics. In: 2016 IEEE International Conference on Systems, Man, and Cybernetics, Budapest, Hungary (2016, to appear)
21. Angelov, P., Filev, D.: Simple_TS: a simplified method for learning evolving TakagiSugeno fuzzy models. In: IEEE International Conference on Fuzzy Systems, Reno, USA, pp. 1068–1073 (2005)
22. Angelov, P.: Autonomous Learning Systems from Data Stream to Knowledge in Real Time. John Wiley & Sons, Ltd., West Sussex (2012)
23. Lughofer, E., Angelov, P.: Handling drifts and shifts in on-line data streams with evolving fuzzy systems. Appl. Soft Comput. J. **11**(2), 2057–2068 (2011)
24. Saw, J., Yang, M., Mo, T.: Chebyshev inequality with estimated mean and variance. Am. Stat. **38**(2), 130–132 (1984)
25. Clustering Datasets - University of Eastern Finland (2016). http://cs.joensuu.fi/sipu/datasets/. Access 12 May 2016

A Big Data Intelligent Search Assistant Based on the Random Neural Network

Will Serrano(✉)

Intelligent Systems and Networks Group, Electrical and Electronic Engineering, Imperial College London, London, UK
g.serrano11@imperial.ac.uk

Abstract. The need to search for specific information or products in the ever expanding Internet has led the development of Web search engines and recommender systems. Whereas their benefit is the provision of a direct connection between users and the information or products sought within the Big Data, any search outcome will be influenced by a commercial interest as well as by the users' own ambiguity in formulating their requests or queries. This research analyses the result rank relevance provided by the different Web search engines, metasearch engines, academic databases and recommender systems. We propose an Intelligent Internet Search Assistant (ISA) that acts as an interface between the user and Big Data search engines. We also present a new relevance metric which combines both relevance and rank. We use this metric to validate and compare the performance of our proposed algorithm against other search engines and recommender systems. On average, our ISA outperforms other search engines.

Keywords: Intelligent Internet Search Assistant · Random Neural Network · Web search · Search Engines · Recommender Systems · Big Data

1 Introduction

The extensive size of the Big Data does not allow Internet users to find all relevant information or products without the use of Web Search Engines or Recommender Systems. Web users can not be guaranteed that the results provided by search applications are either exhaustive or relevant to their search needs. Businesses have the commercial interest to rank higher on results or recommendations to attract more customers while Web search engines and recommender systems make their profit based on their advertisements and product purchase. The main consequence is that irrelevant results or products may be shown on top positions and relevant ones "hidden" at the very bottom of the search list. As the size of the Internet and Big Data increasingly expands, Web Users are more and more dependent on information filtering applications.

We describe the application of neural networks in recommender systems in Sect. 2. In order to address the presented search issues; this paper proposes in Sect. 3 an Intelligent Internet Search Assistant (ISA) that acts as an interface between an individual user's query and the different search engines. We have validated our ISA against other Web search engines and metasearch engines, online databases and recommender systems on Sect. 4. Our conclusions are presented on Sect. 5.

2 Related Work

The ability of neural networks to learn iteratively from different inputs to acquire the desired outputs as a mechanism of adaptation to users' interest in order to provide relevant answers have already been applied in the World Wide Web and recommender systems. S. Patil et al. [1] propose a recommender system using collaborative filtering mechanism with k-separability approach for Web based marketing. They build a model for each user on several steps: they cluster a group of individuals into different categories according to their similarity using Adaptive Resonance Theory (ART) and then they calculate the Singular Value Decomposition matrix. M. Lee et al. [2] propose a new recommender system which combines collaborative filtering with a Self-Organizing Map neural network. They segment all users by demographic characteristics where users in each segment are clustered according to the preference of items using the neural network. C. Vassiliou et al. [3] propose a framework that combines neural networks and collaborative filtering. Their approach uses a neural network to recognize implicit patterns between user profiles and items of interest which are then further enhanced by collaborative filtering to personalized suggestions. K. Kongsakun et al. [4] develop an intelligent recommender system framework based on an investigation of the possible correlations between the students' historic records and final results. C. Chang et al. [5] train the artificial neural networks to group users into different types. They use an Adaptive Resonance Theory (ART) neural network model in an unsupervised learning model where the input layer is a vector made of user's features and the output layer is the different cluster. P. Chou et al. [6] integrate a back propagation neural network with supervised learning and feed forward architecture in an "interior desire system". D. Billsus et al. [7] propose a representation for collaborative filtering tasks that allows the application of any machine learning algorithm, including a feed forward neural network with k input neurons, 2 hidden neurons and 1 output neuron. M. Krstic et al. [8] apply a single hidden layer feed forward neural network as a classifier tool which estimates whether a certain TV programme is relevant to the user based on the TV programme description, contextual data and the feedback provided by the user. C. Biancalana et al. [9] propose a neural network to include contextual information on film recommendations. The aim of the neural network is to identify which member of a household gave a specific rating to a film at a specific time. M. Devi et al. [10] use a probabilistic neural network to calculate the rating between users based on the rating matrix. They smooth the sparse rating matrix by predicting the rating values of the unrated items.

3 The Intelligent Internet Search Assistant Model

The search assistant we design is based on the Random Neural Network (RNN) [11–13]. This is a spiking recurrent stochastic model for neural networks. Its main analytical properties are the "product form" and the existence of the unique network steady state solution. It represents more closely how signals are transmitted in many biological neural networks where they actual travel as spikes or impulses, rather than as analogue signal levels, and has been used in different applications including network routing with cognitive packet networks, using reinforcement learning, which requires the search for paths that meet certain pre-specified quality of service requirements [14], search for exit routes for evacuees in emergency situations [15, 16] and network routing [17], pattern based search for specific objects [18], video compression [19], image texture learning and generation [20] and Deep Learning [21].

Gelenbe, E. et al. have investigated different search models [22–24]. In the case of our own application of the RNN [25]; our ISA acquires a query from the user and retrieves results from one or various search engines assigning one neuron per each Web result dimension. The result relevance is calculated by applying our innovative cost function based on the division of a query into a multidimensional vector weighting its dimension terms with different relevance parameters. Our ISA adapts and learns the perceived user's interest and reorders the retrieved snippets based in our dimension relevant centre point. Our ISA learns result relevance on an iterative process where the user evaluates directly the listed results. We evaluate and compare its performance against other search engines with a new proposed quality definition, which combines both relevance and rank. We have also included two learning algorithms; Gradient Descent learns the centre of relevant dimensions and Reinforcement Learning updates the network weights based on rewarding relevant dimensions and punishing irrelevant ones.

4 Validation

4.1 Web Search Validation

We can affirm the superior search engine has the higher density of better scoring results on top positions based on the result master list. In order to measure numerically Web search quality or to establish a benchmark from we can compare search performance; we propose the following algorithm, where results showed at top positions are rewarded and results showed at lower positions are penalized. We define quality, Q, as:

$$Q = \sum_{\text{result}=1}^{Y} \text{RML} * \text{RSE} \tag{1}$$

where RML is the rank of the result in the master list that represents the optimum result relevance order, RSE is the rank of the same result in a particular search engine and Y is the number of results shown to the user, if the result order is larger than Y, we discard the result in our calculation as it is considered irrelevant.

We define normalized quality, $\overline{Q}$, as the division of the quality, Q, by the optimum figure which it is when the results provided are ranked in the same order as in the master list; this value corresponds to the sum of the squares of the first Y integers:

$$\overline{Q} = \frac{Q}{\frac{Y(Y+1)(2Y+1)}{6}} \tag{2}$$

where Y the total number of results shown to the user.

The Intelligent Internet Search Assistant we have proposed emulates how Web search engines work by using a very similar interface to introduce and display information. We validate our proposed ISA with current Metasearch engines, we retrieve the results from the Web search engines they use to generate the result master list and then compare the results provided by the Metasearch engines against this result master list. This proposed method has the inconvenience that we are not considering any result obtained from Internet Web directories neither Online databases from where Metasearch engines may have retrieved some results displayed. We have selected both Ixquick and Metacrawler as the Metasearch engines we can compare our ISA. After analysing the main characteristics of both Metasearch engines we consider Metacrawler uses (Google Yahoo and Yandex) and Ixquick uses (Google Yahoo and Bing) as their main source of search results. We have run our ISA to acquire 10 different queries based on the travel industry from a user. The ISA then retrieves the first 30 results from each of the main Web search engine driver programmed (Google, Yahoo, Bing, and Yandex), we have therefore scored 30 points to the Web site result that is displayed in the top position, 1 point to the Web site result that is shown in the last position and 0 points to each of the result that belongs to the same Web site and it is shown more than once. After we have scored the 120 results provided by the 4 different Web search engines, we combine them by adding the scores of the results which have the same Web site and rank them to generate the result master list. We have done this evaluation exercise for each high level query. We then retrieve the first 30 results from Metacrawler and Ixquick and benchmark them against the result master list using the Quality formula proposed. We present the average Quality values for the 10 different queries on the below table (Table 1).

Table 1. Web search validation

10 Queries						
Google	Yahoo	Bing	Yandex	MetaC	Ixquick	ISA
0.67	0.66	0.66	0.68	0.59	0.42	0.65

4.2 Online Academic Database Validation

In order to measure search quality we can affirm a better Online Academic Database provides with a list of more relevant results on top positions. We propose the following quality description where within a list of N results we score N to the first result and 1 to the last result, the value of the quality proposed is then the summation of the position score based of each of the selected results. Our definition of Quality, Q, can be defined as:

$$Q = \sum_{i=1}^{Y} RSE_i \tag{3}$$

where RSE_i is the rank of the result i in a particular search engine with a value of N if the result is in the first position and 1 if the result is the last one. Y is the total number of results selected by the user. The best Online Academic Database would have the largest Quality value. We define normalized quality, $\overline{Q}$, as the division of the quality, Q, by the optimum figure which it is when the user consider relevant all the results provided by the Web search engine. On this situation Y and N have the same value:

$$\overline{Q} = \frac{Q}{\frac{N(N+1)}{2}} \tag{4}$$

We define I as the quality improvement between our Intelligent Search Assistant against an Online Academic Database:

$$I = \frac{QW - QR}{QR} \tag{5}$$

Where I is the Improvement, QW is the quality of the Intelligent Search Assistant and QR is the quality reference; we use the Quality of Google Scholar, IEEE Xplore, CiteseerX or Microsoft Academic as QR in our validation exercise in the first iteration and on further iterations, we use as Quality reference the value of the previous iteration.

Our Intelligent Internet Search Assistant can select between the main Online Academic (Google Scholar, IEEE Xplore, CiteseerX or Microsoft Academic) and the type of learning to be implemented. Our ISA then collects the first 50 results from the search engine selected, reorders them according to its cost function and finally shows to the user the first 20 results. Our ISA reorders results while learning on the two step iterative process showing only the best 20 results to the user. We have searched for 6 different queries. We have used the four different Online Academic Databases for each query, 24 searches in total. We have selected Gradient Descent and Reinforcement Learning for 3 queries (12 searches) each. The table shows the average Quality value of the Database search engine and ISA. The first I represents the improvement from ISA against the Online Academic Databases; the second I is between ISA iterations 2 and 1 and finally the third I is between the ISA iterations 3 and 2 (Table 2).

Table 2. Online academic database validation

Gradient Descent Learning: 12 Queries								
Iteration 1			Iteration 2			Iteration 3		
Web	ISA	I	Web	ISA	I	Web	ISA	I
0.44	0.56	29 %	0.48	0.64	13 %	0.50	0.66	3.6 %
Reinforcement Learning: 12 Queries								
Iteration 1			Iteration 2			Iteration 3		
Web	ISA	I	Web	ISA	I	Web	ISA	I
0.41	0.51	25 %	0.44	0.61	19 %	0.46	0.64	5.2 %

4.3 Recommender System Validation

We have implemented our Intelligent Search Assistant to reorder the results from three different independent recommender systems: GroupLens film database, Trip Advisor and Amazon. Our ISA reorders the films or products based on the updated result relevance calculated by combining only the value of the relevant selected dimensions. The higher the value the more relevant the film or product should be. ISA shows to the user the first 20 results including its ranking. The user then selects the films or products with higher ranking; this ranking has been previously calculated by adding user reviews to the same products and calculating the average value. We have included Gradient Descent and Reinforcement Learning for different queries in our validation. Experimental results are shown on the following table (Table 3).

Table 3. Recommder system validation

GroupLens film dataset - Gradient Descent Learning - 5 Queries			
First (Q)	Iteration 1 (I)	Iteration 2 (I)	Iteration 3 (I)
0.71	17.45 %	0.145 %	1.79 %
GroupLens film dataset - Reinforcement Learning - 5 Queries			
0.76	5.26 %	9.14 %	6.05 %
Trip advisor car dataset - Gradient Descent Learning - 5 Queries			
0.94	0.0328 %	0.017 %	0.0007 %
Trip advisor car dataset - Reinforcement Learning - 5 Queries			
0.94	0.798 %	0.004 %	0.0165 %
Trip advisor hotel dataset - Gradient Descent Learning - 5 Queries			
0.94	0.54 %	–0.0607 %	–0.0395 %
Trip advisor hotel dataset - Reinforcement Learning - 5 Queries			
0.94	0.728 %	4.658 %	0.139 %
Amazon dataset - Gradient Descent Learning - 5 Queries			
0.15	33.89 %	8.39 %	–6.97 %
Amazon dataset - Reinforcement Learning -5 Queries			
First (Q)	Iteration 1 (I)	Iteration 2 (I)	Iteration 3(I)
0.15	30.36 %	13.05 %	0.503 %

5 Conclusions

We have proposed a novel approach to Web search and recommendation systems within Big Data; the user iteratively trains the neural network while looking for relevant results. We have also defined a different process; the application of the Random Neural Network as a biological inspired algorithm to measure both user relevance and result ranking based on a predetermined cost function. Our Intelligent Search Assistant performs generally slightly better than Google and other Web search engines however, this evaluation may be biased because users tend to concentrate on the first results provided which were the ones we showed in our algorithm. Our ISA adapts and learns from user previous relevance measurements increasing significantly its quality and improvement within the first iteration. Reinforcement Learning algorithm performs better than Gradient Descent. Although Gradient Descent provides a better quality on the first iteration; Reinforcement Learning outperforms on the second one due its higher learning rate. Both of them have a residual learning on their third iteration. Gradient Descent would have been the preferred learning algorithm if only one iteration is required; however Reinforcement Learning would have been a better option in the case of two iterations. It is not recommended three iterations because learning is only residual.

Acknowledgment. This research has used Groupfilms dataset from the Department of Computer Science and Engineering at the University of Minnesota; Trip Advisor dataset from the University of California-Irvine, Machine Learning repository, Centre for Machine Learning and Intelligent Systems and Amazon dataset from Julian McAuley Computer Science Department at University of California, San Diego.

References

1. Patil, S., Mane, Y., Dabre, K., Dewan, P., Kalbande, D.: An efficient recommender system using collaborative filtering methods with k-separability approach. Int. J. Eng. Res. Appl., 30–35 (2012)
2. Lee, M., Choi, P., Woo, Y.T.: A hybrid recommender system combining collaborative filtering with neural network. In: de Bra, P., Brusilovsky, P., Conejo, R. (eds.) AH 2002. LNCS, vol. 2347, pp. 531–534. Springer, Heidelberg (2002)
3. Vassiliou, C., Stamoulis, D., Martakos, D., Athanassopoulos, S.: A recommender system framework combining neural networks & collaborative filtering. In: International Conference on Instrumentation, Measurement, Circuits and Systems, pp. 285–290 (2006)
4. Kongsakun, K., Kajornrit, J., Fung, C.: Neural network modelling for an intelligent recommendation system supporting SRM for universities in Thailand. In: International Conference on Computing and Information Technology, vol. 2, pp. 34–44 (2013)
5. Chang, C., Chen, P., Chiu, F., Chen, Y.: Application of neural networks and Kanos's method to content recommendation in Web personalization. Expert Syst. Appl. **36**, 5310–5316 (2009)
6. Chou, P., Li, P., Chen, K., Wu, M.: Integrating Web mining and neural network for personalized e-commerce automatic service. Expert Syst. Appl. **37**, 2898–2910 (2010)

7. Billsus, D., Pazzani, M.: Learning collaborative information filters. In: International Conference of Machine Learning, pp. 46–54 (1998)
8. Krstic, M., Bjelica, M.: Context aware personalized program guide based on neural network. IEEE Trans. Consum. Electron. **58**, 1301–1306 (2012)
9. Biancana, C., Gaspareti, F., Micarelli, A., Miola A., Sansonetti, G.: Context-aware movie recommendation based on signal processing and machine learning. In: The Challenge on Context Aware Movie Recommendation, pp. 5–10 (2011)
10. Devi, M., Samy, R., Kumar, S., Venkatesh, P.: Probabilistic neural network approach to alleviate sparsity and cold start problems in collaborative recommender systems. Comput. Intell. Comput. Res., 1–4 (2010)
11. Gelenbe, E.: Random neural network with negative and positive signals and product form solution. Neural Comput. **1**, 502–510 (1989)
12. Gelenbe, E.: Learning in the recurrent Random Neural Network. Neural Comput. **5**, 154–164 (1993)
13. Gelenbe, E., Timotheou, S.: Random neural networks with synchronized interactions. Neural Comput. **20**(9), 2308–2324 (2008)
14. Gelenbe, E., Lent, R., Xu, Z.: Towards networks with cognitive packets. In: Goto, K., Hasegawa, T., Takagi, H., Takahashi, Y. (eds.) Performance and QoS of Next Generation Networking, pp. 3–17. Springer, London (2011)
15. Gelenbe, E., Wu, F.J.: Large scale simulation for human evacuation and rescue. Comput. Math Appl. **64**(12), 3869–3880 (2012)
16. Filippoupolitis, A., Hey, L., Loukas, G., Gelenbe, E., Timotheou, S.: Emergency response simulation using wireless sensor networks. In: Proceedings of the 1st International Conference on Ambient Media and Systems, p. 21 (2008)
17. Gelenbe, E.: Steps towards self-aware networks. Commun. ACM **52**(7), 66–75 (2009)
18. Gelenbe, E., Koçak, T.: Area-based results for mine detection. IEEE Trans. Geosci. Remote Sens. **38**(1), 12–24 (2000)
19. Cramer, C., Gelenbe, E., Bakircloglu, H.: Low bit-rate video compression with neural networks and temporal subsampling. Proc. IEEE **84**(10), 1529–1543 (1996)
20. Atalay, V., Gelenbe, E., Yalabik, N.: The random neural network model for texture generation. Int. J. Pattern Recognit Artif Intell. **6**(1), 131–141 (1992)
21. Gelenbe, E., Yin, Y.: Deep learning with random neural networks, In: International Joint Conference on Neural Networks (IJCNN 2016) World Congress on Computational Intelligence. IEEE Xplore, Vancouver (2016). Paper Number 16502
22. Gelenbe, E.: Search in unknown random environments. Phys. Rev. E **82**(6), 061112 (2007)
23. Gelenbe, E., Abdelrahman, O.H.: Search in the universe of big networks and data. IEEE Netw. **28**(4), 20–25 (2014)
24. Abdelrahman, O.H., Gelenbe, E.: Time and energy in team-based search. Phys. Rev. E **87**, 032125 (2013)
25. Gelenbe, E., Serrano, W.: An intelligent internet search assistant based on the random neural network. In: Iliadis, L., Maglogiannis, I. (eds.) AIAI 2016, IFIP AICT, vol. 475, pp. 141–153. Springer, Switzerland (2016)

RandomFIS: A Fuzzy Classification System for Big Datasets

Oscar Samudio, Marley Vellasco(✉), Ricardo Tanscheit, and Adriano Koshiyama

Department of Electrical Engineering, Pontifical Catholic University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil
{oscarsam,marley,ricardo}@ele.puc-rio.br, as.koshiyama@gmail.com

Abstract. One of the main advantages of fuzzy classifier models is their linguistic interpretability, revealing the relation between input variables and the output class. However, these systems suffer from the curse of dimensionality when dealing with high dimensional problems (large number of attributes and instances). This paper presents a new fuzzy classifier model, named RandomFIS, that provides good performance in both classification accuracy and rule base interpretability even when dealing with databases comprising large numbers of inputs (attributes) and patterns (instances). RandomFIS employs concepts from *Random Subspace* and *Bag of little Bootstrap* (BLB), resulting in an ensemble of fuzzy classifiers. It was tested with different classification benchmarks, proving to be an accurate and interpretable model, even for problems involving big databases.

Keywords: Fuzzy inference system · Classification · Bootstrapping · RandomSubspace · Bag of little Bootstrap

1 Introduction

Classification systems based on fuzzy rules are useful and well known tools for representing and extracting knowledge from data bases involving uncertainty, inaccuracy and nonlinearity [1]. Approaches based on fuzzy logic have the ability to provide accurate models and, at the same time, interpretable linguistic rules to inform the end user of the relationship between input variables and output classes [2]. In order to build models that consider both accuracy and interpretability, most studies have used (i) evolutionary algorithms [3] to elaborate fuzzy rules and (ii) Evolving Fuzzy Inference Systems [4] to create and adapt a fuzzy rule base by gathering new observations.

In [5], a simpler approach, known as AutoFIS-Class, was proposed to automatically generate a Fuzzy Classifier System that provides good accuracy but also favors linguistic interpretability. AutoFIS-Class was compared to other fuzzy-evolutionary models [5] and performed very well in terms of accuracy and interpretability. However, for big data bases AutoFIS-Class suffers from the *curse of dimensionality* [6], especially with respect to computational time and memory consumption.

In order to improve the performance of AutoFIS-Class in classification problems involving big data bases, this paper proposes a new model, named *RandomFIS*, which is based on a combination of multiple fuzzy classifiers [1], also known as ensembles [7], and techniques such as *Bootstrapping* (*Bagging*), and *Random Subspace* [8]. RandomFIS automatically extracts fuzzy rules from big databases comprising large numbers of inputs (attributes) and patterns (instances). It makes use of concepts from *Random Subspace* and *Bag of little Bootstrap* (BLB) [9], which is a scalable version of *Bootstrapping*, resulting in an ensemble of classifiers. This combination provides an interpretable as well as an accurate classifier model.

This paper is organized into three additional sections. Section 2 describes in details the proposed model and how it handles high dimensional datasets. Section 3 presents the experiments carried out for different datasets. Finally, Sect. 4 concludes the work and suggests some lines for future work.

2 RandomFIS

The RandomFIS model is an extension of AutoFIS, specially developed to deal with big databases, whether in terms of number of attributes or of instances (patterns). The main objective is to obtain a scalable classification fuzzy inference system that provides good accuracy but maintains the necessary interpretability.

In order to deal with big databases, RandomFIS is based on the creation of multiple classifiers, developed from databases with reduced number of attributes and instances. It consists of the main blocks shown in Fig. 1.

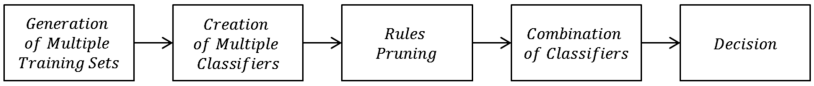

Generation of Multiple Training Sets
Creation of Multiple Classifiers
Rules Pruning
Combination of Classifiers
Decision

Fig. 1. The main blocks of the RandomFIS model

2.1 Generating Multiple Training Sets

In this module, RandomFIS creates different datasets, composed of a reduced amount of instances and variables from the original database, to train multiple classifiers, as illustrated in Fig. 2. The generation of those multiple datasets makes use of concepts from *Bag of Little Bootstrap* (BLB) and *RandomSubspace*.

The generation of multiple training sets employs 2 levels of processing: *Subsampling* and *Resampling,* with random selection of a subset of the input attributes. The first level uses the original database of size n and generates s subsets of size b, so that

$$b = n^{\gamma} \tag{1}$$

where n is the number of patterns (instances) in the original database, b is the size of the reduced dataset and γ is a constant ($0.5 < \gamma \leq 0.9$). The *Subsampling* technique is used to generate s sets by randomly selecting patterns based on an uniform distribution

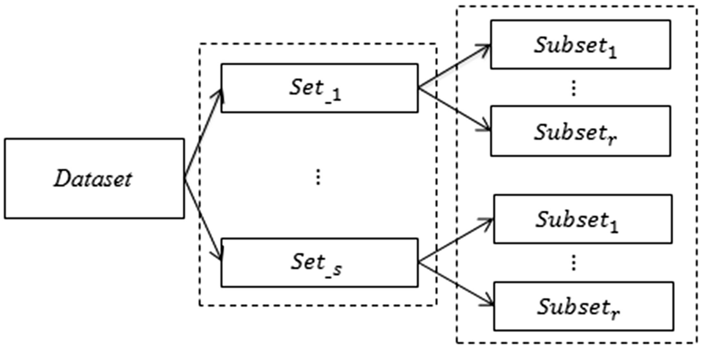


Fig. 2. Process of generating multiple training datasets

without repetition. The second level generates, from each *Set*$_i$, $i \in \{1, \ldots s\}$, a *Bootstrapping* (*Resampling*) *Subset*$_j$, $j \in \{1, \ldots, r\}$, now with repetition. Each of the *r* subsets contains a reduced number (J^*) of randomly selected attributes (known as *RandomSubspace*), given by Eq. (2):

$$J^* = \lfloor \log_2(J) + 1 \rfloor \tag{2}$$

where *J* is the total number of attributes in the original database and *J** is the number of variables that are selected in each *Subset*$_j$.

2.2 Creation of Multiple Classifiers

The following sections describe the steps taken in the creation of *s* groups of *r* classifiers. Each of these is trained with one *Subset*$_j$, following the procedure established for the AutoFIS model [5].

2.2.1 Fuzzification

In classification, the main information consists of *n* patterns $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{iJ}]$ of *J* attributes X_j present in the database $(i = 1, \ldots, n e\ j = 1, \ldots, J)$. A number of *L* fuzzy sets $A_{jl} = \left\{ \left(x_{ij}, \mu_{A_{jl}}(x_{ij}) \right) | x_{ij} \in X_j \right\}$ is associated to each j^{th} attribute, where $\mu_{A_{jl}} : X_j \rightarrow [0, 1]$ is a membership function that associates to each observation x_{ij} a membership degree $\mu_{A_{jl}}(x_{ij})$ to the fuzzy set A_{jl}. Each pattern is associated to a class C_i out of *K* possible ones, that is $C_i \in \{1, 2, \ldots k, \ldots, K\}$. The Fuzzification stage takes into account three aspects: membership function format, the support of each member function $\mu_{A_{jl}}(x_{ij})$ and the appropriate linguistic label. RandomFIS uses the Tukey approach [5], which considers the information from the quartiles.

2.2.2 Rules Extraction

This stage consists of *Formulation* of the rules' premises and *Association* of each premise to a specific consequent.

A. Formulation: This first step consists of building rule premises. In mathematical terms, a premise is usually defined as:

$$\mu_{A_d}(\mathbf{x}_i) = \mu_{A_{1l}}(x_{i1}) * \ldots * \mu_{A_{1l}}(x_{ij}) * \ldots * \mu_{A_{Jl}}(x_{iJ}) \tag{3}$$

where $\mu_{A_d}(\mathbf{x}_i)$ is the joint membership degree of pattern i in premise d, $(d = 1, \ldots, D)$, and $*$ is a t-norm. More generally, a premise can be built from a combination of the $\mu_{A_{1l}}(x_{ij})$ through the use of t-norms, t-conorms, negation operators and linguistic hedges. The negation operator acts upon each element in a premise.

Regarding interpretability, it is desirable to have few rules with few antecedent elements. Thus, a limit is imposed on the maximum size of premises and these are generated in an organized way: initialization with size-1 premises, creation of size-2 premises from the size-1 viable ones, generation of size-3 premises from viable size-2 premises and so on. Besides, a premise viability is evaluated through a set of filters: support, similarity and conflict in classification.

(a) Support Filter: aims at building premises that cover a large number of patterns in the database. The Relative Support of a premise $\mu_{A_d}(\mathbf{x}_i)$ is given by:

$$Sup_d = \frac{\sum\limits_{i=1}^{n} \mu_{A_d}(\mathbf{x}_i)}{n} \tag{4}$$

Given a user-defined tolerance ε_{Sup}, a premise is deemed viable if $Sup_d > \varepsilon_{Sup}$.

(b) Similarity Filter: its objective is to reduce the occurrence of similar or identical premises. Given two premises $\mu_{A_d}(\mathbf{x}_i)$ and $\mu_{A_v}(\mathbf{x}_i)$, the similarity between them can be computed by:

$$Sim_{d,v} = \frac{\sum_{i=1}^{n} min\{\mu_{A_d}(\mathbf{x}_i), \mu_{A_v}(\mathbf{x}_i)\}}{\sum_{i=1}^{n} max\{\mu_{A_d}(\mathbf{x}_i), \mu_{A_v}(\mathbf{x}_i)\}} \tag{5}$$

Given a user-defined tolerance ε_{sim}, two premises are similar if $Sim_{d,v} > \varepsilon_{sim}$. If the similarity is identified, the premise with the lower Relative Support is removed.

(c) *PCD Filter:* this filter aims at reducing the occurrence of similar or conflicting rules by computing the Penalized Confidence Degree (PCD_k) [10]:

$$PCD_k = max\left\{\frac{\sum_{i \in k} \mu_{A_d}(\mathbf{x}_i) - \sum_{i \notin k} \mu_{A_d}(\mathbf{x}_i)}{\sum_{i=1}^{n} \mu_{A_d}(\mathbf{x}_i)}, 0\right\} \tag{6}$$

A premise is not viable if $PCD_k = 0$ for all K classes.

B. Association: This step determines the consequent class that is most compatible to a given premise $\mu_{A_d}(\mathbf{x}_i)$. Premise d associated to class k (i.e. a fuzzy rule) is denoted by $\mu_{A_{d(k)}}(\mathbf{x}_i)$, which describes, in linguistic terms:

$$``If\ X_1\ is\ A_{1l}\ and \ldots and\ X_j\ is\ A_{jl}\ and \ldots and\ A_j\ is\ A_{jl},\ then\ \mathbf{x}_i\ is\ Class\ k"$$

The Weighted Average estimated through Restricted Least Squares (RLS) is used here. For each d^{th} premise, a linear configuration of weights in the range [0, 1] is determined. This can be formulated as:

$$\begin{aligned} \min \sum\nolimits_{i=1}^{n} \left(\mu_{C_i \in k}(\mathbf{x}_i) - \sum\nolimits_{k=1}^{K} \beta_k \mu_{A_d}(\mathbf{x}_i) \right)^2 \\ subject \sum\nolimits_{k=1}^{K} \beta_k = 1 \text{ e } \beta_k \geq 0 \end{aligned} \tag{7}$$

where $\mu_{C_i \in k}(\mathbf{x}_i) \in \{0, 1\}$ is the membership degree of pattern $\mathbf{x}_i$ to class k (binary representation of class k), β_k is the degree of influence of class k on premise $\mu_{A_d}(\mathbf{x}_i)$. If $\beta_k = 0$, then premise $\mu_{A_d}(\mathbf{x}_i)$ is eliminated.

2.3 Rules Pruning

As mentioned in Sect. 2.1, each Set_i, $i \in \{1, \ldots s\}$ generates r fuzzy classifiers, trained with a different $Subset_j$, $j \in \{1, \ldots, r\}$ that contains a reduced number of attributes. In this module all rules generated by the r classifiers, obtained from Set_i, are inspected and combined to obtain a more compact rule base for each class k.

In the pruning process, rule bases created by each classifier, $RB_1, RB_2, \ldots, RB_r$ are grouped according to the rule's consequent (class) and identical rules are eliminated. Figure 3 exemplifies this process. From a certain Set_i dataset, r independent fuzzy rule bases are generated. Rules in blue, associated to Class 1, are identical, as well as the one in red associated to Class 2. Only one instance of blue rules and red rules are considered in the final rule base extracted from dataset Set_i. This process is performed for each dataset Set_i, $i = \{1, \ldots s\}$, resulting in s different classifiers, comprising fuzzy rules for each class k.

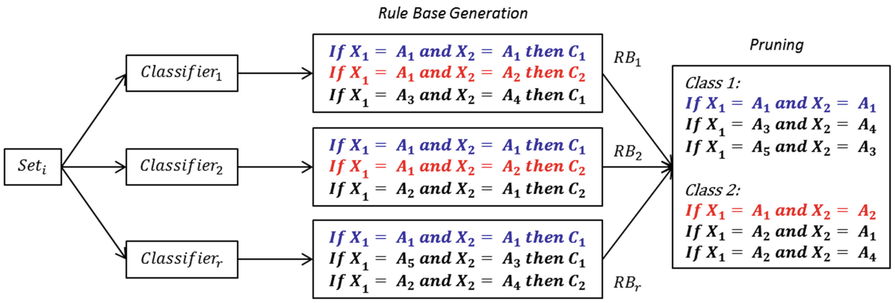

Rule Base Generation
Pruning
Set_i
Classifier_1
Classifier_2
Classifier_r
If X_1 = A_1 and X_2 = A_1 then C_1
If X_1 = A_1 and X_2 = A_2 then C_2
If X_1 = A_3 and X_2 = A_4 then C_1
If X_1 = A_1 and X_2 = A_1 then C_1
If X_1 = A_1 and X_2 = A_2 then C_2
If X_1 = A_2 and X_2 = A_1 then C_2
If X_1 = A_1 and X_2 = A_1 then C_1
If X_1 = A_5 and X_2 = A_3 then C_1
If X_1 = A_2 and X_2 = A_4 then C_2
RB_1
RB_2
RB_r
Class 1:
If X_1 = A_1 and X_2 = A_1
If X_1 = A_3 and X_2 = A_4
If X_1 = A_5 and X_2 = A_3
Class 2:
If X_1 = A_1 and X_2 = A_2
If X_1 = A_2 and X_2 = A_1
If X_1 = A_2 and X_2 = A_4

Fig. 3. Example of the Rules Pruning Process

2.4 Combination of Classifiers

Different methods have been proposed to combine the results of an ensemble of classifiers [7]. Two methods are proposed here to combine the ensemble composed of s fuzzy classifiers: (i) Combination by Average (*RandomFIS_Ave*) and (ii) Combination by Rules (*RandomFIS_Rules*).

In *Combination by Average*, rules are aggregated per classifier, i.e. the membership degree of $\mathbf{x}_i^*$ to each class k is computed for each of the s classifiers and then divided by the number of classifiers in the ensemble. With the predicted average membership degree of pattern $\mathbf{x}_i^*$ for each one of the K available classes in the dataset, the *Decision* block of RandomFIS model defines which class $\mathbf{x}_i^*$ belongs to.

The *Combination by Rules* method, on the other hand, benefits from the diversity of the classifiers generated from *subsampling*, *bootstrapping* and *randomsubspace* subsets. In this case, all rules related to one class, from all different s classifiers, are aggregated before entering the *Decision* stage.

The aggregation procedure for both cases is described below.

2.5 Rule Aggregation

Given a new pattern $\mathbf{x}_i^*$, the aggregation stage combines the activation degree of rules related to a same class. Consider $D^{(k)}$ as the number of rules related to class k. Given an aggregation operator [11], $g{:}[0,1]^{D^{(k)}}$, the membership degree of $\mathbf{x}_i^*$ in each of the K classes $(\hat{\mu}_{C_i\in k}(\mathbf{x}_i^*))$ will be:

$$\hat{\mu}_{C_i\in k}(\mathbf{x}_i^*) = g\left[\mu_{A_1(1)}(\mathbf{x}_i^*), \ldots, \mu_{A_D(1)}(\mathbf{x}_i^*)\right] \tag{8}$$

$$\hat{\mu}_{C_i\in k}(\mathbf{x}_i^*) = g\left[\mu_{A_1(2)}(\mathbf{x}_i^*), \ldots, \mu_{A_D(2)}(\mathbf{x}_i^*)\right] \tag{9}$$

$$\vdots$$

$$\hat{\mu}_{C_i\in k}(\mathbf{x}_i^*) = g\left[\mu_{A_1(K)}(\mathbf{x}_i^*), \ldots, \mu_{A_D(K)}(\mathbf{x}_i^*)\right] \tag{10}$$

Based on the results obtained with AutoFIS [5], RandomFIS uses Weighted Average estimated through Restricted Least Squares (RLS) as the aggregation operator [11]:

$$\hat{\mu}_{C_i\in k}(\mathbf{x}_i^*) = \sum_{d^{(1)}=1}^{D^{(1)}} w_{d^{(1)}} \mu_{A_d(1)}(\mathbf{x}_i) \tag{11}$$

$$\vdots$$

$$\hat{\mu}_{C_i \in k}\left(\mathbf{x}_i^*\right) = \sum_{d^{(K)}=1}^{D^{(K)}} w_{d^{(K)}} \mu_{A_d(K)}\left(\mathbf{x}_i\right) \tag{12}$$

where $w_{d^{(k)}}$ is the weight, or the degree of influence, of $\mu_{A_d(k)}(\mathbf{x}_i)$ in the discrimination of patterns related to class k. The same process described in the *Association* stage is used for finding the weights.

2.6 Decision

The *decision* on the membership of pattern $\mathbf{x}_i^*$ to class k is computed by:

$$\hat{C}_i = arg_k max\left\{\hat{\mu}_{C_i \in 1}\left(\mathbf{x}_i^*\right), \ldots, \hat{\mu}_{C_i \in k}\left(\mathbf{x}_i^*\right), \ldots, \hat{\mu}_{C_i \in K}\left(\mathbf{x}_i^*\right)\right\} \tag{13}$$

where $\hat{C}_i$ is the predicted class: result of the k-th argument that maximizes Eq. (13). Therefore, this method associates pattern $\mathbf{x}_i^*$ to the most pertinent class according to the existing rule base. In case of a tie, $\mathbf{x}_i^*$ will be associated to the majority class.

3 Experiments

In order to evaluate RandomFIS, five databases were selected from the UCI machine learning repository [12]: KDD Cup 1999, Poker Hand, Covertype, Census-Income (KDD), Fatality Analysis Reporting System (FARS). Table 1 presents their main features in terms of number of attributes (J), number of instances (patterns) (n) and number of classes. All databases were evaluated using 10-fold *crossvalidation*.

In the comparison of results for the models presented in [13], all multiple classes datasets have been transformed into binary classification problems. The labels of the datasets in Table 1 indicate the classes used in the binary classification problem.

The parameters used to create each fuzzy classifier have been defined according to the best results obtained in [5]: number of membership for each attribute = 3; product t-norm; active negation; maximum size of premise = 2; ε_{Sup} = 0,075 and ε_{sim} = 0,95.

Parameters such as number of *Subsamples* (s), number of *Bootstraps* (r) and γ, were empirically determined after some preliminary tests, based on results presented in [9]. Best results were obtained for $\gamma = 0.7$ and $r = 100$. In order to compare the performance of RandomFIS to those of other similar models, three different

Table 1. Main features of benchmark databases

Datasets	J	n	#Patterns per class
Kddcup_DOS_vs_normal	41	4856151	(3883370; 972781)
Poker_0_vs_1	10	946799	(513702; 433097)
Covtype_2_vs_1	54	495141	(283301; 211840)
Census	41	141544	(133430; 8114)
Fars_Fatal_Inj_vs_No_Inj	29	62123	(42116; 20007)

Subsamples values were considered: $s = 8, 16, 32$, as in [13]. In that work, the Chi-MapReduce model was proposed to deal with high dimensionality problems.

Results are compared in terms of number of fuzzy rules generated and Classification Accuracy, computed as:

$$Accuracy = \frac{\sum_{i=1}^{n} \left| C_i - \hat{C}_i \right|}{n} \tag{14}$$

where C_i is the real class of pattern $\mathbf{x}_i^*$, and $\hat{C}_i$ is the predicted class. In Eq. (14), $\left| C_i - \hat{C}_i \right| = 0$ if $C_i = \hat{C}_i$; and 1 otherwise.

3.1 Results

Tables 2, 3 and 4 present the results in terms of Accuracy, average Number of Rules and Computational Time for configurations *RandomFIS_Ave* and *RandomFIS_Rules*, for 8, 16 and 32 *Subsamples*. In terms of Accuracy, Table 2 indicates that RandomFIS_Rules provides better results than RandomFIS_Ave. Additionally, the configuration with 32 partitions tends to produce better Accuracy. Regarding the average number of rules generated (Table 3), RandomFIS_Rules outperforms RandomFIS_Ave. The former provides a significant reduction in the final number of rules, considerably enhancing interpretability while maintaining classification accuracy. The average computational time presented in Table 4, for each partition configuration, refers to the time needed for

Table 2. Average accuracy results for RandomFIS versions with 8, 16, 32 partitions.

Datasets	RandomFIS_Ave			RandomFIS_Rules		
	8	16	32	8	16	32
Kddcup_DOS_vs_normal	99.61	99.71	99.75	99.91	99.9	**99.92**
Poker_0_vs_1	57.07	56.77	56.98	58.46	59.27	**60.02**
Covtype_2_vs_1	73.15	73.23	73.18	76.58	76.54	**76.66**
Census	94.29	94.31	94.29	94.53	94.56	**94.58**
Fars_Fatal_Inj_vs_No_Inj	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
Average	84.05	84.04	84.08	85.90	86.05	**86.24**

Table 3. Average Number of Rules results for RandomFIS versions with 8, 16, 32 partitions.

Datasets	RandomFIS_Ave			RandomFIS_Rules		
	8	16	32	8	16	32
Kddcup_DOS_vs_normal	**103.80**	192.50	402.70	**43.8**	50.1	70
Poker_0_vs_1	**623.00**	1271.10	2522.90	111.2	104.2	**96**
Covtype_2_vs_1	**97.00**	186.70	384.60	**31.5**	35.4	44.7
Census	**252.70**	505.20	1012.90	**54.6**	66.7	76.6
Fars_Fatal_Inj_vs_No_Inj	**360.90**	734.40	1447.70	**194.3**	328.5	525.8
Average	**287.48**	577.98	1154.16	**87.08**	116.98	162.62

Table 4. Average runtime elapsed for RandomFIS versions with 8, 16, 32 partitions.

Datasets	Partitions		
	8	16	32
Kddcup_DOS_vs_normal	**126777.77**	250116.3	442169.25
Poker_0_vs_1	**84356.87**	167628.51	330257.05
Covtype_2_vs_1	**3284.65**	6013.94	11536.96
Census	**7148.62**	14666.95	29682.78
Fars_Fatal_Inj_vs_No_Inj	**10865.68**	22299.52	44596.19
Average	**46486.72**	92145.04	171648.45

processing both RandomFIS models in an Intel Core i7-5820 K CPU @ 3.30 GHz, RAM 32 GB, Windows 7 × 64 PC.

The best RandomFIS model – RandomFIS_Rules – was compared to two similar models, proposed in [13]: Chi-FRBCS and Chi-MapReduce. Figures 4 and 5 show the results in terms of Classification Accuracy and Average Number of Rules.

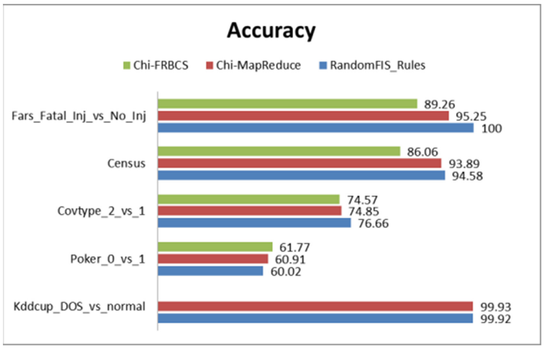


Fig. 4. Comparison of classification performance for RandomFIS_Rules

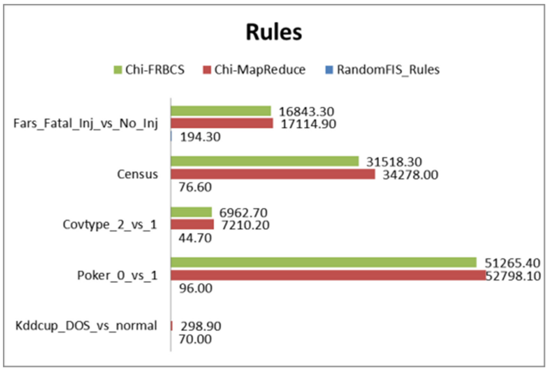


Fig. 5. Comparison of average number of rules for RandomFIS_Rules

As can be seen from Fig. 4, RandomFIS_Rules provides better Accuracy in most databases. Results for Poker_0_vs_1 and Kddcup_DOS_vs_normal are inferior to those presented in [13]. However, Fig. 5 shows that RandomFIS is significantly superior to Chi-FRBCS and Chi-MapReduce regarding the Average Number of Rules.

4 Conclusions

This paper presented a new classification model, called RandomFIS, that is able to deal with high dimensionality problems. The proposed model was evaluated for five different binary databases and results demonstrate that it provides good accuracy with a reduced number of fuzzy rules. This enhances its interpretability when compared to other similar models in the literature. Future work will involve the application of RandomFIS to benchmarks with multiple classes, as well as to real big datasets. Additionally, feature selection methods shall be added to improve its performance, as well as some other ways of combining the generated fuzzy classifiers.

References

1. Kuncheva, L.I.: Fuzzy Classifier Design, vol. 49. Springer, Heidelberg (2000)
2. Zhang, Y., Wu, X.B., Xing, Z.Y., Hu, W.L.: On generating interpretable and precise fuzzy systems based on Pareto multi-objective cooperative co-evolutionary algorithm. Appl. Soft Comput. J. **11**(1), 1284–1294 (2011)
3. Koshiyama, A.S., Vellasco, M.M.B.R., Tanscheit, R.: GPFIS-CLASS: A Genetic Fuzzy System based on Genetic Programming for classification problems. Appl. Soft Comput. **37**, 561–571 (2015)
4. Lemos, A., Caminhas, W., Gomide, F.: Evolving Intelligent Systems: Methods, Algorithms and Applications, pp. 117–159. Springer, Berlin, Heidelberg (2013)
5. Paredes, J., Tanscheit, R., Vellasco, M., Koshiyama, A.: Automatic synthesis of fuzzy inference systems for classification. In: Carvalho, J.P., Lesot, M.-J., Kaymak, U., Vieira, S., Bouchon-Meunier, B., Yager, R.R. (eds.) IPMU 2016. CCIS, vol. 610, pp. 486–497. Springer, Heidelberg (2016). doi:10.1007/978-3-319-40596-4_41
6. Ishibuchi, H., Nakashima, T., Nii, M.: Classification and modeling with linguistic information granules: advanced approaches to linguistic Data Mining (2006)
7. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms (2004)
8. Panov, P., Džeroski, S.: Combining bagging and random subspaces to create better ensembles. In: International Conference on Intelligent Data Analysis, pp. 118–129 (2007)
9. Kleiner, A., Jordan, M.I.: The big data bootstrap. In: Proceedings of 29th International Conference Machine Learning, p. 8 (2012)
10. Fernández, A., Calderón, M., Barrenechea, E., Bustince, H., Herrera, F.: Solving multi-class problems with linguistic fuzzy rule based classification systems based on pairwise learning and preference relations. Fuzzy Sets Syst. **161**(23), 3064–3080 (2010)

11. Calvo, T., Kolesárová, A., Komorníková, M., Mesiar, R.: Aggregation operators: properties, classes and construction methods. Aggreg. Oper. New Trends Appl. **97**(1), 3–104 (2002)
12. Lichman, M.: UCI Machine Learning Repository (2013). http://archive.ics.uci.edu/ml. Accessed 01 Mar 2016
13. del Río, S., López, V., Benítez, J.M., Herrera, F.: A mapreduce approach to address big data classification problems based on the fusion of linguistic fuzzy rules. Int. J. Comput. Intell. Syst. **8**(3), 422–437 (2015)

Big Data for a Linked Open Economy

M. Vafopoulos[1(✉)], I. Anagnostopoulos[2(✉)], D. Negkas[3], G. Razis[2], G. Vafeiadis[1], I. Skaros[3], K. Glykos[3], A. Tzani[3], and E. Galanos[1]

[1] Software and Knowledge Engineering Laboratory, IIT, NCSR-"Demokritos", Athens, Greece
{mvafopoulos,gvaf,egalanos}@iit.demokritos.gr

[2] Division Software and Knowledge Engineering Laboratory, Organization IIT, NCSR-"Demokritos", Athens, Greece
{janag,razis}@dib.uth.gr

[3] University of Piraeus Research Centre, Piraeus, Greece
linkedeconomy@gmail.com

Abstract. As the volume data grows exponentially, more and more big data handling approaches are also applied in the linked data cloud. Thus, semantic triplets which are nucleus of the Resource Description Framework (RDF) must be harmonized to the demanding needs of the 4 Vs. This paper presents the architecture and the main components of a big linked data repository named LinkedEconomy. The scope of the platform is to collect, process, interlink and publish unprecedented high-detailed economic data in machine-readable format, in order to (a) provide a new data corpus for enriching research efforts in Economic, Statistics and Business studies, and (b) contribute to Big data analytics for corporate decision making.

Keywords: Big data · Linked data · Open data · Semantic web · Ontology · Economy · Public finance · Prices

1 Introduction – Related Works

The present paper describes the underlying structure and functionality of LinkedEconomy[1], which stands as a co-funded project under the National Strategic Reference Framework (NSRF). LinkedEconomy could be considered as a basic component of Web economy providing a universal access to Greek and international economy data, as well as at promoting the benefits of linking heterogeneous sources under the concept of big, open and reusable data in this critical domain [1]. These data flows are of high added value not only for exploration purposes, but also for exploiting the benefits of transparency and openness to the citizens, the research and business communities, and the government itself.

Related research approaches are characterized by a two-fold approach. In the first fold, there are efforts that analyse the benefits of using open data in economy, trying in parallel to provide a unified framework and modelling strategy [2, 3] and [4]. In the

[1] http://linkedeconomy.org/en/.

second fold, can be found initiatives that integrate the benefits of openness, thus creating a flourishing linked ecosystem of economy data [5–7] and [8].

Through our developed platform, we aim at tackling the challenge of building a common terminology for the basic financial and economy activities, which will -in turn- facilitate the research over new linked data and sources. All described components that are analyzed in the following sections, form a system capable of linking economy-related data at large scale, creating in parallel a framework for collect, validate, clean and publish linked big data streams. According to our knowledge, linkedeconomy.org is quite innovative effort and consist some of the cutting edge semantic and big data handling approaches.

2 Process Model and Infrastructure for Handling Big Data

Open economic data related to public budgeting, spending and prices are characterized by high volume, velocity, variety and veracity. Thus, we have decided to build custom components under the common logic of transforming static data to linked and big open data streams. This decision was made for the benefit of flexibility that helps us to address the 3 V's of economic data and cannot be found at Linked (Open) Data Management Suites (e.g. UnifiedViews[2]).

The big picture of our approach is depicted in Fig. 1. As shown, data are stored in raw (as harvested from sources), in RDF and JSON formats. Enriched data are distributed though five channels: data dumps (CKAN open source data portal software[3]), SPARQL queries, Web, social media and structured inputs to Business Intelligence (BI) systems. Additionally, data can be further analysed and exchanged with relevant platforms (e.g. SPARQL to R). The validation component runs through out the whole process to safeguard high data quality by detecting errors. The messaging component works as an internal messaging and alert system for all components. It also produces and automatically posts messages to Twitter related to data availability (e.g. Daily #opendata for public financial decisions (2/16) @diavgeia @YDMED @CKANproject @OpenGovGr @ODIAthens #diavgeia http://goo.gl/oGmfDP).

As core infrastructure we use okeanos[4], which is an established cloud-based service provided for the Greek research and academic community. Moreover, our computational stack consists of 12 virtual machines with memory and storage capacities that span from 4 GB to 16 GB RAM and 100 GB to 500 GB respectively, as well as a non-commodity (physical) server of 12 CPUs, 128 GB RAM and a storage capacity of more than 8 TB. The first column of Table 1 indicates the nodes of our infrastructure; the second column highlights their distinct role and functionality provided (in both internal and external procedures), while the last column depicts the third-party dependencies and services employed, as well as the stored data sources.

[2] https://www.semantic-web.at/unifiedviews.

[3] http://ckan.org/.

[4] https://okeanos.grnet.gr/home/.

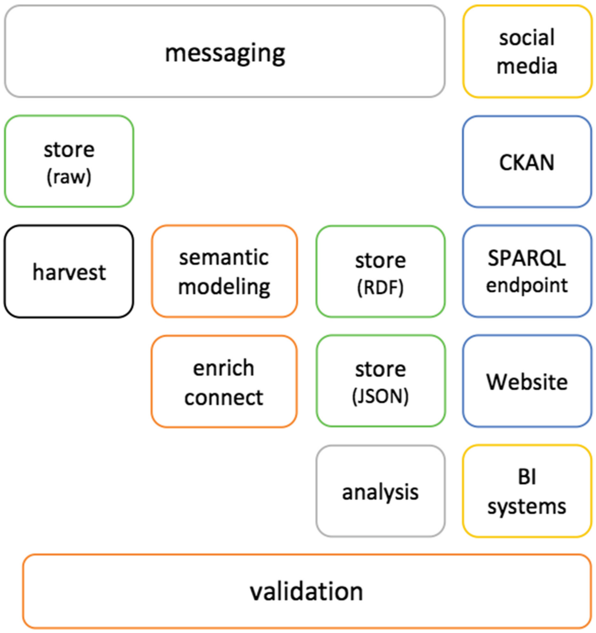


Fig. 1. LinkedEconomy process layers

Table 1. LinkedEconomy general infrastructure

	Functionalities/Components	Services/Data sources
VM1	Linkedeconomy.org	Apache, PHP, MySQL, Drupal
VM2	SPARQL endpoint, demo site	OLV, Apache, PHP, MySQL, Drupal
VM3	Harvester	CouchDB, Lucene, Apache, MySQL/ CKAN (Greek Datasets)
VM4	Harvester, Messenger	MySQL, LinkedEconomy Dropbox
VM5	Storage - Secondary triplestore	CouchDB, OLV, CouchDB-Lucene, Docker, 4store
VM6	Harvester	Apache, PHP, MySQL, Drupal/ CKAN (Foreign Datasets)
VM7	SPARQL endpoint	OLV (Foreign graphs), 4store
VM8	Management	JIRA, MySQL, Tomcat
VM9	Dashboard	front-end, CMS, INSPINIA
VM10	System administration	VPN, firewalls
VM11	Core Triplestore	4store
VM12	Core Triplestore	4store
Physical	Storage - Core Triplestore	OLV (Greek graphs)

As large triplestore infrastructure, we use Open Link Virtuoso for 15 different sources of nearly 1B triples, while during the last couple of months we support the pilot inclusion of nearly ten extra data sources per month by employing parallel processing nodes of the Garlik 4store[5] under the GPLv3 license.

In the context of the Big Data Europe project (funded by EU to develop the main Big Data infrastructure for European organisations[6]) we are testing a more robust software for handling real time Big data streams and producing analytics.

Figure 2 illustrates the processes involved when handling in real time loads from three different sources (A: Diavgeia, B: E-Procurement and C: National Strategic Reference Framework or in short NSRF) in order to deal with variety, veracity, velocity and volume with million triples. As data acquisition agents against the respective sources, we employ Apache Flume[7], which is a distributed open service for efficiently collecting, aggregating, and moving large amounts of source log data in cooperation with Apache Kafka[8]. As a result, data streams are partitioned and spread over a dedicated cluster of our VMs, thus supporting larger data streams. Unified views of aggregated results are directly handled by Spark[9] through our OLV triplestore, while the user receives them as a easy-to-consume web page.

In LinkedEconomy.org, as *unified view* we define the result of a rich-content web page, which is enhanced with semantified aggregated information derived from

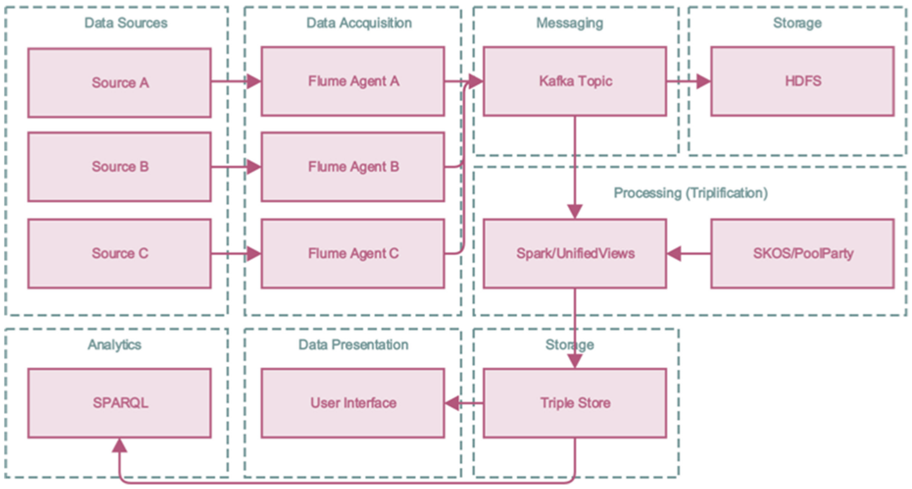


Fig. 2. Handling Big Data in LinkedEconomy infrastructure

5 https://github.com/garlik/4store.

6 https://www.big-data-europe.eu/.

7 https://flume.apache.org/.

8 http://kafka.apache.org/.

9 http://spark.apache.org/.

multiple sources and according to the processes of Fig. 2. We next present an instance of a unified view for organizations (see Fig. 3, case: Greek Ministry of Culture, http://linkedeconomy.org/en/page/paymentAgents?=afm=090283815). It consists of the tabs DIAVGEIA, E-Procurement (KHMDHS) and NSRF. Each tab presents related data concerning the organization according to each source A, B, and C described above. In this way, the user can have the "big picture" of the provided information in a single web point.

More analytically, the first tab shows aggregated information of a buyer and/or a seller in respect to payments as it is published in DIAVGEIA. The user can see the basic information of a buyer/seller (e.g. name, vat id, address), as well as payments and procurements from 2010 up to the current year. Also, the user is informed with detailed data on payments (such as commitments, approvals and finalized decisions) and procurements (e.g. assignments, notices and awards). The second tab shows information about a buyer or seller, as published in the Central Electronic Public Procurement Register (e-Procurement). Similarly to the above tab, the user can see the basic information about a buyer or a seller, as well as the respective tenders, contracts and payments from 2013 to this year. For these issues more details are also provided, such as the date, the name of the seller (or buyer), the amount, the common procurement vocabulary and source. The third tab displays information about the NSRF (National Strategic Reference Framework) subsidies received by a beneficiary. The user receives the basic information of each beneficiary (e.g. name, vat id, address), and the information about the respective subsidies, such as budget, contracts and payments. Finally, the user also receives detailed information on subsidies, such as the budget, the related contracts and payments.

In a similar way, LinkedEconomy.org provides unified profiles for other cases, such as the statistics based on the Common Procurement Vocabulary (CPV)[10] and on Diavgeia[11,12], as well as profile pages based on single data sources. Some examples are the profiles and the statistics of the Central Market of Thessaloniki (KATH)[13,14], as well as profiles and statistics from products[15,16] or shop/market points[17,18].

[10] http://linkedeconomy.org/en/page/cpv?=cpv=09000000-3.

[11] http://linkedeconomy.org/en/diaugeia/stats-diaugeia.

[12] http://linkedeconomy.org/en/diaugeia/errorsvat-diaugeia.

[13] http://linkedeconomy.org/en/kathprices?=product=16.

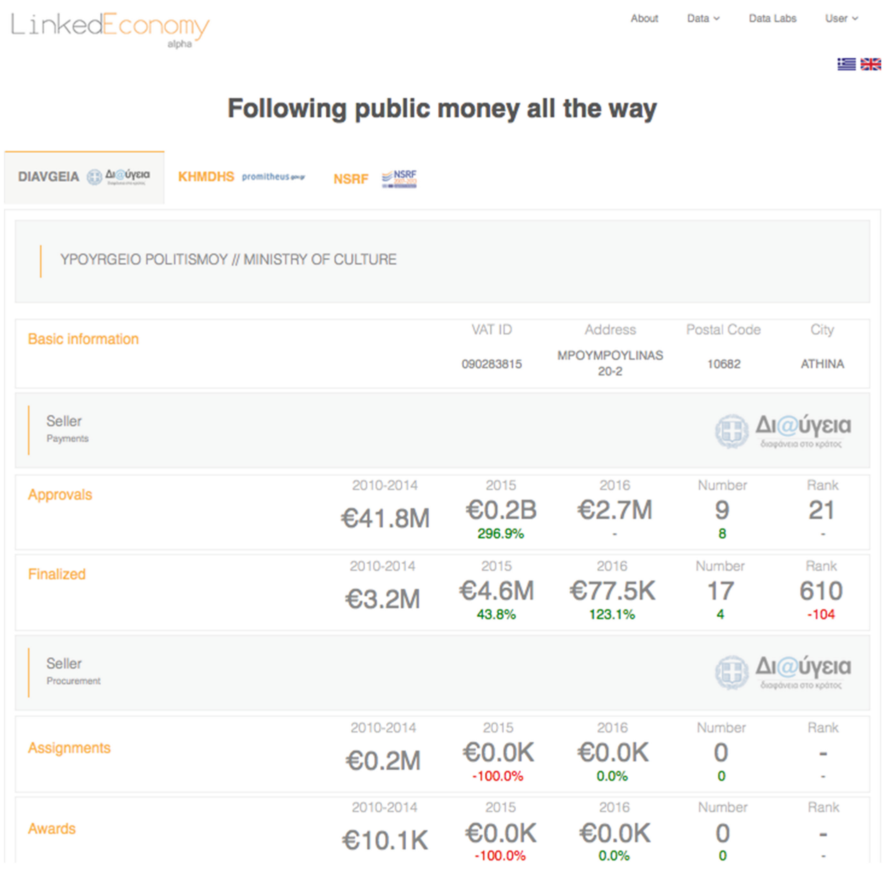
[14] http://linkedeconomy.org/en/kath-stats.

[15] http://linkedeconomy.org/en/page/eprices-product?=id=107.

[16] http://linkedeconomy.org/en/page/eprices-product-stats.

[17] http://linkedeconomy.org/en/page/eprices-shop?=id=1090.

[18] http://linkedeconomy.org/en/page/shops-stats.



Following public money all the way

YPOYRGEIO POLITISMOY // MINISTRY OF CULTURE

Basic information	VAT ID	Address	Postal Code	City
	090283815	MPOYMPOYLINAS 20-2	10682	ATHINA

Seller
Payments

	2010-2014	2015	2016	Number	Rank
Approvals	€41.8M	€0.2B 296.9%	€2.7M -	9 8	21 -
Finalized	€3.2M	€4.6M 43.8%	€77.5K 123.1%	17 4	610 -104

Seller
Procurement

	2010-2014	2015	2016	Number	Rank
Assignments	€0.2M	€0.0K -100.0%	€0.0K 0.0%	0 0	- -
Awards	€10.1K	€0.0K -100.0%	€0.0K 0.0%	0 0	- -

Fig. 3. The result of a unified view instance - the Greek Ministry of Culture

3 Basic Components for Dealing with Big Open Economic Data

In this section, we present the basic components and processes that mutually interoperate in order to support LinkedEconomy.org as an ecosystem of big data, web technologies and semantics in the economy domain. All kinds of our data are textual based information that is related with the economy field, supporting both static and dynamic streaming threads, while the adaptation of the 3Vs in our paradigm can be summarized as:

Variety refers to the different types and nature of data we can now use. This helps people who analyze it to effectively use the resulting insight. In the past, we focused on structured data that neatly fits into tables or relational databases such as financial data (for example, sales by product or region). In fact, 80 percent of the world's data is now unstructured and therefore cannot easily be put into tables or relational databases.

Veracity refers to the messiness or trustworthiness of the data in terms that the quality of captured data can vary greatly, affecting accurate analysis. Big data and analytics technology now allows us to work with these types of data. The volumes often make up for the lack of quality or accuracy.

Volume refers to the vast amount of data generated and stored in a small time unit, in a sense that no traditional database can efficiently store it. With big data technology we can now store and use these data sets with the help of distributed systems, where parts of the data is stored in different locations, connected by networks and brought together by software when needed.

3.1 Variety: Harvest Data from Several Economic Sources in Different Formats

In LinkedEconomy.org the Harvester Component is a software module that gathers online information in an automated way, storing it in a machine-readable format (preferable as tabular data).

Harvester function is two-fold. It either fetches data from publicly available web sites that use APIs or from web sites that do not offer data via an API. Thus, in the later case, the harvesting procedure differs from source to source, due to the different ways of data storage and data provision in each source. Moreover, there are sites that store their data on excel files formats, others where data are presented in HTML tables, while in other cases data are presented in a tabular format where filtering and shorting is allowed, usually through the Flash technology. The most trivial case is when a data source shares a link for data provision. Yet, depending on our needs and the way we want to use the data, we usually transform the data from one form to another (e.g. from excel to Comma Separated Value/CSV format) by employing some third party libraries (e.g. JXLS[19]), and then we save the fetched data to a plain-text CSV file. A simply harvesting case is when we read tabular data in HTML format.

The use of dedicated libraries facilitates the procedure since data in table rows can be stored as programming arrays for advanced processing. The most complicated procedure for the Harvester Component is the one that needs some kind of manual actions from our part when fetching the data from its source. An example for this case is the sources that provide budget information for the city of Thessaloniki[20] and the city of Kalamaria[21]. In both of cases, the information is served through interactive functions (radio buttons, dropdown menus), thus making necessary manual adjustments prior to any automating harvesting procedure. For tackling the issue we use advanced programming techniques (e.g. the UI4 J[22] Java library) that gives us the ability to automate all manual actions.

[19] http://jxls.sourceforge.net/.

[20] http://www.thessaloniki.gr/egov/budget.html.

[21] http://opendata.kalamaria.gr:8080/accounting/opendata/budgetView.

[22] https://github.com/ui4j/ui4j.

3.2 Veracity: Normalize, Validate and Map the Data

The phase following the data acquisition is the modeling of a data source or, in other words, the generic conceptualization (abstract definition) of its information. The modeling is performed according to the semantification component where we apply the domain experts' knowledge, in order to convert the input data of the data sources into a semantically enriched knowledge graph. The input of the semantification component varies according to the source type and the way the data are provided to the end-users, such as through APIs, open tabular formats (e.g. CSV) or even through specific source-oriented harvesting procedures. Often, the data need validation and therefore external services and third-party dependencies are employed. Figure 4 illustrates the role of the Semantification Component and the related processes in all data life-cycle from their modeling up to their provision in LinkedEconomy.

For the semantification component, the core programming technology is Java and more specific the Apache Jena[23], which is a free and open source Java framework for building Semantic Web and Linked Data applications. This framework provides all the necessary methods for creating resources, literals and properties, in order to model the input data and to generate RDF graphs. These graphs are then stored in an OpenLink Virtuoso (OLV) that serves as a semantic triplestore. Then through SPARQL queries users can satisfy their information needs. Finally, the results (in many cases combined from multiple linked data sources) are transformed in JSON format and then are stored in NoSQL back-end infrastructures (CouchDB) for supporting efficiently the front-end data provision in our platform.

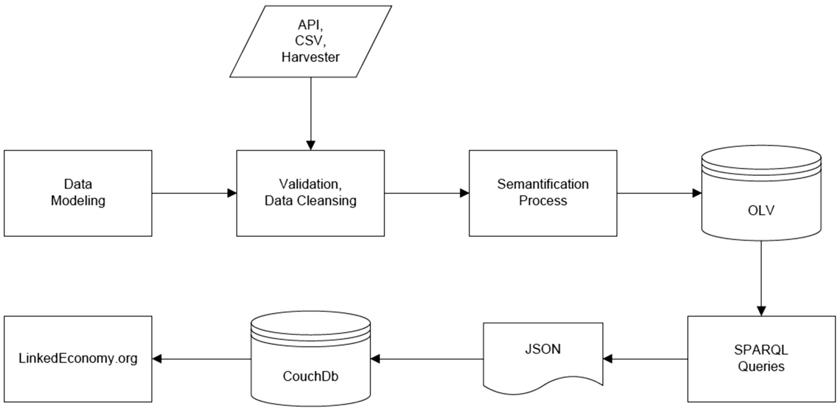


Fig. 4. The semantification process in LinkedEconomy data life-cycle

[23] https://jena.apache.org/index.html.

3.3 Volume: Handle the Data: Storage, Indexing and Publish the Data

LinkedEconomy data are stored in a CouchDB[24] instance in document format. CouchDB is a consistent document-oriented DB that stores data in JSON format with a RESTful HTTP API for executing create, read, update and delete (CRUD) commands. In our case, we maintain more than 50 folders containing up to 300.000 documents. The updates occur weekly or monthly by using the Bulk Document API[25].

For full-text search we use CouchDB-Lucene[26], which integrates CouchDB and Lucene text-based search engine. The related CouchDB document fields are indexed, while the queries are submitted to the created Inverted Indices[27], thus providing fast responses.

Finally, in LinkedEconomy, we use CKAN data portal to store and serve the data after the harvesting procedures. In order to fully automate the procedure, the CKAN component uploads periodically the data (in zip format). For each dataset that needs to be handled and uploaded to CKAN, the user (member of our team) creates two distinct files with all necessary manual settings. The first one contains the parameters that are needed for choosing the right data and creating the compressed file, while the second contains information that is needed for basic CKAN functionalities.

4 Discussion - Future Work

This paper presents the basic components and process flows of LinkedEconomy, which lead to publishing data of high added value not only for exploration purposes, but also for exploiting the benefits of transparency and openness to the citizens, the research community, and the government itself. Our efforts produced a CKAN repository, which publishes datasets from sources that are being updated regularly and contain valuable information in respect to social and economic research[28]. Citizens and economy stakeholders (government, local authorities) can exploit 27 datasets from 14 classified data sources in many different formats (XLSX, CSV, RDF). Examples include economy data, such as public procurements, budgets, prices, expenditures, taxes and fines.

Finally, users of LinkedEconomy receive high-level aggregated information through the traditional Web 2.0 paradigm (web pages with rich-content, social media, online community) with the help of advanced search capabilities and intelligent Structure Data Markup provision for the returned results. However, apart from traditional client-server model techniques, our platform: (a) employs big data handling procedures in terms of variety, veracity and high volume data, (b) envisage the use of Web 3.0 technologies by offering a publicly available SPARQL endpoint[29] (offering machine-readable results from a fast growing semantic repository of more than 1B

[24] http://couchdb.apache.org/.

[25] https://wiki.apache.org/couchdb/HTTP_Bulk_Document_API.

[26] https://github.com/rnewson/couchdb-lucene.

[27] https://lucene.apache.org/core/3_6_0/api/all/org/apache/lucene/search/Similarity.html.

[28] http://ckan.linkedeconomy.org.

[29] http://linkedeconomy.org/sparql.

triplets in total), and (c) supports the open community by sharing all ontological schemas and related specifications in GitHub[30].

Having the necessary expertise and based in the same processes and components, we currently work on modeling a long series of global economic datasets. We expect to support really soon the provision of more than 15 different economic datasets from Europe, UK, Australia, USA and Canada, while we plan to further extend this economy linked data cloud in the near future.

References

1. Vafopoulos, M.: The Web economy: goods, users, models and policies. Found. Trends Web Sci. **3**(1–2), 1–136 (2011)
2. O'Riain, S., Curry, E., Harth, A.: XBRL and open data for global financial ecosystems: a linked data approach. Intl. J. Account. Inf. Syst. **13**(2), 141–162 (2012)
3. Hodess, R.: Open budgets: the political economy of transparency, participation, and accountability. J. Econ. Lit. **52**(2), 545–548 (2014)
4. Tygel, A.F, Attard, J., Orlandi F., Campos, M.L., Auer, S.: "How much?" Is Not Enough-An Analysis of Open Budget Initiatives, arXiv:1504.01563 (2015)
5. Petrou, I., Meimaris, M., Papastefanatos, G.: Towards a methodology for publishing Linked Open Statistical Data. JeDEM-eJournal eDemocracy Open Gov. **6**(1), 97–105 (2014)
6. Höffner, K., Martin, M., Lehmann, J.: LinkedSpending: openspending becomes linked open data. Semantic Web **7**(1), 95–104 (2015)
7. Alvarez-Rodríguez, J.M., Vafopoulos, M., Llorens, J.: Enabling policy making processes by unifying and reconciling corporate names in public procurement data. The CORFU Technique, Comput. Stan. Interfaces **41**, 28–38 (2015)
8. Vafopoulos, M.N., Vafeiadis, G., Razis, G., Anagnostopoulos, I., Negkas. D., Galanos, L.: Linked Open Economy: Take Full Advantage of Economic Data, SSRN 2732218 (2016)

[30] https://github.com/LinkedEcon/LinkedEconomyOntology-ELOD.

Smart Data Integration by Goal Driven Ontology Learning

Jingliang Chen[1,2], Dmytro Dosyn[3], Vasyl Lytvyn[3], and Anatoliy Sachenko[4,5](✉)

[1] State Key Lab of Software Engineering, Computer School, Wuhan University, Wuhan, China
chenjl520@163.com

[2] Computer School, Hubei University of Technology, Wuhan, China

[3] Lviv National Polytechnic University, 12 Bandery Street, Lviv 79013, Ukraine
{dosyn, lytvyn}@ukr.net

[4] Silesian University of Technology, ul. Roosevelta 26-28, 41-800 Zabrze, Poland
as@tneu.edu.ua

[5] Ternopil National Economic University, 3 Peremogy Square, Ternopil 46020, Ukraine

Abstract. The smart data integration approach is proposed to compose data and knowledge of the different nature, origin, formats and standards. This approach is based on the selective goal driven ontology learning. The automated planning paradigm in a combination with a value of the perfect information approach is proposed to be used for evaluating the knowledge correspondence with the learning goal for the data integration domain. The information model of a document is represented as a supplement to the Partially Observable Markov Decision Process (POMDP) strategy of a domain. It helps to estimate the document a pertinence as the increment of the strategy expected utility. A statistical method for identifying the semantic relations in the natural language texts for their linguistic characteristics is developed. It helps to extract the Ontology Web Language (OWL) predicates from the natural language text using data about sub semantic links. A set of methods and means based on ontology learning was developed to support the smart data integration process. A technology uses the Natural Language Processing software Link Grammar Parser, WordNet Application Programming Interface (API) as well as the OWL API.

1 Introduction

Authors of the EU Horizon-2020 program confess that there is an underdeveloped data sharing culture. An industry needs a systematic transfer of knowledge and technology across different sectors. Using data across sectors for offering new services opens the new opportunities to meet the business and societal challenges. A lack of agreed standards and formats, and the low rates of publishing data assets in machine discoverable formats further hold back the data integration. A fact that textual data appear in many languages creates an additional challenge for sharing and linking such data.

The issue of the integration data assets, knowledge and technologies is tremendous, and it has many solutions and consists of many significant sub-problems. The most promising solution maybe should be connected with both an agent approach and a semantic web, because big amounts of data interconnect as a whole in different aspects and for different purposes only by means of logically defined semantic relations. Their implementation ultimately depends on ontology of a particular domain – its content, volume, structure, adaptivity and learning method. We are considering the ontology as the formal explicit representation of the common terminology and its logical interdependence for a certain subject domain. The ontology formalizes the intensional meaning of the domain - e.g. set of rules in terms of formal logic, while its extensional meaning is defined in the knowledge base as a set of facts about instances of concepts and relationships between them.

The process of filling the knowledge base is called knowledge markup, or ontology population. In return the ontology learning (OL) methods are methods for automatic (semi-automatic) ontology structure development, based on natural language processing (NLP) and machine learning. Far less attention is paid to the approaches developed in the field of automated planning (AP) in particular to the hierarchical task network (HTN) approach.

The bottleneck of the smart data integration approach lays in an effective OL technology. A manual OL is too expensive and time-consuming. On the other side, the ontology structure should fit needs of domain tasks. The methods of simple collecting of facts, implemented by most ontology population tools, are not effective enough. The absence of the logical structure causes a lack of integrating the collected data. They are classified only, and it's not enough for decision making in the domain problem solving. A key issue of smart data integration therefore is to build the OL strategy that could take into account the particular domain tasks, their solving methods and tools. It means the ontology structure should include (during an OL process) all related terms and semantic relations. For this purpose the OL method must recognize the measure of "usefulness" of logical (semantic) structures of data, detected by applied NLP method – knowledge pertinence.

We suppose the AP approach could provide such infrastructure to estimate the knowledge pertinence. Therefore including AP into ontology structure, based on the planning paradigm, should solve the problem of the selective OL with the ultimate aim of smart data integration.

2 Relative Works

A very first approach of data integration in a field of text mining using NLP was based on the processing text documents (classification, ranking by relevance) [1]. There were used not the systematic metadata about the document but the whole body of a unstructured text document and it was presented as a set of words or concepts (i.e. "bag of words") [1, 2].

The first systematic researches of the ontology automatic construction (ontology learning) considered the three main aspects of the problem (1) methodology; (2) assessment; (3) using the scenarios in the particular subject area. A modern information retrieving paradigm was implemented at the existing digital indexing, retrieval, and navigation resources: PubMed [3], Web of Science [4], CiteSeer, Scopus, Google Scholar and Textpresso [5].

Formalization techniques of actions and measures, that person can perform, were proposed by applying the methodology of automatic planning, or more precisely, the approach based on the POMDP model [6].

Artificial Intelligent (AI) planning techniques were employed to automate the composition of web services using the semantic web ontology [7], it was proposed to combine services using the description logic reasoning and OWL descriptions of their preconditions and effects.

In [8] the domain-independent knowledge-engineering and planning framework is described to integrate the planning process with the description logic (DL) reasoning. A state of the world is presented as a set of OWL facts using the Resource Description Framework (RDF) graph. Actions are described as a RDF graph transformations, planning goals – as RDF graph patterns.

More general approach was proposed in [9] using the ontology to represent planning tasks and developing the algorithms to convert specifications from OWL to HTN, and vice-versa. To operate by ontology and write the transformed planning tasks as a JSHOP file the OWL API was used [10]. This approach in combination with the previous one should serve as the evidence of a possibility to build the domain ontology as the general domain HTN, useful to estimate the new knowledge pertinence.

3 Goal Driven Ontology Learning (GDOL)

The key difference between an ordinary and a goal driven ontology learning consists in the selective updating ontology only by data with the high enough pertinence.

A problem facing the developers of GDOL methods and tools is extremely complex because it includes the following sub-problems:

- Natural Language (NL) text linguistic analysis;
- building the message model, identified in text document, using a formal knowledge representation language;
- constructing the ontology and knowledge base of the intelligent agent as a model of information needs (interests) for a client of the information retrieval system;
- automatic constructing the optimal strategy of the intelligent agent;
- numerical evaluating the pertinence of the message, detected in text document, for the client of the information retrieval system;
- numerical evaluating the reliability of the received message.

Only a combination of the above tools allows us to start solving the problem facing the developers of GDOL tools.

The interdisciplinary nature of the GDOL makes us to consider the process of the knowledge extraction from NL texts as the technology engineering in the field of knowledge engineering similar to the design of vehicles, or, when attempting to build a prototype of an artificial intelligence system - similar to the design of an aircraft. This similarity is due to the need to use different technology nodes (i.e., different IT) as a subsystems of an entirely integrated mechanism for the discovery of new knowledge, its storing and ordering to identify and solve the most actual problems with the aim of max effectiveness. In this regard, it is difficult to rely on existence of the problem universal solution for automatic construction of ontology or, respectively, the united metric for assessing the pertinence of the new knowledge. Instead, at this point it is important, firstly, to identify principles based on which it is possible to create a system of recognition of meaning of NL text and search for new knowledge in it, and secondly, to determine the performance criteria of such system, which in turn will provide an opportunity to seek ways to improve its structure and functions with the aim to achieve the certain optimal performance.

3.1 NL Text Linguistic Analysis

The case study was produced using the information retrieving system CROCUS, designed on the basis of Protege-OWL API with help of Carnegie Mellon Link Grammar Parser (further – LGP) [11], Word Net [12] and a few other open sources of software libraries. In the linguistic subsystem of CROCUS the semantic relation recognition process is running using the Bayesian method for recognition of signs set, stored in patterns ontology of known semantic relations.

Learning *d* signs of *j*-th semantic relation is running according to a formula:

$$p(X|C_j) \propto \prod_{k=1}^{d} p(X_k|C_j). \tag{1}$$

To recognize the *j*-th semantic relation by the *d* revealed signs we use:

$$p(C_j|X) \propto p(C_j) \prod_{k=1}^{d} p(X_k|C_j). \tag{2}$$

The results of parsing the NL sentence using LGP parser onto pairs of words linked by meta-semantic links were used as the descriptors (signs) of certain semantic links with object and subject as the parameters.

As an example, for the simple test sentence:

```
[(a)(test.n)(is.v)(an)(example.n)].
```

LGP parsing result:

```
[[0 1 0 (Ds)][1 2 0 (Ss)][2 4 0 (Ost)][3 4 0 (Ds)]].
```

The results of Bayesian recognizing the semantic relation using (2) after the short learning:

(1) cause: 1.0882684165532656E-4;
(2) caused-by: 0.013810506200916856;
(3) **is-a: 0.024124901979118252;**
(4) is-about: 0.0;

(5) part-of: 0.0022765542079946285;
(6) same-as: 0.0;
(7) similar-to: 1.0261341731138478E-6.

The same parsing data were used to recognize the whole triplet with its subject and object. A set of such semantic triplets forms the RDF_T-form axioms for ontology of the message author, or directly the RDF_A graph of the message as it mentioned in [8].

3.2 Building a Formal Model of the Message

A message has usually the two main parts: a stating part as the preamble and a constructive part with an essence. A first part would help an addressee of the message to find out the adequate context to that message in author's ontology. A second part should be used to update information in that context (Fig. 1).

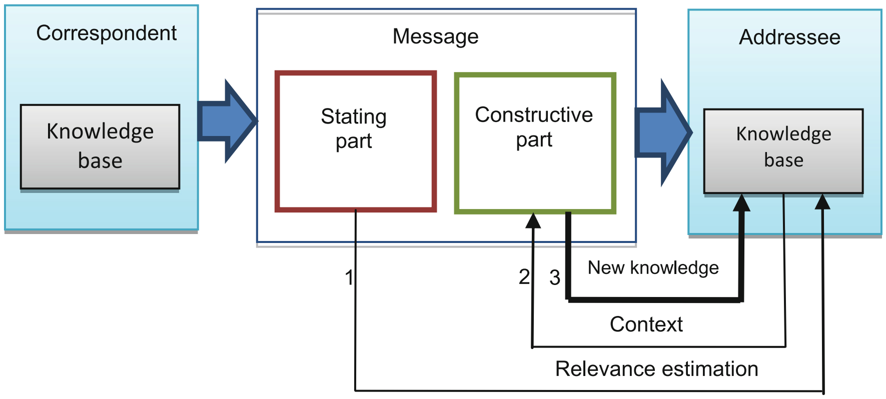


Fig. 1. Message model consisting of two main components: stating and constructive part.

A formal OWL-DL model of the message, contained in the processed NL text document, is the RDF_A graph. The process of its building includes the following stages:

- text preprocessing, where all pronouns are substituted by correspondent nouns (noun groups);
- dividing the complex sentences onto simple with connection them by basic logical dependencies;
- creating the model of ontology for author of the NL text document on the basis of simplified own ontology, completed by RDF_T data from the document;
- building an initial model of the message as a RDF graph of semantic triplets, connected between each other by logical links, defined in previous stages;
- supplementing the created RDF model with concepts and relations from model ontology of the message author with the aim to understand (i.e. formulate) which problem is solved;
- eliminating the logical redundancy and inconsistency of the model.

Resulting OWL model of a message which is contained in a NL text document could be used by a potential user for evaluation message pertinence.

3.3 Goal Driven Learning Process

The process of ontology learning was widely discussed last years [12]. But all elaborated approaches don't take into account the problem of reaching the optimal structure and dimension of ontology. It is known that computational complexity of graph processing algorithms doesn't enable to work, at least, in the real time scale, with the especially large ontology, for example in a case of solving tasks of eliminating the knowledge inconsistency. Therefore it's extremely important not just simply add the all accessible relevant knowledge data to the ontology, but to refine the existing knowledge structures inside of the defined optimal volume. This task must follow some criteria of optimality, most valuable of which is the data usefulness, applicability or, other words, knowledge pertinence.

To estimate the data usefulness as the optimality main criterion is possible to count the mention frequency of connected concepts and relations in ontology. This is a formal evidence of the data importance, because the ontology represents a sphere of agent interests if the learning procedure is correct enough [13]. The more precise approach has take into account the main task of agent's activity [14] or, to be more specific, the tasks hierarchy and a gain from its solving as the most adequate criterion of the information usefulness. It's possible to solve this task because the ontology should contain an explicit specification of that task hierarchy. Moreover, if a first dimension of ontology structure is the taxonomy, a second one must be built as a HTN.

The HTN ontology concept is described in [8]. The ontology learning process is based on NLP procedures with the aim to find out subtasks of the agent's main task, the appropriate task solving strategies and their components – resources, actions (operators), methods, preconditions, effects, dependencies and etc. It's performed on different levels of text coverage – from separate found term definitions up to the whole message meaning as a problem solving recipe.

The goal driven learning process includes next stages: (i) a task from own ontology, which is appropriate to the task described in a message, extracted from NL text document, must be selected; (ii) a new approach taken from message is applied to solve the selected task; (iii) a maximum expected utility of a new strategy is estimated and compared to the previous one; (iv) if the utility value is changing significantly and the message reliability is high enough then the ontology is corrected, otherwise the message data is stored in a knowledge base as a reference information about it's source.

An agent learns also the different task solving patterns - decision making using different heuristics. In one of them for simplicity each task could be represented as the POMDP model. It is a generalization of the standard completely observable Markov Decision Process (MDP) that allows estimating the optimal strategy using the imperfect (incomplete) information about the system state. Such formalism possesses well by developed solving algorithms, and it's useful therefore for implementation [15].

3.4 Knowledge Pertinence Evaluation

All needed procedures for evaluating the knowledge are contained in the analyzed message. It depends on message context which is explicitly expressed in the agent ontology and the optimal strategy for reaching its goal. The evaluation method is based on the Expected Value of Perfect Information

$$\mathrm{EVPI} = \mathrm{EV|PI} - \mathrm{EMV}, \tag{3}$$

where EMV is the probability weighted sum of possible payoffs per each alternative;

$$\mathrm{EMV} = \max_i \sum_j p_j R_{ij},\ \sum_j p_j R_{ij} - \text{is the expected payoff for action } i;$$

EV|PI is the expected or average return if we priori have the “perfect” (i.e. new) information for best i choice:

$$\mathrm{EV|PI} = \sum_j p_j (\max_i R_{ij}). \tag{4}$$

To estimate new knowledge EVPI we must have the EMV value for each solving approach (action for (3)) to each task from HTN of our ontology. To obtain them all we have create and solve the appropriate POMDP task:

$$EMV_i \equiv U(S_i) = R(S_i) + \gamma \cdot \max_{A_{ik}} \sum_k P(S_i, A_{ik}, S_j) U(S_j). \tag{5}$$

An Eq. (5) describes the reward for taking the action giving the highest expected return. The additional information decreases the model uncertainty and the expected common reward (utility) is not less than without such information.

If we use POMDP algorithms, such as value and policy iteration as well as gradient ascent algorithms [16, 17] for particular domain model, then we may evaluate the expected utility for both cases: with new information and without it.

3.5 Reliability Evaluation

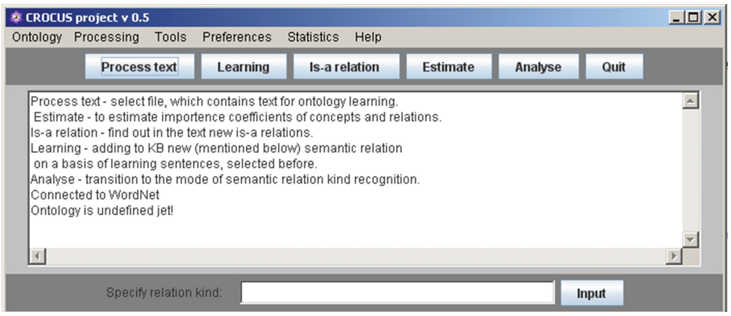
All obtained information is verified for its consistency. If any contradiction appears then a logical conflict is solving by rejecting data with less reliability. In such case the reliability of the source D also decreases:

$$D_{n,i+1} = \frac{D_{n,i}}{(2-s)} + \frac{1-D_{n,i}}{2} \cdot s \tag{6}$$

where: s is the truth of the statement. It takes a value 1 if the statement is true or 0 – otherwise; i is the step number confirmation/denial of the truth of the one statement of n-th source.

4 Implementation

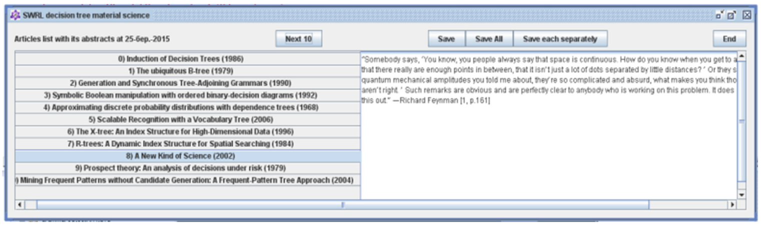
The presented OL concept above was implemented in the system of Cognition Relations Or Concepts Using Semantics (CROCUS). The system is built on Java Protégé-OWL API using OWL-DL as a knowledge representation language. The WordNet API was added there for providing the recognition of new domain concepts at the interactive learning mode. Using the DBMS MySQL in the CROCUS system it's possible to store, process and employ a statistics for many domains and users simultaneously and independently during the learning process (Fig. 2).



```
Process text - select file, which contains text for ontology learning.
 Estimate - to estimate importence coefficients of concepts and relations.
Is-a relation - find out in the text new is-a relations.
Learning - adding to KB new (mentioned below) semantic relation
 on a basis of learning sentences, selected before.
Analyse - transition to the mode of semantic relation kind recognition.
Connected to WordNet
Ontology is undefined jet!
```

Fig. 2. CROCUS main interface.

It performs two main interdependent functions: learning the ontology in both supervised and unsupervised modes, and searching the new knowledge in articles abstracts using ontology to recognize and estimate a measure of its importance (Fig. 3). CROCUS structure includes a set of tools preparing the training texts for ontology learning. Standard means allow tuning all basic interfaces on needed language. A whole system is an open-source product with a free license.



Articles list with its abstracts at 25-6ep.-2015

0) Induction of Decision Trees (1986)
1) The ubiquitous B-tree (1979)
2) Generation and Synchronous Tree-Adjoining Grammars (1990)
3) Symbolic Boolean manipulation with ordered binary-decision diagrams (1992)
4) Approximating discrete probability distributions with dependence trees (1968)
5) Scalable Recognition with a Vocabulary Tree (2006)
6) The X-tree: An Index Structure for High-Dimensional Data (1996)
7) R-trees: A Dynamic Index Structure for Spatial Searching (1984)
8) A New Kind of Science (2002)
9) Prospect theory: An analysis of decisions under risk (1979)
) Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach (2004)

"Somebody says, 'You know, you people always say that space is continuous. How do you know when you get to a
that there really are enough points in between, that it isn't just a lot of dots separated by little distances?' Or they s
quantum mechanical amplitudes you told me about, they're so complicated and absurd, what makes you think tho
aren't right.' Such remarks are obvious and are perfectly clear to anybody who is working on this problem. It does
this out." —Richard Feynman [1, p.161]

Fig. 3. Data search results interface.

5 Conclusions

A new approach to the data integration as a goal driven ontology learning is proposed and the pertinence evaluation method as the key technology was elaborated. It's shown: if the pertinence of the new information is under the threshold value then such information will be rejected from the ontology learning process as the irrelevant one to the goal of the domain. To build the agent optimal strategy for solving the defined task means for automated ontology learning were elaborated. It's implemented in the Java programming language as the semantic analysis package CROCUS and tested on English educational texts.

A whole framework could serve as a prototype for developed knowledge discovery means. That helps to supply ontology learning tools with the pertinent information and use such learned ontology to evaluate the information pertinence.

The elaborated approach of the goal driven ontology learning in the framework of the hierarchical task network-OWL planning can be widely implemented using the existing open source software libraries. Their fast development presages an appropriate development of the all relevant scientific fields: natural language processing, automated planning and ontology learning. That changes revolutionary the information search technique and means. But before the developed approach can be widely used, a developing the new ontology architecture and ontology learning paradigm are the main challenges on that way.

Acknowledgments. This work is supported by China 973 fundamental research and development project, grant number 2014CB340404; the National Natural Science Foundation of China, grant number 61373037.

References

1. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw Hill, New York (1983)
2. Meadow, C.T., et al.: Text Information Retrieval Systems. Elsevier, Burlington (2007)
3. PubMed Celebrates its 10th Anniversary; Technical Bulletin. United States National Library of Medicine. 2006-10-05. Cited 22 March 2011
4. Jacso, P.: The impact of Eugene Garfield through the prism of web of science. Ann. Libr. Inf. Stud. **57**, 222 (2010)
5. Muller, H.M., Kenny, E.E., Sternberg, P.W.: Textpresso: an ontology-based information retrieval and extraction system for biological literature. PLoS Biol. **2**(11), e309 (2004). doi:10.1371/journal.pbio.0020309
6. Tschantz, M.C.: Formalizing and enforcing purpose restrictions. Ph.D. thesis (2012)
7. Sirin, E., Parsia, B.: Planning for semantic web services. In: Proceedings of the Semantic Web Services Workshop at 3rd International Semantic Web Conference (ISWC 2004) (2004)
8. Bouillet, E., Feblowitz, M., Liu Z., Ranganathan, A., Riabov, A.: A knowledge engineering and planning framework based on OWL ontologies. In: Proceedings of the Second International Competition on Knowledge Engineering (ICKEPS 2007) (2007)
9. Freitas, A., Schmidt, D., Meneguzzi, F., Vieira, R., Bordini, R.H.: Using ontologies as semantic representations of hierarchical task network planning domains. In: Proceedings of WWW (2014)
10. Horridge, M., Bechhofer, S.: The OWL API: a Java API for OWL ontologies. Semant. Web **2**(1), 11–21 (2011)

11. Sleator D., Temperley D.: Parsing English with a link grammar. Carnegie Mellon University Computer Science Technical report CMU-CS-91-196, October 1991
12. Wong, W., Liu, W., Bennamoun, M.: Ontology learning from text: a look back and into the future. ACM Comput. Surv. (CSUR) **44**(4), 20 (2012)
13. Lytvyn, V., Medykovskyj, M., Shakhovska, N., Dosyn, D.: Intelligent agent on the basis of adaptive ontologies. J. Appl. Comput. Sci. **20**(2), 71–77 (2012)
14. Arboleda, H., Paz, A., Jiménez, M., Tamura, G.: A framework for the generation and management of self-adaptive enterprise applications. In: 10th Computing Colombian Conference (10CCC) (2015)
15. Hauskrecht, M.: Value-function approximations for partially observable Markov decision processes. JAIR **13**, 33–94 (2000)
16. Braziunas, D.: POMDP solution methods, Technical report, Department of Computer Science, University of Toronto (2003)
17. Halbert, T.R.: An Improved Algorithm for Sequential Information-Gathering Decisions in Design under Uncertainty. Master's thesis, Texas A&M University (2015). http://hdl.handle.net/1969.1/155384

An Infrastructure and Approach for Inferring Knowledge Over Big Data in the Vehicle Insurance Industry

Aikaterini K. Kalou(✉) and Dimitrios A. Koutsomitropoulos

High Performance Information Systems Laboratory (HPCLab),
Computer Engineering and Informatics Department, School of Engineering,
University of Patras, Building B, 26500 Patras-Rio, Greece
{kaloukat, kotsomit}@hpclab.ceid.upatras.gr

Abstract. In recent years, insurance organizations have turned their attention to tapping into massive amounts of data that are continuously produced across their IT ecosystem. Even though the concept of Big Data provides the needed infrastructure for efficient data management, especially in terms of storage and processing, the aspects of Value and Variety still remain a topic for further investigation. To this end, we propose an infrastructure that can be deployed on top of the legacy databases of insurance companies. The ultimate aim of this attempt is to provide an efficient manner to access data on-the-fly and derive new value. In our work, we propose a Property and Casualty ontology and then exploit an OBDA system in order to leverage its power.

Keywords: Insurance sector · Big data · Ontology · Linked data · OBDA

1 Introduction

In recent years, an impressive growth of generated data has been considered across business and government organizations. New information is constantly added, and existing information is continuously changed or removed, in any format and coming from any type of data sources (internal and external). So, the manipulation of these massive amounts of data went beyond the power and the performance of conventional processes and tools. At the same time, the volume of data offers greater and broader opportunities for developing existing business areas or driving new ones by improving on insight, decision-making and detecting sources of profit [15, 17].

Big Data [18], designated among the most common buzzwords dominating the IT world, can cover efficiently and effectively the aforementioned need and thus they are currently influencing most aspects of business units. In brief, the new technological achievement can be thoroughly described by the following characteristics: Variety (the number of types of data), Velocity (the speed of data generation and processing), Volume (the amount of data) [21] and Value (the knowledge). However, the lack of semantics seems to be restrictive towards the 4th V of Big Data. The aspect of the 1st V of Big Data also remains a topic for further discussion.

Insurance is a sector with traditionally high dependency in data availability as well as knowledge-based decision making. Managing big volumes of insurance information not only has to face associated technical limitations, but also requires expressive knowledge analysis strategies to become valuable. Semantic technologies, including either the more lightweight (Linked Data) or the most expressive (ontology) version, can have considerable repercussions in addressing the Variety and the Value of Big Data for insurance. Regarding the Linked Data principles [11] the whole data management can be easily simplified. For example, data coming from unstructured databases and agents can be linked to data in traditional OLTP and OLAP stores without changing existing schemas.

On the other side, an ontology, by default, defines a common set of terms that are used to describe and represent the basic concepts in a domain and the relationships among them. With the power of reasoning, new facts, which are unexpressed explicitly in an ontology, can then be derived. Therefore, by exploiting the notion of ontology in the context of big data, new perspectives are added such as providing semantics to raw data, connecting data in various concept levels across domains and giving knowledge for further analysis, including more accurate risk identification and assessment. Customized insurance policies, fraud detection and marketing are only a few of the areas that can be identified as benefiting from applying semantic analytics methods over Big Data [9].

In this work, we make an attempt to take advantage of all the aforementioned benefits that Semantic Web technologies provide in terms of Big Data by implementing a real usage scenario in the Insurance sector. More precisely, based on the Property & Casualty Ontology [6] and a big dataset coming from an existing insurance company, we develop an infrastructure that could easily consume various data and infer knowledge in reasonable time and without sacrificing performance. The key component of the proposed infrastructure is an OBDA (Ontology Based Data Access [13, 19]) system that ensures (i) scalable big data infrastructure, (ii) end-to-end processing of data and (iii) managing the diverse roles of people in the data life cycle.

The rest of this paper is organized as follows. In Sect. 2, we start by discussing the usage of Ontologies in Insurance Business and providing some broad definitions and concepts about semantic technologies over Big Data and the approach of OBDA. Furthermore, in Sect. 3, we present in short the P&C ontology. Section 4 elaborates the architecture of our proposed system and the data manipulation strategy that we adopt. Next, Sect. 5 outlines several intelligent queries performed over real insurance data in order to illustrate the ability for knowledge inference. Finally, Sect. 6 summarizes our conclusions.

2 Background

2.1 Big Data Ontology Access

In recent years, quite a few data storage and processing technologies have emerged in terms of Big Data. Platforms like NoSQL, Yarn, Hadoop, MapReduce, HDFS are now some of the most familiar terms within this growing ecosystem [22]. On the semantic technologies' side, the "traditional" triplestores are continually evolving by following the vision of exploring large data sets. Oracle Spatial and Graph with Oracle Database 12c, AllegroGraph, Stardog, OpenLink Virtuoso v6.1 are only some examples that expand their scalability and performance features to meet these premium requirements.

A lot of initiatives and synergies are on progress, having as main purpose the smooth integration of the widely-known Big Data technologies with whatever is coming from the area of semantic technologies. For example, AllegroGraph 6, has recently been released with a certification on Cloudera Enterprise[1], among the leader companies of Apache Hadoop-based software. Furthermore, AllegroGraph has implemented extensions allowing users to query MongoDB databases using SPARQL and to execute heterogeneous joins [10].

Besides all the above notable new features of triplestores, the issue of querying data from various legacy relational databases still stumbles on the required ETL (Extract – Transform - Load) processes [20]. In these cases, the data flows can be summarized in extracting the data from the database, transforming into triples and then storing in conventional RDF stores. The approach of OBDA systems seems to cope well with this scenario by leveraging the pure nature of ontology notion. With OBDA, an ontology is used so as to expose data in a conceptual manner by abstracting away from the schema-level details of the underlying data. The connection between data and ontology is performed via mappings. The combination of ontology and mappings allows to automatically translate queries posed over the ontology into data-level queries that can be executed by the specific underlying database management system. Ontop[2], Mastro[3], Stardog[4] are among the most popular OBDA systems.

2.2 Modeling the Insurance Sector

The necessity for big data manipulation and analysis is only currently emerging in the insurance industry, possibly due to the tight margin profits and increasing competition [8]. A starting point for the semantic formalization of the business functions and associated processes of the Insurance sector can be offered by the Property and Casualty Data Model [5], developed by the Object Management Group [16]. Organizations that incorporate the aforementioned data model into their procedures can be substantially benefited. P&C data model can be applied in a wide variety of warehousing, business intelligence, and data migration projects. In addition, this model can be leveraged to improve the use of data and to support data governance within the enterprise.

The major components of the P&C data model are the entities, attributes and relationships. An *Entity* can stand for a person, organization, place, thing, or concept of interest to the enterprise by covering a very broad and varied set of instance data, or something very specific. This idiom is referred to as *levels of abstraction*.

---

[1] http://franz.com/about/press_room/ag-6.0_2-8-2016.lhtml.

[2] http://ontop.inf.unibz.it/.

[3] http://www.dis.uniroma1.it/~mastro/?q=node/2?.

[4] http://stardog.com/.

A *Relationship* should be a verb or a verb phrase that is always established between two entities. A relationship is used in order to form a sentence between its two entities. Moreover, the type of relationship can be 'Identifying', 'Non-identifying' or 'Subtype'.

Attributes are usually defined within an entity and are considered as properties or descriptors for this entity. An attribute is meaningless by itself. Every attribute in the data model is connected to a domain that provides for consistent names, data types, lengths, value sets, and validity rules. The main elements in order to define an attribute for an entity are the Attribute Name, Entity Name and Data Type.

Figure 1 illustrates how several entities, such as Insurable Object, Vehicle, Claim and Claim Folder, can be specified in terms of attributes and how they can be linked to each other via relationships in the context of the P&C data model.

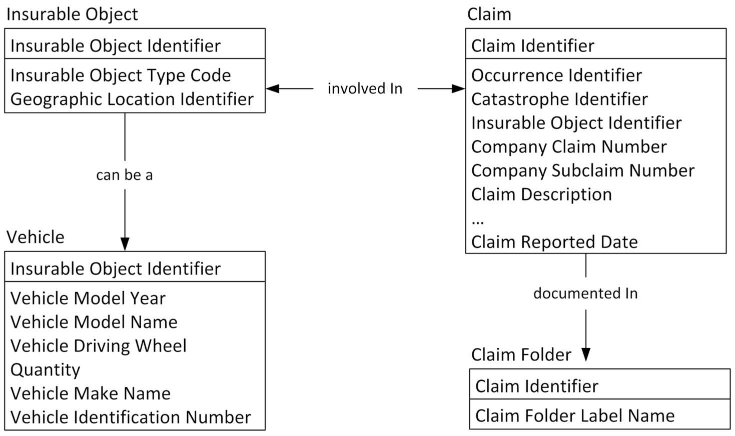


Fig. 1. A snippet of the P&C data model.

3 Insurance Ontology

In this section, we describe briefly how an ontology, fully expressed in the Web Ontology Language (OWL) [2], can be designed by considering the Property and Casualty model. We present how data entities representing most of P&C insurance business processes can be converted into ontology components. Taking into account the InfoSphere Data Architect guidelines [14], we can simply correlate logical data model's data types to OWL data types, logical data model's entities to ontology elements as well as logical data model's relationships to OWL object properties. In detail, Table 1 gathers all the restrictions that must be followed for an effective conversion of these relationships.

Figure 2 depicts part of the neighborhood of the *InsurableObject, Claim, Agreement, Person* and other major entities of the model, presented as classes in the resulting ontology, based on the class hierarchy and property relations that may associate instances of one class with another.

Table 1. Logical data model to OWL object mappings.

Logical data model	OWL ontology elements
Entity	**owl:Class**
Entity - Name	rdf:about
Entity - Label	rdfs:label
Entity - Namespace	Namespace of owl:Class
Entity - Definition	rdfs:comment
Entity - Primary Key	owl:hasKey
Relationship	**owl:ObjectProperty**
Relationship – Name	rdf:about
Relationship – Label	rdfs:label
Relationship – Owner	rdfs:domain
Relationship – Namespace	Namespace of owl:ObjectProperty
Relationship – Child Entity	rdfs:domain
Relationship – Parent Entity	rdfs:range
Relationship – Annotation	rdfs:seeAlso, rdfs:DefinedBy, owl:FunctinalProperty,owl:InverseFunctionalProperty owl:AnnotationProperty
Attribute	**owl:DatatypeProperty**
Attribute – Name	rdf:about
Attribute – Label	rdfs:label
Attribute – Namespace	rdfs:domain
Attribute – Data Type	rdfs:range
Attribute – Length/Precision	xsd:maxLength, xsd:length, xsd:totalDigits
Attribute – Scale	xsd:fractionDigits
Attribute – Primary Key	owl:hasKey
Attribute – Documentation	rdfs:comment
Attribute – Annotation	rdfs:seeAlso, rdfs:isDefinedBy, owl:AnnotationProperty

4 Architecture

4.1 Data Manipulation Strategy

In our experiments, we considered large volumes of offline data originating from the data archives of a well-known vehicle insurance company. Data were actually SQL dumps into CSV files, each corresponding to a separate relational table. Each of the 5 tables includes about 0.5M tuples containing between 16–56 columns, resulting in 31 columns on average. For a trivial mapping, a single column of each tuple can form a seperate RDF triple, therefore the resulting graph would contain about 77M triples.

To facilitate access to these data and to enable intelligent queries on top of them, we use an OBDA system, namely Ontop [4]. Ontop was selected based on its ease of use, intuitive mapping support, and high performance capabilities. In addition, Ontop supports reasoning at the OWL 2 - QL level, which is a lightweight reasoning profile, but expressive enough to support inferences on very large volumes of instance data [7].

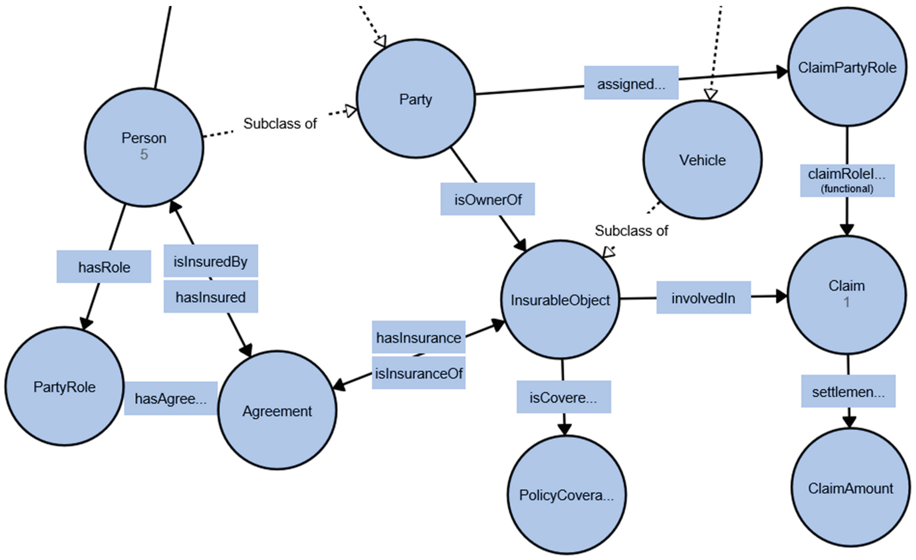


Fig. 2. The mapping of *InsurableObject* and other entities in the P&C ontology.

In systems such as Ontop, there are two main elements, an *ontology* which models the application domain by using a shared vocabulary, and the *mappings,* which relate the ontological terms with the schemata of the underlying data sources. Therefore, the ontology and mappings combined, expose a setting, so that end-users can formulate queries without deeper understanding of the data sources, the relation between them, or the encoding of the data.

The outcome of the mappings in conjunction with the defined ontology can be thought as an RDF view, which can be materialised or not. In the materialization case, the data are triplified and then are directly used within an RDF triplestore without requiring additional interaction with the initial data sources. In the second case, which is the one followed by our approach, the RDF view is called a *virtual graph* and can be queried at query execution time, therefore allowing for on-the-fly ontology based access to and inference on data. At the same time, this approach relieves from the burden of data replication.

In our setting, the ontology is offered by our implementation of the P&C model into OWL, described in the previous section. To construct the mappings, we used the mapping tool integrated within the ontop Protégé plugin. Some of the mappings designed to create the virtualized graph are shown in Table 2 in the form of *ID*, *Source* and *Target*.

4.2 Infrastructure

After the ontology definition and the design of mappings, the next step is to set up how raw data are consumed. Ontop can be configured to access DBMS data or other

Table 2. A partial set of mappings used for the semantic modeling of insurance data.

ID	Source (SQL query)	Target (Triples template)
1	SELECT plate, contract FROM InsuredItemVehicle	:{plate} a :Vehicle ; :hasInsurance :{contract} . :{contract} a :Policy.
2	SELECT customerCode as c, iv.plate FROM InsuredItemCustomer as ic, InsuredItemVehicle as iv WHERE ic.contract = iv.contract;	:{c} a :Person ; :isOwnerOf :{plate}.
3	SELECT policyNumer as pol, ClaimNumber as r, totalPayAmount as amnt, iv.contract, plate FROM Claims, InsuredItemVehicle as iv where pol = iv.contract;	:Amount_{pol}_{r} :hasAmount {amnt}^^xsd:decimal . :Claim_{pol}_{r} a :Claim ; :settlementResultsIn : Amount_{pol}_{r} . :{plate} : involvedIn :Claim_{pol}_{r}

sources, through the Teiid data virtualization platform[5]. For our purposes, we interfaced the data dumps through a JDBC driver, by means of a H2/MySQL database. Given the appropriate mappings, it is also possible to include data flows from other sources as well, such as federated databases, insurance brokes, agents filling in claims, vehicle sensors and other possibly unstructured data, in a similar manner.

Having established the connection to data, queries can then be performed using SPARQL. To this purpose, Ontop exposes a SPARQL interface via the Protégé plugin. For query evaluation, the system parses the query and communicates with its internal reasoner module, which has already loaded the ontology, in order to infer any implicit semantic implications. Next, the original query is internally transformed into one or more SQL queries addressed to the federation of data sources, by considering the mappings already defined and the outcomes of the reasoning proccess.

As a result, query answering and reasoning take place on-the-fly, without the need to replicate data into ontology instances first or materialize the graph resulting from the mappings. This also means that online updates to data are directly reflected into the answers received by SPARQL queries, as all evaluation occurs during query execution time. Ontop can also be used as a standard SPARQL HTTP endpoint by extending the Sesame Workbench, a web application for administering RDF repositories [3]. The data flow and the query evaluation process are depicted in Fig. 3.

5 Intelligent Queries on Insurance Data

A set of example queries over the insurance dataset is presented in this section. All three queries involve some form of reasoning, so as to demonstrate the added-value ontology-based inferences can have on insurance data. They are also indicative of the expressivity of logical axioms allowed in OWL 2 - QL, like inheritance, hierarchy, inverse property and existential restriction.

[5] http://teiid.jboss.org/.

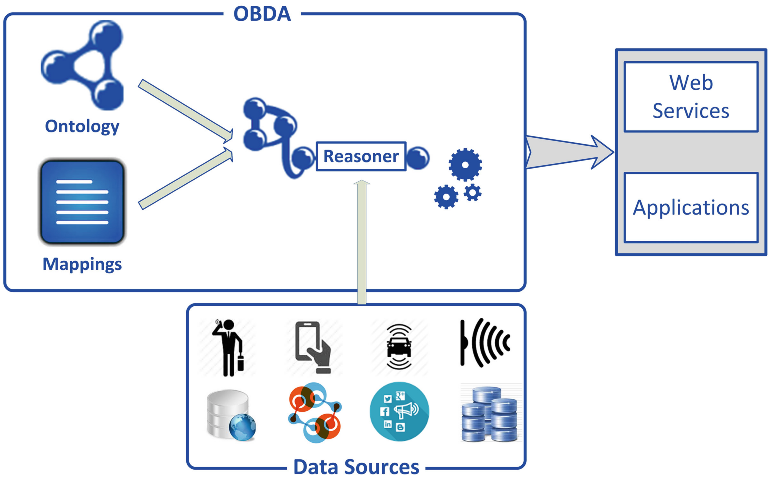
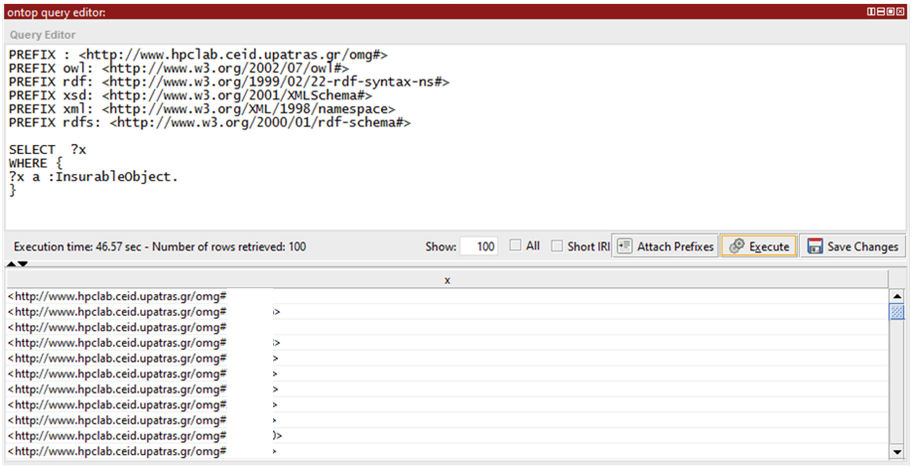


Fig. 3. Architecture of the experimental setup for data flow and query answering.

In the first query, shown in Fig. 4, even though we have not designed a mapping rule defining instances of the *InsurableObject* classes, the instances of *Vehicle* are automatically classified as such, due to *Vehicle* being defined as subclass of *InsurableObject* in the ontology (see Fig. 2).

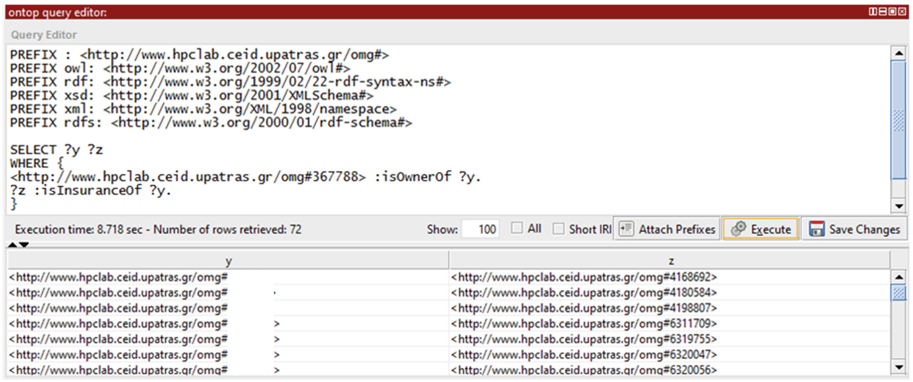


```
PREFIX : <http://www.hpclab.ceid.upatras.gr/omg#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX xml: <http://www.w3.org/XML/1998/namespace>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT  ?x
WHERE {
?x a :InsurableObject.
}
```

```
Execution time: 46.57 sec - Number of rows retrieved: 100

x
<http://www.hpclab.ceid.upatras.gr/omg#
<http://www.hpclab.ceid.upatras.gr/omg#        >
<http://www.hpclab.ceid.upatras.gr/omg#
<http://www.hpclab.ceid.upatras.gr/omg#        >
<http://www.hpclab.ceid.upatras.gr/omg#        >
<http://www.hpclab.ceid.upatras.gr/omg#        >
<http://www.hpclab.ceid.upatras.gr/omg#        >
<http://www.hpclab.ceid.upatras.gr/omg#        >
<http://www.hpclab.ceid.upatras.gr/omg#        >
<http://www.hpclab.ceid.upatras.gr/omg#        )>
<http://www.hpclab.ceid.upatras.gr/omg#        >
```

Fig. 4. SPARQL query – Example 1.

The next example, shown in Fig. 5, retrieves the insurance policies for all vehicles that are owned by a specific *Party*. Note that we are able to use the *isInsuranceOf* property in the query, instead of *hasInsurance* as specified by mapping #1 in Table 2, because they are defined as inverses in the ontology (see Fig. 2).

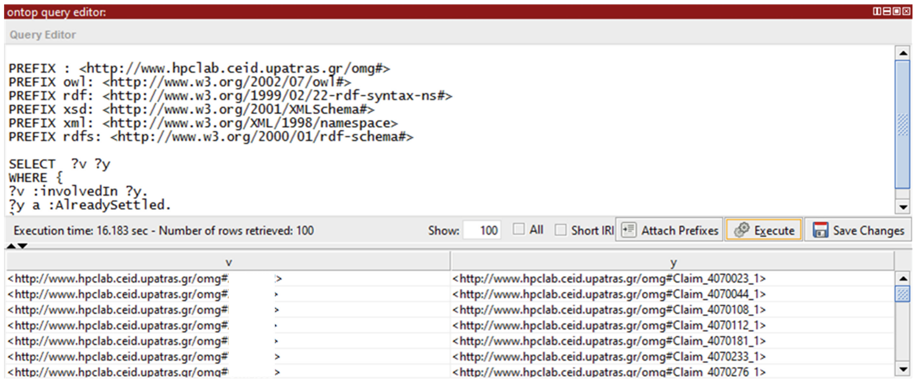


```
PREFIX : <http://www.hpclab.ceid.upatras.gr/omg#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX xml: <http://www.w3.org/XML/1998/namespace>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?y ?z
WHERE {
<http://www.hpclab.ceid.upatras.gr/omg#367788> :isOwnerOf ?y.
?z :isInsuranceOf ?y.
}
```

Execution time: 8.718 sec - Number of rows retrieved: 72

y	z
<http://www.hpclab.ceid.upatras.gr/omg#	<http://www.hpclab.ceid.upatras.gr/omg#4168692>
<http://www.hpclab.ceid.upatras.gr/omg#	<http://www.hpclab.ceid.upatras.gr/omg#4180584>
<http://www.hpclab.ceid.upatras.gr/omg#	<http://www.hpclab.ceid.upatras.gr/omg#4198807>
<http://www.hpclab.ceid.upatras.gr/omg# >	<http://www.hpclab.ceid.upatras.gr/omg#6311709>
<http://www.hpclab.ceid.upatras.gr/omg# >	<http://www.hpclab.ceid.upatras.gr/omg#6319755>
<http://www.hpclab.ceid.upatras.gr/omg# >	<http://www.hpclab.ceid.upatras.gr/omg#6320047>
<http://www.hpclab.ceid.upatras.gr/omg# >	<http://www.hpclab.ceid.upatras.gr/omg#6320056>

Fig. 5. SPARQL query – Example 2.



```
PREFIX : <http://www.hpclab.ceid.upatras.gr/omg#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX xml: <http://www.w3.org/XML/1998/namespace>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT  ?v ?y
WHERE {
?v :involvedIn ?y.
?y a :AlreadySettled.
```

Execution time: 16.183 sec - Number of rows retrieved: 100

v	y
<http://www.hpclab.ceid.upatras.gr/omg# >	<http://www.hpclab.ceid.upatras.gr/omg#Claim_4070023_1>
<http://www.hpclab.ceid.upatras.gr/omg# >	<http://www.hpclab.ceid.upatras.gr/omg#Claim_4070044_1>
<http://www.hpclab.ceid.upatras.gr/omg# >	<http://www.hpclab.ceid.upatras.gr/omg#Claim_4070108_1>
<http://www.hpclab.ceid.upatras.gr/omg# >	<http://www.hpclab.ceid.upatras.gr/omg#Claim_4070112_1>
<http://www.hpclab.ceid.upatras.gr/omg# >	<http://www.hpclab.ceid.upatras.gr/omg#Claim_4070181_1>
<http://www.hpclab.ceid.upatras.gr/omg# >	<http://www.hpclab.ceid.upatras.gr/omg#Claim_4070233_1>
<http://www.hpclab.ceid.upatras.gr/omg# >	<http://www.hpclab.ceid.upatras.gr/omg#Claim_4070276_1>

Fig. 6. SPARQL query – Example 3.

With mapping #3, we relate a *Vehicle* to a *Claim* and the *Claim* to its resulting settlement *ClaimAmount*. The following query (Fig. 6) discovers vehicles *involvedIn Claims* that have already been settled with a *ClaimAmount* through *settlementResultsIn*. This is possible using the auxiliary class *AlreadySettled*, specified as an existential restriction on the *settlementResultsIn* property.

6 Conclusions and Future Work

Nowadays, the data ecosystem of a typical insurance enterprise is significantly expanded by including not only internal databases but also third-party data sources such as social media, smartphones, sensors and other consumer and industrial devices. To this data heap, the public-sector data, recently available in the form of "Open Data" [12], can be appended. Even though such proliferation and variety of data can considerably profit the insurance sector by providing additional inputs to alleviate fraud and risk management, the obstacle of data gathering from different access points in different formats, as well as its semantic analytics, still remains.

In this work, we have exploited the capabilities of OBDA systems so as to highlight the contribution of semantic technologies in the industry of vehicle insurance. Having at our disposal a voluminous dataset from an existing insurance company, we set an infrastructure to inference knowledge in an efficient manner. The key component is our proposed P&C ontology and a set of logical axioms and mappings that correlate the data with the ontology. We have shown that the ontology can be utilized as an intermediate level for accessing directly legacy, existing databases and perform reasoning-enabled queries on them.

While graph materialization appears to perform well with Ontop, there is some delay when importing raw data into the relational schema. This may prove a bottleneck further on, so it might be worth investigating further concurrent data flow approaches for RDF virtualization and SPARQL querying, like C-SPARQL [1].

References

1. Barbieri, D.F., Braga, D., Ceri, S., Della Valle, E., Grossniklaus, M.: Querying RDF streams with C-SPARQL. SIGMOD Rec. **39**(1), 20–26 (2010)
2. Bechhofer, S., Van Harmelen, F., Hendler, J., Horrocks, I., Mc Guinness, D.L., Patel-Schneider, P.F., Stein, L.A.: OWL Web Ontology Language Reference, W3C Recommendation http://www.w3.org/TR/2004/REC-owl-ref-20040210/
3. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: a generic architecture for storing and querying RDF and RDF Schema. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 54–68. Springer, Heidelberg (2002)
4. Calvanese, D., Cogrel, B., Komla-Ebri, S., Kontchakov, R., Lanti, D., Rezk, M., Rodriguez-Muro, M., Xiao, G.: Ontop: answering SPARQL queries over relational databases. Semantic Web – Interoperability, Usability, Applicability (2016, in Press). ISSN: 1570-0844
5. Jenkins, W., Molnar, R., Wallman, B., Ford, T.: Property and Casualty Data Model Specification (2011)
6. Kalou, A.K., Koutsomitropoulos, D.A.: Linking data in the insurance sector: a case study. In: Iliadis, L., Maglogiannis, I., Papadopoulos, H., Sioutas, S., Makris, C. (eds.) AIAI 2014. IFIP AICT, vol. 437, pp. 320–329. Springer, Heidelberg (2014)
7. Lanti, D., Rezk, M., Xiao, G., Calvanese, D.: The NPD benchmark: reality check for OBDA systems. In: Proceedings of the 18th International Conference on Extending Database Technology (EDBT), pp. 617–628 (2015)
8. Llull, E.: Big data analysis to transform insurance industry. Technical article, Financial Times (2016)
9. Marr, B.: How Big Data is changing insurance forever. Technical article, Forbes (2015)
10. Michel, F., Faron-Zucker, C., Montagnat, J.: A mapping-based method to query MongoDB documents with SPARQL. In: Hartmann, S., Ma, H. (eds.) DEXA 2016. LNCS, vol. 9828, pp. 52–67. Springer, Heidelberg (2016). doi:10.1007/978-3-319-44406-2_6
11. Mitchell, I., Wilson, M.: Linked Data: Connecting and exploiting big data. White paper. Fujitsu UK (2012)

12. World Bank Group. Transport and ICT: Open Data for Sustainable Development. Technical report (2015)
13. Poggi, A., Lembo, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Linking data to ontologies. In: Spaccapietra, S. (ed.) Journal on Data Semantics X. LNCS, vol. 4900, pp. 133–173. Springer, Heidelberg (2008)
14. Soares, S.: IBM InfoSphere: A Platform for Big Data Governance and Process Data Governance. MC Press Online, LLC, February 2013
15. Sodenkamp, M., Kozlovskiy, I., Staake, T.: Gaining IS business value through big data analytics: a case study of the energy sector. In: Proceedings of the Thirty Sixth International Conference on Information Systems (ICIS), Fort Worth, USA, pp. 13–16 (2015)
16. The Object Management Group (OMG). MDA Guide Version 1.0.1 (2003)
17. Tsai, C.W., Lai, C.F., Chao, H.C., Vasilakos, A.C.: Big data analytics: a survey. J. Big Data **2**(21), 1–32 (2015)
18. Ylijoki, O., Porras, J.: Perspectives to definition of big data: a mapping study and discussion. J. Innov. Manag. **4**(1), 69–91 (2016)
19. Lenzerini, M.: Ontology-based data management. In: Proceedings of CIKM 2011, pp. 5–6 (2011)
20. Rodriguez-Muro, M., Calvanese, D.: Quest, an OWL 2 QL reasoner for ontology-based data access. In: Proceedings of the 9th International Workshop on OWL: Experiences and Directions (OWLED 2012). CEUR Electronic Workshop Proceedings, vol. 849 (2012)
21. Laney, D.: 3D data management: Controlling data volume, velocity and variety. META Group Research Note 6, 70 (2001)
22. Press, G.: Top 10 hot big data technologies. Technical article. Forbes (2016)

Defining and Identifying Stophashtags in Instagram

Stamatios Giannoulakis(✉) and Nicolas Tsapatsoulis

Department of Communication and Internet Studies, Cyprus University of Technology, 30, Arch. Kyprianos str., CY-3036 Limassol, Cyprus
{s.giannoulakis,nicolas.tsapatsoulis}@cut.ac.cy

Abstract. Instagram could be considered as a tagged image dataset since it is reach in tags -known as hashtags- accompanying photos and, in addition, the tags are provided by photo owners/creators, thus, express in higher accuracy the meaning/message of the photos. However, as we showed in a previous study, only 30 % of Instagram hashtags are related with the visual content of the accompanied photos while the remaining 70 % are either related with other meta-communicative functions of the photo owner/creator or they are simply noise and are used mainly to increase photo's localization and searchability. In this study we call the latter category of Instagram hashtags as 'stophashtags', inspired from the term 'stopwords' which is used in the field of computational linguistics to refer to common and non-descriptive words found in almost every text document, and we provide a theoretical and empirical framework through which stophashtags can be identified. We show that, in contrary to descriptive hashtags, stophashtags are characterized by high normalized subject (hashtag) frequency on irrelevant subject categories while normalized image frequency is also high.

Keywords: Instagram · Hashtags · Image tagging · Noise · Stopwords · Machine learning

1 Introduction, Motivation and Related Work

Photo sharing is a common phenomenon in the Web. About 300 million photos are uploaded on Facebook per day[1] and another 80 million are uploaded on Instagram[2]. Instagram users from January 2011 onwards have the possibility to annotate their photos with hashtags [2] and since April 2015 they are able to use emoji as hashtags. Hashtags are tags or words prepended with # and used to indicate the content of pictures allowing their localization and increasing their visibility through the Instagram search engine. Emoji are pictograms, which presents emotions a user wants to give to a picture. Instagram hashtags can describe the visual content of an image but there are also noisy tags

[1] The Top 20 Valuable Facebook Statistics. Online at: https://zephoria.com/top-15-valuable-facebook-statistics/.

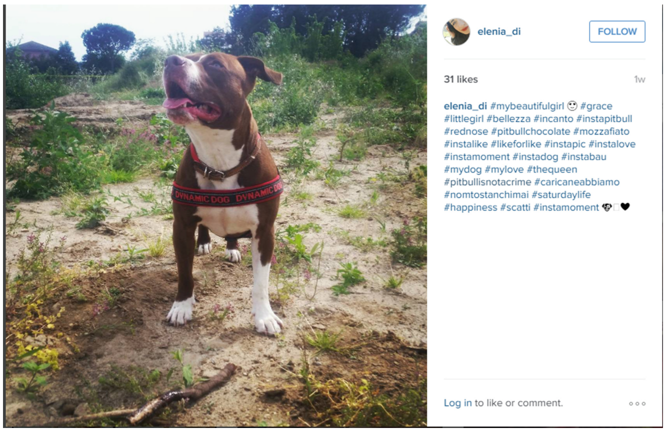
[2] Instagram: Stats. Online at: https://www.instagram.com/press/?hl=en.

and hashtags related to metacommunicative use. For instance, Instagram users sometimes follow the trends and what are the most popular hashtags in Instagram discussions/searches in order to annotate their photos. Moreover, quite frequently users give hashtags a metacommunicative use. According to Daer *et al.* [5] metacommunicative usage of hashtags can be grouped into five code categories: (a) Emphasizing, used to give emphasis or call attention, (b) Iterating, used to expresses humor, (c) Critiquing, expressing judgment or verdict, (d) Identifying, employed to refer to the author of the post, and (e) Rallying, used to bring awareness or support to a cause. Finally, it is quite common nowadays that Instagram users annotate their photos with 'noise' hashtags in order to increase the visibility their photo posts. Chua *et al.* [4] found that, approximately, 50 % of the tags Instagram users use to comment their images correspond to 'noise' (spam) hashtags while half of the 'true' labels are missing. Giannoulakis and Tsapatsoulis [8] in their effort to evaluate the descriptive value of Instagram hashtags concluded that only 30 % of hashtags can be clearly related with the visual content of photos posted in the Instagram.

Despite the previous discussion, Instagram remains a rich source of annotated photos that can serve Automatic Image Annotation(AIA) in the context of 'learning by example' paradigm [16]. AIA refers to the process of automatically attaching tags to images with the aid of concept models [11,14]. The latter represent visual objects learned from image examples and are developed with the aid low-level feature extraction and machine learning algorithms. AIA techniques calculate the correlation between image features and textual words from example datasets in order to predict keywords for unseen images [10]. The first step of AIA, is, therefore, the creation of good training examples, i.e., pairs of images and relevant tags. Since manual, i.e., performed by humans, creation of such training examples for a variety of concepts is extremely tedious and inefficient, automatic creation of training examples via crawling is an attractive alternative. Instagram contains images along with labels provided, through hashtags, by their creators/owners. Although, a large part of hashtags are not directly related with image's visual content, tagging information for photos still exist and the key is to find effective and efficient means to locate it. In this context and given that hashtag collections can be seen as a primary form of text we can apply natural language processing techniques for mining those hashtags that serve the AIA needs. Hashtag filtering is a first step towards this goal. In an analogy to document indexing, where stopwords are removed [3], in this work we suggest an innovative framework through which 'stophashtags' can be identified and filtered out. We define as 'stophashtags' usually meaningless hashtags that appear quite frequently in different, and in many cases, irrelevant hashtag categories [1].

To the best of our knowledge this is the first work that attempts to locate meaningless hashtags in Instagram for automating the process of hashtag selection for AIA. However, this problem falls under a broader category dealing with stopwords or spamwords removal and filtering. Armano *et al.* [1] in their research to locate stopwords in different document categories they propose novel metrics analyzing the informative content of each term and measuring the discriminant and characteristic capability, so in order a term to be discriminant has to distinguish in a category against the others and in order to characterize a words as a stopword has to be common all over the categories. In addition to these hashtags that promote glorification of self-harm are no longer searchable according to Instagram community guidelines [9].

Drewe [6], with the aid of Instagram API and a list of popular hashtags, created a list of unsearchable hashtags. This list, however, is unofficial, incomplete and needs regular update since it is created using ad-hoc processes and not a scientific methodology. Sedhai and Sun [13] in their research to locate spam tweets concluded that 40 % of spam tweets have three or more hashtags and it is more likely to use the word 'follow' as part of the tweet hashtags. Fan *et al.* [7] dealt with the problem of spam hashtags in Flickr and developed an algorithm to clean spam tags through cross-modal tag cleansing and junk image filtering.



elenia_di #mybeautifulgirl 🙄 #grace #littlegirl #bellezza #incanto #instapitbull #rednose #pitbullchocolate #mozzafiato #instalike #likeforlike #instapic #instalove #instamoment #instadog #instabau #mydog #mylove #thequeen #pitbullisnotacrime #caricaneabbiamo #nomtostanchimai #saturdaylife #happiness #scatti #instamoment 🐶 ❤

Fig. 1. An example of Instagram image along with its hashtags. Note the presence of meaningless hashtags such as #likeforlike, #instapic, #instalike.

Yang and Lee [17] extract descriptive keywords from web pages and they measure the relatedness between web pages and tags in order to detect spam tags in social bookmarking. Tang *et al.* [15] in their research to eliminate noise tags in a folksonomy system they propose a two-stage semantic-based method. First, they remove non-descriptive tags and then the semantic similarity between tags is examined in order to remove noise tags. Zhu *et al.* [18] in their approach for tag refinement they propose a form of convex optimization which considers, tag characteristics, error sparsity, content consistency and tag correlation.

According to a previous study [8] only 30 % of Instagram hashtags are relevant to the visual content of posted photos. The remaining 70 % can be attributed to metacommunicative use, trends mimicry and/or users' effort to fool Instagram search engine so as to attract views for their photos. Figure 1 shows an indicative example depicting hashtags belonging to the above categories. The picture depicts a pittbull along with the hashtags the creator/owner used to annotate it. Hashtags such as #mydog, #pitbullchocolate are quite descriptive regarding the visual content of this image while #pitbullisnotacrime is a hashtag, used in a metacommunicative way, expressing rallying. On the other hand, hashtags such as #likeforlike, #instapic, #instalike are meaningless 'trendy' hashtags without any actual descriptive or metacommunicative value. These hashtags can be considered as stophashtags since they appear in many, visually irrelevant, images in a similar manners as common words in irrelevant documents are considered stopwords and discarded in document indexing and classification.

2 Methodology

In order to derive concrete conclusions in our study we suggest a mathematical framework which allows us to extract the results, i.e., candidate stophashtags, combined with social science research methodology to evaluate the obtained results. Moreover, we propose a novel and innovative Algorithm for calculating hashtag score and locating stophashtags.

As already mentioned we consider as stophashtags hashtags that are meaningless and appear in different images retrieved by different and generally non-related hashtag categories. We select N subjects / hashtags that have no obvious thematic relation (e.g. the subjects 'food' and 'gadgets') and retrieve all images related with these subjects. We argue that common hashtags that appear in the retrieved images have no obvious relation with images' visual content; thus, they would not have descriptive value. The proposed methodology is analytically presented in the form of pseudocode in Algorithm 3. It involves six basic steps: (1) Create a list N of independent hashtags, (2) for each hashtag H collect relevant images, (3) for each image I find all hashtags, (4) for each hahstag H compute the stophashtag score S_H, (5) compute threshold T of S_H scores with the aid of Otsu method, (6) if $S_H > T$ add to H Stophashtag List.

2.1 The Stophashtag Score

Let us denote with N_H the number of subjects/hashtags in whose retrieved photos we locate a specific hashtag H. A total of N, randomly selected for investigation, subjects is selected for this purpose. As already explained it is better these subjects to be unrelated and independent each other. We also denote with I_H the number of distinct photos we locate the aforementioned hashtag H and I the total number of photos retrieved through these N subjects. In order to formally estimate the likelihood of a hashtag to be a stophashtag we define a score, S_H, that combines the normalized subject frequency of a hashtag, S_F, and the normalized image frequency of a hashtag, I_F, as follows:

$$S_F = \frac{N_H - 1}{N - 1} \tag{1}$$

Algorithm 1. Locating Stophashtags

```
procedure HASHTAGS($H_{List}$)
/* $H_{List} = \{H_1, H_2, ..., H_N\}$: $N$ independent hashtags given by the user */
    $H \leftarrow []$                                                    ▷ $H$: an empty hash table
    for $H_i$ in $H_{List}$ do:
        $H[H_i] \leftarrow$ List of URLs of Instagram images tagged with $H_i$
        for $I_j$ in $H[H_i]$ do:
            $H[H_i][I_j] \leftarrow$ list of hashtags of image $I_j$
    return $H$

procedure HASHTAGLISTS($H, H_{List}$)
/* $H$: hash table obtained by procedure Hashtags */
/* $H_{List}$: List of $N$ independent hashtags given by the user */
/* $h[s] \subseteq H_{List}$: List of hashtag topics in whose images hashtag $s$ appears in */
/* $p[s]$: List of images, obtained by $H_{List}$ topics, that have $s$ as a hashtag in */
    $S \leftarrow$ set of distinct hashtags in hash table $H$
    $P \leftarrow$ set of distinct image URLs in hash table $H$
    for $s$ in $S$ do:
        $h[s] \leftarrow \{\}, p[s] \leftarrow \{\}$
        for $H_i$ in $H_{List}$ do:
            for $I_j$ in $H[H_i]$ do:
                if $s \in H[H_i][I_j]$ then
                    if $I_j \notin p[s]$ then
                        $p[s] \leftarrow p[s] + \{I_j\}$
                    if $H_i \notin h[s]$ then
                        $h[s] \leftarrow h[s] + \{H_i\}$
    return $h, p, S, P$

procedure HASHTAGSCORE($h, p, S, P, H_{List}$)
/* $H_{List}$: List of $N$ independent hashtags given by the user */
/*$S$: set of distinct hashtags in hash table $H$ */
/* $h$: Hash table showing for each hashtag the list of hashtag topics it appears in*/
/* $p$: Hash table showing for each hashtag the list of images it appears in*/
    $N \leftarrow$ length of $H_{List}$                        ▷ Total number of topics / subjects
    $I \leftarrow$ length of $P$                     ▷ Total number of distinct image URLs retrieved
    $S_H \leftarrow []; S_F \leftarrow []; I_F \leftarrow []; a_N \leftarrow 0.8$          ▷ $a_N$: a weighting factor indicated in Eq. 3
    for $s$ in $S$ do:
        $N_H \leftarrow length(h[s]); I_H \leftarrow length(p[s])$
        $S_F[s] \leftarrow (N_H - 1)/(N - 1)$
        if $N_H == 1$ then
            $I_F[s] \leftarrow 0$
        else
            $I_F[s] \leftarrow I_H/I$
        $S_H[s] \leftarrow a_N \cdot S_F[s] + (1 - a_N) \cdot I_F[s]$
    return $S_H, S_F, I_F$

procedure STOPHASHTAGLIST($S_H, S$)
/*$S_H$: list of hashtag scores */
/*$S$: set of distinct hashtags in hash table $H$ */
    $SHL \leftarrow \{\}$                                               ▷ Empty stophashtag list
    $T \leftarrow$ Otsu threshold of $S_H$ list values
    for $s$ in $S$ do:
        if $S_H > T$ then
            $SHL \leftarrow SHL + \{s\}$
    return $SHL$
```

where $N > 2$. We use $N_H - 1$ in the nominator of Eq. 1 to indicate that the hashtag used to retrieve the photos within a subject (lets call it retrieval keyword) is, by definition, relevant to the content of photos retrieved within this particular subject and cannot be considered as stophashtag. For instance assume that one of the N subjects/hashtags used to collect photos is the hashtag #car. In order to estimate the normalized subject score for the hashtag #car itself we should count its frequency of appearance in the remaining subjects excluding the one it was used as a retrieval keyword. In this respect the number of independent subjects (used in the denominator of Eq. 1) is $N - 1$.

$$I_F = \begin{cases} 0, N_H = 1 \\ \\ \frac{I_H}{I}, N_H > 1 \end{cases} \tag{2}$$

$$S_H = a_N \cdot S_F + (1 - a_N) \cdot I_F \tag{3}$$

where $0 < a_N < 1$ is a weighting factor indicating the relative importance of normalized subject and image frequencies. Recommended values, found through experimentation, for a_N lie in the interval [0.75 1).

2.2 Data Collection

In order to apply our methodology first we had to define the N subjects/hashtags we would examine to locate the stophashtags (in our case 30). Then for each subject image URLs appeared in the results page were automatically collected using the Beautiful Soup[3] library of Python. Results page includes up to 35 images; from those we chose, manually, only those images that according to our interpretation, are more related to the hashtag, i.e., retrieval keyword, we used for retrieving the photos. We then collected automatically all hashtags that accompany the selected photos for each subject. The photos and the related hashtags were gathered between 20–25 March 2016.

3 Experimental Results and Evaluation

The stophashtag list we got applying the proposed methodology is shown in Table 1. In order to define which hashtags are stophahstags we used the Otsu method [12] to calculate the threshold T that would be applied to stophashtag score S_H. It appeared that a value of T around 0.20 provides the best classification of hashtags into descriptive ($S_H \leqslant 0.20$) and stophahstags ($S_H > 0.20$). We see in Table 1 that 45 hashtags were identified as stophashtags. However, human evaluation was required to confirm which of them are not related with the visual content of images to which they appear as hashtags.

Human judgement was designed as follows: Users were asked to choose among hashtags associated with a photo the ones that describe the visual content of the

[3] http://www.crummy.com/software/BeautifulSoup/bs4/doc/.

Table 1. Stophashtag list identified using the proposed methodology (with * are denoted stophashtags that confirmed through human evaluation - see Table 2)

Hashtag	Score(S_H)	Hashtag	Score(S_H)	Hashtag	Score(S_H)
love*	0.53	like4like*	0.49	likeforlike*	0.49
l4l*	0.49	likes4likes*	0.49	likesforlikes*	0.49
Like4Like*	0.49	L4L*	0.49	l4like*	0.49
instagood*	0.46	beautiful*	0.45	photooftheday	0.42
follow4follow*	0.37	f4f*	0.37	followforfollow*	0.37
picoftheday*	0.35	follow*	0.34	photo	0.34
instalike*	0.32	instalikes*	0.32	igers	0.32
instagramers*	0.32	instagram*	0.29	amazing	0.29
style	0.26	followme*	0.26	travel	0.25
vscocam*	0.24	vsco*	0.24	vscophile	0.24
vscogood	0.24	vscogram	0.24	vsco_hub	0.24
nature	0.24	sun	0.24	spring	0.24
instamood*	0.24	fun	0.24	cute	0.22
food	0.22	handmade	0.22	girl	0.21
photography	0.21	like*	0.21	likes*	0.21

photo in question. From the retrieved, through the N subjects, photos 30 were randomly selected and included in a online questionnaire which was delivered to users through SurveyMokey[4]. Participants had to select among eight hashtags, including both descriptive hashtags and stophashtags but all of them assigned to the photo by its owner/creator, for each one of the questionnaire photos.

The 30 photos were put into two separate questionnaires, containing 15 pictures each, in order to reduce the fill in time and avoid fatigue effects. Among the eight choices given for each photo of 2–3 correspond to descriptive hashtags and the rest to stophashtags. In any case, participants were not aware that any of the given choices were considered either descriptive or stophashtags; thus, they were free to select as many of them as they wished according to their interpretation of the shown photo. Choices not selected by participants are likely to correspond to stophashtags since participants consider them as not descriptive for the photo presented to them. In that context we can use effectiveness measures such as Precision and Recall to identify which of the identified by the algorithm stophashtags were also -indirectly- classified as such by humans and vice versa. For this purpose, however, we needed to create the list of stophashtags according to human judgement.

Let us denote H_T the times that a hashtag was selected by the users and H_S the times that users had the possibility to choose it as descriptive for a picture's visual content for the 30 pictures presented to the users. In order a descriptive score D_H for each hashtag H score we can use the following formula:

[4] http://www.surveymonkey.com/.

Table 2. Stophashtags identified through human judgement

Hashtag	Recall	Hashtag	Recall	Hashtag	Recall
cunard	0.00	detailersofinstagram	0.00	flex	0.00
instacool	0.00	iphoneonly	0.00	likealways	0.00
lotr	0.00	motorsport	0.00	police	0.00
phonephotography	0.00	recentforrecent	0.00	tagstagram	0.00
teamfollowback	0.00	tv_transport	0.00	tweegram	0.00
vsco	0.00	webstagram	0.00	igdaily	0.02
F4f (Follow4follow)	0.02	vscocam	0.02	follow	0.03
followme	0.03	artesanal	0.04	corn	0.04
likers	0.04	paint	0.04	string	0.04
insane	0.04	sick	0.04	followforfollow	0.05
instagood	0.05	instadaily	0.05	instafollow	0.06
australiagram	0.06	igersaustria	0.06	instagramers	0.06
scratchmap	0.06	tagsforfollow	0.06	tv_sea	0.06
L4l (Like4like(s))	0.08	instagram	0.07	bestoftheday	0.08
instalike(s)	0.08	crocodile	0.08	facebook	0.08
followers	0.08	tagsforlike(s)	0.08	instamood	0.11
corncobpipe	0.12	frenchieoftheday	0.12	gourmet	0.12
landscape	0.12	instaphotos	0.13	lionporn	0.13
pic(ture)oftheday	0.15	instapic(ture)	0.16	instacolourful	0.16
love	0.17				

$$D_H = \frac{H_T}{H_S} \tag{4}$$

For instance consider that a hashtag was selected two times and that hashtag was given to the users as a choice only for one image. In case this photo was accessed 25 times, that is appeared in 25 filled questionnaires, then D_H equals 2/25. Based on D_H and by using a threshold T_D computed with aid of Otsu method [12] we classified the hashtags to descriptive ($D_H > T_D$) and stophashtags ($D_H \leqslant T_D$) according to human judgement. T_D was found to be 0.17.

Table 2 shows the stophashtags identified through human judgement with the procedure explained above. By comparing Tables 1 and 2 we can see that 26 hashtags appear in both tables while the 17 hashtags with the highest stophashtag score S_H, shown in Table 1, were -indirectly- verified by human judgement. Thus, we have a clear indication about the effectiveness of the proposed method especially as fas as the definition of stophashtag score (see Eq. 3) is concerned. On the other hand, there are some hashtags identified by the proposed algorithm, such as #picoftheday, #vscogood, #vsco_hub, etc., that were not confirmed by human judgement although it is clear that they lack descriptive power.

For a typical information retrieval based evaluation we consider the list of hashtags in Table 2 as the benchmark set and compute the recall R, precision P and F-scores. This gives us $R = 26/58 = 0.448$, $P = 26/45 = 0.578$ and $F = 0.505$. From the above discussion we conclude that the thresholds obtained through the Otsu method are not optimal in terms of precision and F-score. For instance a value of T equal to 0.30 would lead to $R = 20/58 = 0.345$, $P = 20/22 = 0.909$ and $F = 0.521$.

4 Conclusion

In this study we proposed a methodology for filtering Instagram hashtags as per their descriptive value for the photos they accompany. We defined a stophashtag score which proved to be very effective in modeling the likelihood for an Instagram hashtag to be a stophashtag. In an attempt to evaluate the effectiveness of the proposed method 30 subjects/hashtags were chosen and one photo containing descriptive hashtags and stophashtags was selected from each subject. Then eight hashtags, 2–3 descriptive and the rest stophashtags, were chosen among the hashtags used by the image owner. Two online questionnaires containing 15 images each were distributed to evaluators so they can choose the best suitable hashtag for every image according to their interpretation. Our hypothesis was that evaluators would not select stophashtags as descriptive to the questionnaire photos. From the results and the evaluation process we find that 26 out of 45 as stophashtags identified by the proposed algorithm coincide with -indirect-human judgement. Three (sun, girl, food) out of the 45 were erroneously considered as stophashtags while for 16 out of 45 we were are unable to conclude because they were not evaluated by the users as options in the questionnaires.

Overall, it appears that thresholding the stophashtag score for identifying the list of hashtags was not so effective at least when compared to human judgement. This is caused partially by the fact that Instagram users tend to consider as descriptive many hashtags that clearly lack descriptive power; thus, benchmarking the proposed method against indirect human evaluation, as we did in this study, may not be the most appropriate way of evaluation. In the near future we will try evaluate the hashtags for all retrieved photos and not only a randomly selected subset of them. The scalability of the proposed method in more subjects ($N >> 30$) will be also investigated.

References

1. Armano, G., Fanni, F., Giulian, A.: Stopwords identification by means of characteristic and discriminant analysis. In: Proceedings of the 7th International Conference on Agents Artificial Intelligence (ICAART 2015), pp. 353–360. Lisbon, Portugal (2015). doi:10.5220/0005194303530360
2. Baranovic, M.: What #hashtags mean to mobile photography (2013). http://connect.dpreview.com/post/1256293279/hastag-photography
3. Bramer, M.: Principles of Data Mining. Springer, London (2007)

4. Chua, T.-S., Tang, J., Hong, R., Li, H., Luo, Z., Zheng, Y.: Nus-wide: a real-world web image database from National University of Singapore. In: Proceedings of ACM Conference on Image and Video Retrieval, pp. 368–375. Santorini, Greece (2009). doi:10.1145/1646396.1646452
5. Daer, A.R., Hoffman, R., Goodman, S.: Rhetorical functions of hashtag forms across social media applications. Commun. Des. Q. Rev. **3**(1), 12–16 (2014). doi:10.1145/2721882.2721884
6. Drewe, N.: The Hilarious List of Hashtags Instagram Wont Let You Search. http://thedatapack.com/banned-instagram-hashtags-update/#more-171
7. Fan, J., Shen, Y., Zhou, N., Gao, Y.: Harvesting large-scale weakly-tagged image databases from the web. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2010), pp. 802–809. San Francisco, United States, (2010). doi:10.1109/CVPR.2010.5540135
8. Giannoulakis, S., Tsapatsoulis, N.: Instagram hashtags as image annotation metadata. In: Proceedings of the 11th International Conference on Artificial Intelligence Applications and Innovations (AIAI'15), pp. 206-220. Bayonne, France (2015). doi:10.1007/978-3-319-23868-5_15
9. Instagram's New Guidelines Against Self-Harm Images & Accounts. In: Instagram Inc (2016). http://blog.instagram.com/post/21454597658/instagrams-new-guidelines-against-self-harm
10. Jin, R., Chai, J.Y., Si, L.: Effective automatic image annotation via a coherent language model and active learning. In: Proceedings of the 12th ACM International Conference on Multimedia (ACM Multimedia 2004), pp. 892–899. New York, United States (2004). doi:10.1145/1027527.1027732
11. Jin, C., Jin, S.-W.: Automatic image annotation using feature selection based on improving quantum particle swarm optimization. Sig. Process. **109**, 172–181 (2015). doi:10.1016/j.sigpro.2014.10.031
12. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (1979). doi:10.1109/TSMC.1979.4310076
13. Sedhai, S., Sun, A.: HSpam14: a collection of 14 million tweets for hashtag-oriented spam research. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 223–232. Santiago, Chile (2015). doi:10.1145/2766462.2767701
14. Snoek, C.G., Worring, M.: Concept-based video retrieval. Found. Trends Inf. Retrieval **2**(4), 215–322 (2008). doi:10.1561/1500000014
15. Tang, R., Zuo, J., Xu, K., Zheng, J., Wang, Y.: An intelligent semantic-based tag cleaner for folksonomies. In: Proceedings 2010 International Conference on Intelligent Computing and Integrated Systems (ICISS2010), pp. 773–776. Guilin, China (2010). doi:10.1109/ICISS.2010.5657118
16. Theodosiou, Z., Tsapatsoulis, N.: Crowdsourcing annotation: modelling keywords using low level features. In: Proceedings of the 5th International Conference on Internet Multimedia Systems Architecture and Application (IEEE IMSAA 2011), pp. 1–4. Bangalore, India (2011). doi:10.1109/IMSAA.2011.6156351
17. Yang, H.-C., Lee, C.H.: Identifying spam tags by mining tag semantics. In: Proceedings 3rd International Conference on Data Mining and Intelligent Information Technology Applications (ICMiA), pp. 263–268. Macao, China (2011)
18. Zhu, G., Yan, S., Ma, Y.: Image tag refinement towards low-rank, content-tag prior and error sparsity. In: 18th ACM international conference on Multimedia (MM 2010), pp. 461–470. Firenze, Italy (2010). doi:10.1145/1873951.1874028

Big Data and the Virtuous Circle of Railway Digitization

Howard J. Parkinson[1(✉)] and Gary J. Bamford[2(✉)]

[1] Digital Rail Limited, Lancaster, UK
hjparkinson@railsystems.co.uk
[2] NTTX Advisory Limited, Congleton, UK
gary.bamford@nttx.co.uk

Abstract. Presently there are many disruptive technologies and philosophies that will change the way we learn from accidents and carry out safety management on the railway. These ideas and technologies include Big Data, Safety II, the Digital Railway, and Simulation. These have all come to the fore in the last 5 years or so to produce a potential virtuous circle that will provide significant synergy between them. This offers all railway stakeholders emergent opportunities including greater safety, better system reliability and efficiency.

This paper is concerned with railway safety and how it has improved over many years, yet requires new thinking to get further gains. A Big Data Analytics approach is described using the ELBowTie tool and the type of data available. Analytics are discussed and how a digital railway will provide a backbone for this based upon an Internet of Things architecture. Simulation and the use of digital information is discussed and how this can close the virtuous circle.

Keywords: Simulation · Railway safety · Accidents learning · Big Data · Digital rail

1 Introduction

Railways safety has improved over the years often as a result of learning hard lessons from railway catastrophes. The modern approach is to try to envisage the accident before it happens and put in place mitigations. We are entering an age now where we have complex socio-technological systems that rely upon computer control and human-machine interfaces. This is particularly true in the railway industry which is moving towards more sophisticated automatic train protection (ATP). This could mean that the human workers who are still involved will find that a full understanding of the systems and operations is almost an intractable problem (Hollnagel 2015).

The situation is not all bad news though as there are other technologies and philosophies emerging that could, when taken together actually help take the railway to a new low level in accident and incident occurrence. These opportunities are described in this paper as a digital virtuous circle, which is illustrated in Fig. 1.

The key elements are: a new way of considering safety termed Safety II (Hollnagel 2015), the advent of Big Data (BD) and the Internet of Things (IoT) (Parkinson and Bamford 2016), the Digital Railway initiative (Network Rail 2015), and the potential

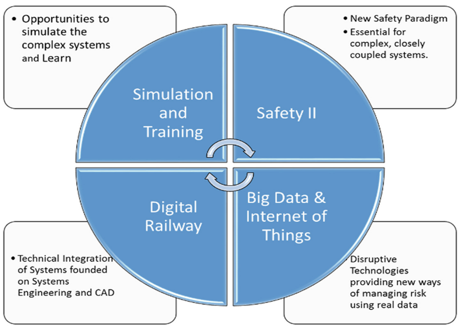

• Opportunities to simulate the complex systems and Learn
• New Safety Paradigm
• Essential for complex, closely coupled systems.
Simulation and Training
Safety II
Digital Railway
Big Data & Internet of Things
• Technical Integration of Systems founded on Systems Engineering and CAD
Disruptive Technologies providing new ways of managing risk using real data

Fig. 1. The digital virtuous circle

innovation of simulation, virtual and augmented reality and how they might fit together.

The next section contains a brief discussion of the philosophy behind railway safety and why we may need a new approach. Section 3 contains a discussion of Safety II, and how the existing techniques are no longer adequate. Section 4 provides an overview of the disruptive technologies that are around at present affecting our approach to safety.

Section 5 introduces an approach to assessing risk, building in the potential of Big Data. Section 6 describes how the digital rail initiative coupled with Systems Engineering (SE) and Computer Aided Design (CAD) data would provide a foundation for the Big Data approach. Section 7 describes how simulation using all the aforementioned techniques might close the digital virtuous circle and Sect. 8 reviews how this all can support the intent of Safety II.

2 Railway Safety

The unique characteristics of railway safety are that there is a vehicle with a large mass travelling at high velocity with low friction, steel wheel to steel rail contact, making for very long braking distances that are often longer than the driver's field of vision. Also, the train travels on guiding track which allows no opportunity to avoid collision by steering action. High collision speeds and high kinetic energy makes for high severity consequences. To make risks acceptable, therefore, it is necessary to reduce the probability of train accidents to a vanishingly small number.

Railway tragedies over many years have led to numerous safety improvements which include (Erdos 2004):

- Braking Systems: Vacuum brakes, compressor based braking systems, the Westinghouse Brake system,
- Communication Systems: Electric telegraph, telephone and radio systems,
- Vehicle Design: Crashworthiness avoiding telescoping,
- Signalling: Absolute Block Sections, Multiple-Aspect Signals, Track Circuits, Interlocking, Automated level crossings, ATP,

- Platform Interface: Screen doors, Door to traction interlocking

Figure 2 depicts the downward trend in fatal railway train accidents. A plateau seems to have been reached with technical failures and individual worker errors being reduced to very low levels. For example, no track worker fatalities occurred last year in the UK (RSSB 2015). The largest number of deaths result from suicides and trespassers, which swamp the all other causes.

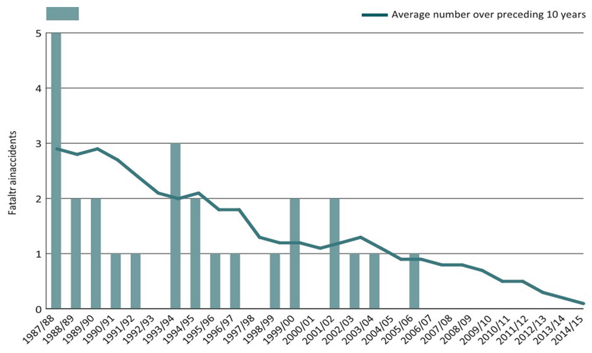


Fig. 2. Train accidents with passenger or workforce fatalities (RSSB 2015)

There is, of course, still the danger of large unexpected railway accidents such as the recent ones in Bavaria (BBC 2016) and Santiago de Compostela in Spain (RSSB 2015) together resulting in over one hundred deaths. These are attributable to organisational type accident factors and are always complex in nature and usually the result of human error (Hollnagel 2015). The railway appears to be catching people out with more and more complex systems and rules.

So, in the railway as in other industries, there are tightly coupled technologies with non-linear functioning, producing ‘outlier’ events that usually result in significant impacts which human supervisors find difficult to control in the normal and degraded operation. The next section contains a discussion of these implications in a little more detail and what might be needed to control them.

3 Safety II

Hollnagel (2015) has suggested that instead of focusing upon failure we focus upon how the system actually works in the real world and not as it is imagined to work by a development team. Having an understanding of how the system interacts with humans, it can then be established how best to safely manage them. Hollnagel (2015) has described this as Safety II which would supersede the traditional Safety I approach, which would still be suitable for non-complex systems.

Figure 3 taken from Hollnagel (2015) illustrates the imbalance that occurs when the focus is on what goes wrong and not on how things go right.

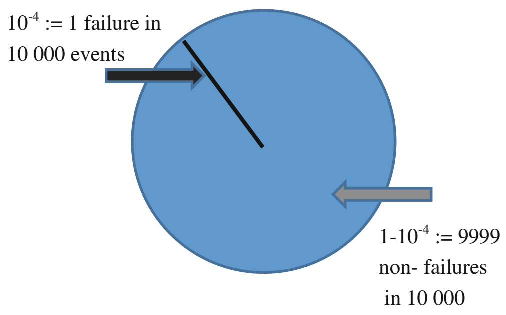

10^{-4} := 1 failure in 10 000 events
$1\text{-}10^{-4}$:= 9999 non- failures in 10 000

Fig. 3. The imbalance between things that go right and things that go wrong (from Hollnagel 2015).

Safety I has its focus on analysing failure and has evolved from the analysis of more simple electro/mechanical system though 3 ages as describe by (Hale and Hovden 1998), i.e. the age of technology, the age of human factors and the age of safety management. Many of the techniques still used are founded in simpler times such as, failure mode and effect analysis (FMEA) and fault tree analysis (FTA) amongst others. A thorough understanding by the practitioner of past incidents is essential in all these processes. Without this knowledge it will be impossible to understand errors, failures, and how hazards propagate into accidents.

As technical failures have been reduced, and as the complexity and difficulty in understanding complex systems has increased, more and more accidents are being attributed to human error. It is becoming increasingly difficult to train workers to respond to the various hazardous states that these complex systems can find themselves in. So, new approaches are needed to understand, analyse and optimise day-to-day safety management. Modern systems are very reliable and focussing upon failure is the wrong way to analyse these systems in order to achieve the next step change in safety.

As stated earlier new techniques, data and initiatives are available that will enable a new focus upon how systems actually work. In addition, technologies to simulate environments exist to help increase the ability of the human worker to deal with system perturbation and increase the resilience of the safety system. The next section contains an overview of some of these new technologies.

4 Disruptive Technologies

It is a widely held belief that we are entering a revolution in technology and computer intelligence, driven primarily by increases in computing power and the development of intelligent algorithms. This is being described as the 3rd Industrial Revolution (Ford 2015). There are several so-called disruptive technologies present which include a Big Data (BD), the Internet of Things (IoT), Virtual Reality (VR), Augmented Reality (AR), and Machine Learning (ML) to name a few.

This interaction of disruptive technologies can be viewed as a key part of the virtuous circle. The pace at which ML is happening is now hard to deny (Parkinson, Bamford and Kandola 2016). Many of these technologies have promised great things in the past and not delivered. For example, neural networks were very much the flavour in the 1990 s, however, due to the lack of computational power and the lack of understanding of the how to combine them with other approaches, they did not deliver the required accuracy for safety related predictions (Iwnicki and Parkinson 1999). Gartner (2016) has identified a curve that describes this trend in the uptake of new technology as shown in Fig. 4.

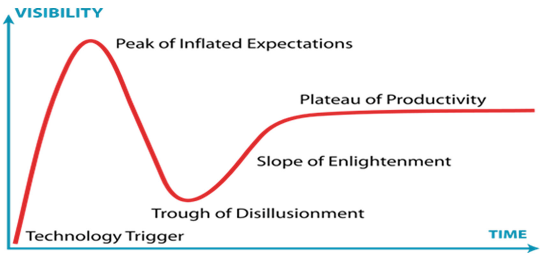


Fig. 4. The Uptake in New Technology (Gartner 2016)

Many of these disruptive technologies are now beginning to climb the slope of enlightenment and are approaching the plateau of productivity.

5 Big Data Approach

We are now in the Big Data Analytics (BDA) age where mountains of data should enable us to understand how complex socio technical systems actually work. This is a central idea of Safety II.

In addition, visualisation will be the key to providing the 'users' with a view of system dynamics. Data/information overload is a problem that will need to be overcome. Possibly by only presenting information critical to the business or the individual. This new area of critical visualisation is being called the Internet of Me - what matters for you view of the world (IET Turing lecture 2016).

Spending in railway systems in the next twenty years is set to explode, and it is critical that best practices in safety, data analysis and systems engineering are employ. (Parkinson and Bamford) have defined a BDA approach to learning from accidents which will enable the identification of heightened risk. The approach uses an accident causation model called an ELBowTie, illustrated in Fig. 5. This model is based upon the bowtie methodology that is already well utilised in the safety engineering world (Yaneira 2013).

(Parkinson and Bamford) first establish a railway Enterprise Data Taxonomy (EDT) which lists the available sources of data that can be linked to railway operational risks. These include, for example: condition based monitoring information from sensors, either analogue or digital, that would provide digital information, including vibration (accelerometers), machine vision, heat, displacement, strain, humidity, particle ingress, etc. which would be classified as 'Real Time, remote monitoring', which is already an

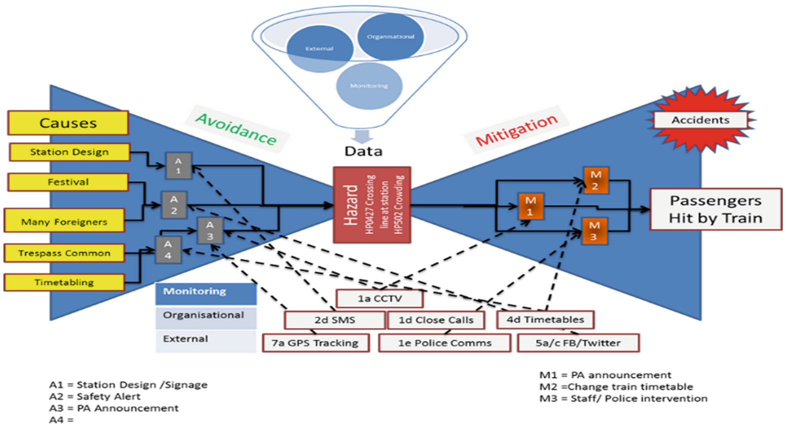


Fig. 5. ELBowTie Big Sat Railway Risk Assessment Tool (Parkinson and Bamford 2016)

accepted means of classifying this type of data. Other data types are less well defined, for example, data from industry reports, staff morale, organisational culture, but can be equally as important in safety evaluations.

The EDT is aimed at providing a mechanism for systematically classifying data related to safety incidents to facilitate assessment of data sources in a traceable and consistent manner, giving the new approach its name: “Enterprise data taxonomy Linked Bow-Tie” (ELBowTie),

The research by (Parkinson and Bamford) looked a series of railway accidents and established their potential hazardous causes and conditions. These causes and conditions were then linked to the EDT. The EDT is then linked to the bowtie elements to predict heightened railway risk. It will be necessary to develop a bowtie for every railway hazard to develop the full risk picture.

At the heart of the tool are a series of Machine Learning algorithms that will be trained to analyse a stream of data and recognise when the system is outside of its normal bounds. These outputs are then amalgamated to take into account the complex interaction of the typical accident scenario (Parkinson, Bamford and Kandola 2016). The data is linked to the barriers on either side of the bowtie to enable real time interventions to take place either to prevent hazards occurring or to prevent hazards propagating into accidents. When a red flag situation transpires, the particular initiators are known and thus enabling automated warnings to be activated. Figure 6 depicts this

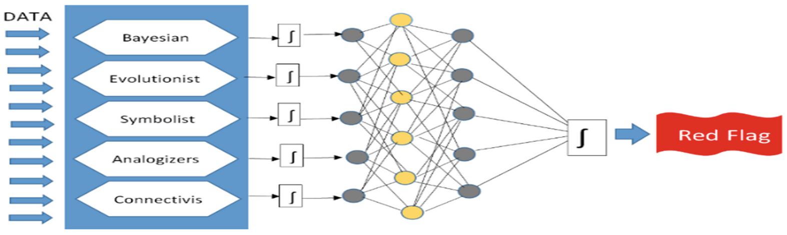


Fig. 6. Proposed Structure for Combining Data (Parkinson, Bamford and Kandola 2016)

relationship. The paper by Parkinson, Bamford and Kandola (2016) also has an extensive discussion of the various machine learning approaches that might be taken and how these could be amalgamated.

6 The Digital Railway

The Digital Railway is an initiate in the UK mainly driven by Network Rail (Network Rail Ltd 2015) and is depicted in Fig. 7. Network Rail are the organisation responsible for maintaining the mainline railway infrastructure in Great Britain. Several other organisations called TOCs (Train Operating Companies) are responsible for running trains and there are a myriad of private maintenance contractors and engineering support companies.

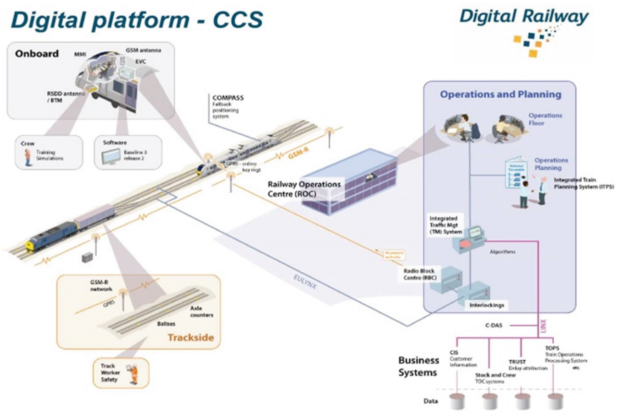


Fig. 7. Digital Rail Architecture (Network Rail 2015)

The core element of the Digital Railway will be the ERTMS (European Rail Traffic Management System) which, simplistically, comprises an on-board computer that provides automatic train protection by calculating a safe train movement authority based upon track conditions and the status of the railway. The interlocking sends signals to the train, relating whether a route is set and whether other trains are present. Around this core system all the communication, passenger information, status, etc. will be provided.

The Digital Railway will provide a platform for an IoT approach with enhanced communication systems and inbuilt sensors in assets.

7 Simulation and Training

Finally, we have moved into areas of simulation and training. With the aid of the digital models created during system development, and using AR and VR, we can create scenarios for workers to investigate how systems really function. This allows managers and designers to communicate with the various stakeholders about how the system works in all its complexity and not simply how they "think" it will work. This will enable effective training to be designed.

To build upon the ability to simulate systems, the recent training development of gamification is useful. Research suggest this is the optimum approach (Kapp 2012) to training were trainees are stimulated to learn through competition, collaboration and reward. E-learning has struggled to deliver on its promise due to lack of good training design and creativity. However, the latest simulation technology and gamification opportunities for blended delivery will improve the situation. Blended delivery is the combination of face to face delivery with e-learning and online training interventions.

Training and simulation enables the understanding of safety concepts and the working principles to be properly laid out. The training coarse attendees discover the skills and knowledge by playing the game and come away more motivate and better able to apply the principles.

8 Closing the Virtuous Circle

A virtuous circle has been described that could take railways to a lower level of safety risk. Disruptive technologies could be threats to the way the railway is operated and its continual viability but if the chances for efficiencies and improvements in safety are taken then the railway's prospects are improved. Machine learning and supporting technologies (Ford) are coming together now that will make the way we undertake much white collar work radically different in 5 years' time and much potentially redundant in 10 years' time. This is likely to apply to railway risk and safety management activities too.

How does this affect the future of safety management? It will mean the replacement of risk assessment with real data. The sacred cow of risk assessment and quantification no longer works for socio technological systems as the world of railways is becoming too complex. Quantification is ineffective as it seeks to model the world using a gross simplification by deriving numbers that imply a large level of accuracy. Approaches like the (RSSB) safety risk model, which uses historical data to determine quantified risk figures for hazards on the railway, would be made redundant as the approach described in the ELBowTie would use real data in real time.

A data driven approach will replace "conventional safety engineering". If one looks through the railway safety and reliability standards, CENELEC EN50126 and the recommended activity at each life-cycle stage it can be seen that nearly all the items are data driven. By using Big Data and Machine Learning the majority of the work undertaken now by safety engineers could be eliminated.

9 Conclusion

The railway industry has advanced a long way with safety methods over the past fifty years and accidents are at a record low, but now there must be a major change if further improvements are to be realised. The approach of blaming 80 % of our accidents on human error (Hollnagel 2015) is clearly not right. Safety II, Big Data, the Digital Railway, and Simulation have all come along in the last 5 years or so to produce a virtuous circle that will provide massive synergies. The complexity of modern socio technological systems requires that a new approach is required and the elements discussed in this paper provide a way forward. This will offer railway stakeholders striking opportunities including greater safety, happier workers, better reliability, greater efficiency and less waste.

References

BBC Report Bad Aibling Rail Disaster (2016). http://www.bbc.co.uk/news/world-europe-35530538. Accessed 23 Mar 2016

CENELEC EN50126

Erdos, G.D.: The Evolution of Rail Safety IRSE Technical Meeting: Adelaide 29 October 2004 (2004)

Ford, M.: The Rise of the Robots. Oneworld, London (2015)

Gartner (2016). http://www.gartner.com/technology/research/methodologies/hype-cycle.jsp. Accessed 23 Mar 2016

Hale A, Hoyden J.: Management and culture: the third age of safety (1998)

Hollnagel, E.: Safety I and Safety II. Ashgate, Farnham (2015)

IET and BCS Turing Lecture 2016: The Internet of me: It's all about my screens. http://conferences.theiet.org/turing/. Accessed 29 Mar 2016

Iwnicki, S., Parkinson, H.J.: Fault Free Infrastructure: Effective Solutions to Improve Efficiency, Derby, England, 23–24 November 1999. Hardcover Published Nov-1999 ISBN 1860582338, "Assessing railway vehicle derailment potential using Neural Networks". International IMechE Conference

Kapp, K.M.: The Gamification of Learning and Instruction. Pfieffer, Wiley, Sanfrancisco (2012)

Network Rail 2015, Digital Railway Program. http://www.networkrail.co.uk/Digital-Railway-Introduction.pdf. Accessed 26 Feb 2016

Parkinson, H.J., Bamford, G.: The potential for using big data analytics to predict safety risks by analyzing rail accidents. In: Accepted for 3rd International Conference on Railway Technology: Research, Development and Maintenance, Cagliari, Sardinia, Italy, 5–8 April 2016

Parkinson, H.J., Bamford, G., Kandola, B.: The development of an enhanced bowtie railway safety assessment tool using a big data analytics approach. In: IET Conference ICRE Brussels, May 2016

RSSB Annual Safety Performance Report 2014/15 (2015). http://www.rssb.co.uk/Library/risk-analysis-and-safety-reporting/2015-07-aspr-key-findings-2014-15.pdf. Accessed 26 Feb 2016

Yaneira, E.S., et al.: Bowtie diagrams in downstream hazard identification and risk assessment. In: 8th Global Congress on Process Safety Houston, TX, 1–4 April 2012. American Institute of Chemical Engineers (2013)

Unified Retrieval Model of Big Data

Asma Al-Drees[1(✉)], Reem Bin-Hezam[2], Ruba Al-Muwayshir[3], and Wafa' Haddoush[4]

[1] Information Systems Department, College of Computer and Information Sciences, King Khaled University, Abha, Saudi Arabia
edrees@kku.edu.sa

[2] Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia
rybinhezam@pnu.edu.sa

[3] Information Systems Department, College of Computer and Information Sciences, Aljouf University, Sakakah, Saudi Arabia
rkmuwayshir@ju.edu.sa

[4] Information Systems Department, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia
wafaahaddoush@yahoo.com

Abstract. With the huge growth of big data, effective information retrieval methods have gained research focus. This paper addresses the difficulty of retrieving relevant information for a large system that involves fusion of data. We propose a retrieval model to enhance and improve the retrieving process along with the user's metadata learning to develop and enhance a retrieval system.

Keywords: Big data · Indexing data · Storage · Retrieval model · Challenges and issues · Location-based service · Metadata

1 Introduction

Recently, there is exponential increase in the amount of information being produced in the web. Web data is broadly distributed in different formats and is prone to extremely frequent changes. Every day, millions of web pages with different formats are added or updated (around 44 million pages). This rapid and uncontrollable change to the web data is what makes the World Wide Web a distinct source of information, and therefore the retrieval techniques associated with traditional data sources do not work appropriately in the case of the web. Effective decision-making based on such huge amounts of data can be achieved only if useful knowledge is extracted automatically from them [1]. Hence, effective information retrieval methods are warranted for an efficient assimilation of such information leading to timely and productive decision-making. These retrieval models are using suitable user profiles that provide the means to categorize data. This facilitate in the reuse and organization of data for synthesizing knowledge required for recommender systems [2].

Ideally, the capable aspects of social recommender retrieval model should be adopted to provide decision support as it attempts to discover patterns, trends and correlations hidden in data to help in making effective decisions. However, traditional retrieval model techniques are not capable of combining text, image and user profile data to retrieve pertinent and relevant information for providing good recommender systems for effective and timely decision-making [4–6].

This paper addresses the overview of the big data including its structure and analysis techniques in Sect. 2. In Sect. 3 we discuss the retrieval models approaches along with the big data. Some issues and challenges of big data in developing an effective retrieval model will be explained in Sect. 4. Sect. 5 provides the proposed effective information retrieval model. This retrieval model is based on user' metadata learning by combining TPO information' (Time, Position, Occasion) from user's location by using some GPS techniques that facilitate in arriving at a reduced data set intelligently. Finally, conclusion will be presented in Sect. 6.

2 Big Data Overview

We are awash in a flood of data today every day we create about 2.5 quintillion bytes of data. This big data comes from everywhere and everyone: machines such like sensors which used to gather climate information, posts to social media sites, digital pictures and videos, purchase transaction records, and cell phone GPS signals to name a few [7, 8].

Big data refers to data sets or combinations of data sets whose size (volume), complexity (variability), and rate of growth (velocity) make them difficult to be captured, managed, processed or analysed by conventional technologies and tools, such as relational databases and desktop statistics or visualization packages, within the time necessary to make them useful [9–11]. While the size used to determine whether a particular data set is considered big data is not firmly defined and continues to change over time, and it is expected that the rate of growth of big data will continue to increase for the foreseeable future [40]. When big data is effectively and efficiently captured, processed, and analysed. It is influencing the way of doing business, and drive real business value. Big data can support value creation for organizations as following:

1. Creating transparency by making big data openly available for business and functional analysis (quality, lower costs, reduce time to market).
2. Supporting experimental analysis in individual locations that can test decisions or approaches, such as specific market programs.
3. Assisting, based on customer information, in defining market segmentation at more narrow levels.
4. Supporting Real-time analysis and decisions based on sophisticated analytics applied to data sets from customers and embedded sensors.
5. Facilitating computer-assisted innovation in products based on embedded product sensors indicating customer responses [12, 13, 28].

2.1 Big Data Characteristics

1. Volume: volume measures the amount of data available to an organization, which does not necessarily have to own all of it as long as it can access it [14–16]. As data volume increases, the value of different data records will decrease in proportion to age, type, richness, and quantity among other factors.

2. Velocity: velocity measures the speed of data creation, streaming, and aggregation that is the speed at which this data moves from endpoints into processing and storage. Ecommerce has rapidly increased the speed and richness of data used for different business transactions (for example, web-site clicks). Data velocity management is much more than a bandwidth issue; it is also an ingest issue (extract transform- load) [17–19].
3. Variety: variety is a measure of the richness of the data representation – text, images video and audio. From an analytic perspective, it is probably the biggest obstacle to effectively using large volumes of data [21, 22]. Incompatible data formats, non-aligned data structures, and inconsistent data semantics represents significant challenges that can lead to analytic sprawl [23, 24].
4. Complexity: Complexity measures the degree of interconnectedness (possibly very large) and interdependence in big data structures such that a small change (or combination of small changes) in one or a few elements can yield very large changes or a small change that ripple across or cascade through the system and substantially affect its behaviour, or no change at all [25, 27, 29].

2.2 Big Data Analysis

There are several approaches to collecting, storing, processing, and analysing big data (unstructured data). Unstructured data refers to information that either does not have a pre-defined data model or does not fit well into relational tables. Unstructured data is the fastest growing type of data. MapReduce, in conjunction with the Hadoop Distributed File System (HDFS) and HBase database, as part of the Apache Hadoop project is a modern approach to analyse unstructured data [3, 30, 31].

1. *MapReduce:* It is a framework for distributed computing and large datasets on a scale-out shared-nothing architecture to address processing large unstructured data sets. Hadoop is an open source Apache project, which was implemented on these concepts. Hadoop is both a distributed file system modeled, a distributed processing framework, using MapReduce concepts and a distributed database called HBase [33–35].
2. *Hadoop Distributed File System (HDFS):* The premise of Hadoop was to bring processing to the data in a distributed file system fashion. Therefore, a single file is split into blocks and the blocks are distributed in the Hadoop cluster nodes. The input data in HDFS is treated in write-once fashion and processed by MapReduce, and the results are written back in the HDFS [36, 37]. The data in HDFS is protected by a replication mechanism among the nodes. This provides reliability and availability despite node failures. There are two types of HDFS nodes: DataNode, which stores the datablocks of the files in HDFS; and NameNode, which contains the metadata, with enumeration of blocks of HDFS and a list of DataNodes in the cluster [38].

   – *Hadoop MapReduce:* It provides a mechanism for programmers to leverage the distributed systems for processing data sets. MapReduce can be divided into two distinct phases:

- Map Phase: Divides the workload into smaller subworkloads and assigns tasks to Mapper, which processes each unit block of data. The output of Mapper is a sorted list of (key, value) pairs. This list is passed (also called shuffling) to the next phase.
- Reduce: Analyzes and merges the input to produce the final output. The final output is written to the HDFS in the cluster [39].

2.3 Big Data Structure

The defining processing capabilities for big data structure are to meet the volume, velocity, variety, and value requirements. Unique distributed (multi-node) parallel processing architectures have been created to parse these large data sets. There are differing technology strategies for real-time and batch processing requirements. For real-time, key-value data stores, “NoSQL”, allow for high performance, index-based retrieval. For batch processing, a “Map Reduce,” filters data according to a specific data discovery strategy. After the filtered data is discovered, it can be analysed directly, loaded into other unstructured databases, sent to mobile devices, or merged into traditional data warehousing environment and correlated to structured data [41, 42] (Fig. 1).

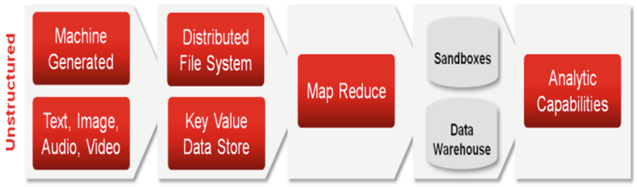


Fig. 1. Big data structure capabilities [41]

3 Retrieval Model Approaches in Big Data

Information Retrieval “IR” refers to the retrieval of unstructured and semi-structured data, especially textual documents, in response to a query or topic statement. Of course, other kinds of data can also be unstructured, e.g., photographic images, audio and video. However, the main focus of the IR research is the text-based approach. IR typically seeks to find documents in a given collection that are “about” a given topic or that satisfy a given information need [43, 44]. The topic or information need is expressed by a query, generated by the user. Documents that satisfy the given query in the judgment of the user are said to be “relevant.” Documents that are not about the given topic are said to be “non-relevant” [18, 45].

Information retrieval (IR) systems are based, either directly or indirectly, on models of the retrieval process. These retrieval models specify how representations of text documents and information needs should be compared in order to estimate the likelihood that a document will be judged relevant. The estimates of the relevance of documents to a given query are the basis for the document rankings that are now a familiar part of IR systems [46].

Generally, there are two major categories of IR technology and research: semantic and statistical. Semantic approaches attempt to implement some degree of syntactic and semantic analysis [47, 48]. In statistical approaches, the documents that are retrieved or that are highly ranked are those that match the query most closely in terms of some statistical measure. Statistical approaches fall into a number of categories: Boolean, vector space, and probabilistic. Statistical approaches break documents and queries into terms. These terms are the population that is counted and measured statistically [20].

4 Big Data Issues and Challenges in Retrieval Models

A major challenge for IT researchers and practitioners is that the growth rate of data is very fast and exceeds the ability to handle and analyse data effectively. Many issues and challenges raised regarding retrieving these data.

4.1 Issues

There are many issues raised while dealing with retrieving big data. The most fundamentals are three: storage, management, and processing issues [26]. Each one has its own technical problems and challenges that need to deal with.

1. *Storage Issues:* Previously, the quantity of data exploded when each new storage medium is introduced. On the other hand, nowadays the most recent explosion, which is due to social networks and media, has been evolved without introducing any new storage medium. Moreover, data is created by everyone and everything, not by only professionals like before.
2. *Management Issues:* Management may be the most difficult issue to deal with big data. This is because the sources of these data are varied, temporally and spatially, by format, and by method of collection. Not all these data sources have adequate metadata descriptions. Actually, there is no perfect big data management solution until know, and this raise an important research challenge.
3. *Processing Issues:* When big data is needs to be processed entirely in an effective manner, extensive parallel processing and new analytical algorithms are needed in order to provide timely and actionable information [49–51].

4.2 Challenges

Retrieving information from big data faces many challenges regarding its dynamic design and analytics. Under each challenge there are many issues and concerns [1, 26, 28].

- Design
    1. *Quality versus Quantity:* As users acquire and have access to more data (quantity), they often want even more. On the other hand, a big data user may be

concerned more on quality which means not having all the data available, but having a (very) large quantity of high quality data that can be used to figure specific and high-valued conclusions. The level of precision that the user requires is important to solve this issue.

2. *Structured versus Unstructured Data:* Translation between structured and unstructured data suitable for analytics can delay end-to-end processing performance. The emergence of non-relational, distributed, analytics oriented databases such as NoSQL, MongoDB, and SciDB provides dynamic flexibility in representing and organizing information.
3. *Data Ownership:* Data ownership presents a critical and ongoing challenge, particularly in social media. While huge amount of social media data reside on the servers of Facebook and Twitter for example, it is not really owned by them, although they may compete so because of residency. Actually, the owners of the pages or accounts believe they own the data, and this is a dichotomy which needs to be solved by law.
4. *Security:* In certain domains, there is a fear that certain organizations will know too much about individuals, as more data is accumulated about them. A key research problem is to randomize personal data among a large data set enough to ensure privacy. Perhaps the biggest threat to personal security is the unregulated accumulation of data by numerous social media companies. Clearly, some big data must be secured with respect to privacy and security laws and regulations [26, 52].

4.3 Analysis

1. *Scale:* Managing large and rapidly increasing volumes of data has been a challenging issue for many decades. A critical issue is whether or not an analytic process scales as the data set increases by orders of magnitude. Data volume is scaling faster than compute resources, and CPU speeds are static. Over the last five years the processor technology has made a dramatic shift. Also the move towards cloud computing, which now aggregates multiple disparate workloads with varying performance goals [1].
2. *Timeliness:* The opposite side of size is speed. The larger the data set to be processed, the longer it will take to analyze. The design of a system that effectively deals with size is likely also result in a system that can process a given size of data set faster. There are many situations in which the result of the analysis is required immediately. It is obviously impractical to scan the entire data set to find suitable elements. Rather than this, index structures are created in advance to permit finding-qualifying elements quickly [1, 28].
3. *Privacy:* The privacy of data is another huge concern, and one that increases in the context of Big Data. There is great public fear regarding the inappropriate use of personal data, particularly through linking of data from multiple sources. Managing privacy is effectively both a technical and a sociological problem, which must be addressed jointly from both perspectives to realize the promise of big data [54]. The wealth of individual-level information that Google, Facebook, and some mobile phone and credit card companies would jointly hold if they ever were to

group their information is in itself a concern [32]. Another privacy issue rises with big data recommendation systems that use the user location for better prediction. This affect the user privacy by tracking his location and may cause many critical problems to the user, although in some systems, the user himself is the one who publish his location on a social network site like Foursquare, Google Latitude or Facebook Places [53, 55].

5 Proposed Retrieval Model

The proposed retrieval model based on the idea of: using the user's metadata to filter the results and get more valuable responds for the user. To search the contents through the Internet, search engines should make use of some information such as current time, user's location, common occasions and so on (Metadata). We propose an information retrieval model to effectively use both user's query and database of metadata based on (Time, Position, Occasion).

In this model, query part is reducing user's overload information by choosing the relevant metadata. And a user becomes possible to retry sorting the matching results. Then, in the matching part, the search engine compares user's metadata with database' metadata to judge whether both of metadata are matching or not along with the user's query. The quantity of matching functions should be sufficient to compare any user's metadata and database's metadata. Matching functions compare values of user's metadata and database's metadata. When a lot of metadata are stored into user's storage, a mechanism automatically selecting metadata is desired. These metadata can be gathered by using some techniques related to the GPS (Global Position System) to determine the exact location of the user. After that, the results are ranked according to some scoring methods; static or dynamic scoring functions. Finally, the respond is sent to the user (Fig. 2).

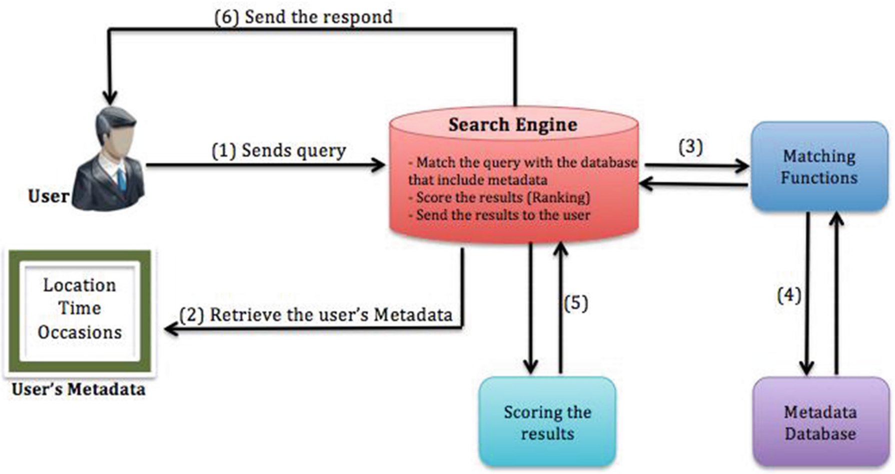


Fig. 2. Information retrieval Model based on the user's metadata

This retrieving model solves a lot of the traditional retrieving models' issues; it reduces the storage size and allows the scaling part to be done more effectively and efficiently. Also, it increases the speed of responding the results to the user.

6 Conclusion

This paper discussed the importance of demonstrating efficient retrieving model in the huge big data with proposed an effective retrieval model.

The system framework was implemented in automating the process of a physical search in the web. As the huge data comprises of many formats it has become mandatory to perform intelligent automation and to devise the methods of retrieving the information, which is relevant, and fast with respect to a user's metadata information.

The proposed model shows the process of providing the more relevant results for the user, considering user's metadata along with the user's query information. It contains capability of narrowing to the user's needs and preferences and based on them considering user current place.

The real-life implementation has demonstrated that analysing a user-query along with his/her metadata makes the retrieval system highly effective.

References

1. Agrawal, D., Bernstein, P., Davidson, S.: Challenges and Opportunities with Big Data. A community white paper developed by leading researchers across the United States, p. 17 (2011)
2. Arai, A., Fujikawa, K., Sunahara, H.: A proposal of information retrieval method based on TPO metadata (2009)
3. Bakshi, K.: Considerations for Big Data: Architecture and Approach (2012)
4. Begoli, E., Horey, J.: Design principles for effective knowledge discovery from big data. In: Joint Working Conference on Software Architecture & 6th European Conference on Software Architecture, p. 4 (2012)
5. Big Data for Development: Challenges & Opportunities. Global Pluse, 47 (2012)
6. Big Data Survey. Giga Spaces, 5 (2011)
7. Big data: The next frontier for innovation, competition, and productivity. McKinsey Global Institute, 156 (2011)
8. Bindra, A., Ashish Bindra, S., Ashish Bindra, K.: Distributed big advertiser data mining. In: IEEE 12th International Conference on Data Mining Workshops, p. 1 (2012)
9. Borkar, V., Carey, M.J., Li, C.: Inside "Big Data Management": ogres, onions, or parfaits? In: EDBT/ICDT 2012 Joint Conference, Berlin, Germany, p. 12 (2012)
10. Cavoukian, A.: Privacy, security, big data–yes, you can! In: Information and Privacy Commissioner Ontario, Canada, p. 26 (2013)
11. Chandramouli, B., Goldstein, J., Duan, S.: Temporal analytics on big data for web advertising. In: IEEE 28th International Conference on Data Engineering, p. 12 (2012)
12. Chen, H., Chiang, R.H.L., Storey, V.C.: Business intelligence and analytics: from big data to big impact. Bus. Intell. Res., 25 (2012)

13. Clement, M., Sokol, L., Gary, L.: Robust decision engineering: collaborative big data and its application to international development/aid. In: 8th International Conference on Collaborative Computing: Networking, Applications and Worksharing, p. 8 (2012)
14. CS4103 Distributed Systems Coursework Part 1: Big Data (2012)
15. Demchenko, Y., Zhao, Z., Grosso, P., Wibisono, A., de Laat, C.: Addressing big data challenges for scientific data infrastructure. In: IEEE 4th International Conference on Cloud Computing Technology and Science, p. 4 (2012)
16. Distributed Systems Coursework Part 1: Big Data (2012). http://www.luisramalho.com/wp-content/uploads/2012/04/bigdata.pdf
17. Dumbill, E.: Making sense of big data. 2BD, 2 (2013)
18. Geron, T.: Live: Facebook Launches Graph Search, A Social Search Engine, With Bing Partnership (2013). http://www.forbes.com/sites/tomiogeron/2013/01/15/live-facebook-announces-graph-search/
19. Greengrass, E.D.: Information Retrieval: A Survey (2000)
20. Guo, Z., Wang, J.: Information Retrieval from Large Data Sets via Multiple-winners-take-all (2011)
21. Han, X., Tian, L., Yoon, M., Lee, M.: A big data model supporting information recommendation in social networks. In: Second International Conference on Cloud and Green Computing, p. 4 (2012)
22. HPCC Systems (n.d.). HPCC Systems: Models for Big Data. White paper, 17
23. IBM big data success stories. IBM Corporation, 76 (2011)
24. Intel IT Center. Big Data Analytics. Intel's IT Manager Survey on How Organizations are Using Big Data, 27 (2012)
25. Jain, M., Singh, S.K.: A survey on: content based image retrieval systems using clustering techniques for large data sets. Int. J. Manag. Inf. Technol. (IJMIT) **3**(4), 17 (2011)
26. Ji, C., Li,, Y., Qiu, W., Awada, U., Li, K.: Big data processing in cloud computing environments. In: International Symposium on Pervasive Systems, Algorithms and Networks, p. 7 (2012)
27. Borrero, J.D., Gualda, E.: Crawling big data in a new frontier for socioeconomic research: testing with social tagging. J. Spat. Organ. Dyn. - Discussion Papers Number 12, 23 (2012)
28. Kaisler, S., Armour, F., Alberto Espinosa, J., Money, W.: Big data: issues and challenges moving forward. In: 46th Hawaii International Conference on System Sciences, p. 10 (2013)
29. Kejariwal, A.: Big data challenges a program optimization perspective. In: Second International Conference on Cloud and Green Computing, p. 6 (2012)
30. Kirkpatrick, R.: BIG data for development. BD3, **1**(1), 2 (2013)
31. Kraska, T.: Finding the Needle in the Big Data Systems Haystack, p. 3. Brown University (2013)
32. Laurila, J.K., Imad Aad, I., Perez, D.J. (n.d.).: The Mobile Data Challenge: Big Data for Mobile Computing Research
33. Lioma, C.: Big Data Challenges for Information Retrieval. University of Copenhagen-Department of Computer Science, p. 12 (2012)
34. Logothetis, D., Yocum, K.: Data Indexing for Stateful, Large-scale Data Processing (2009)
35. Lumley, T., Rice, K.: Storing and retrieving large data. UW Biostatistics, p. 18 (2009)
36. Meij, E.: *Large-scale Data Processing for Information Retrieval #nlhug,* 12 April 2012. http://www.slideshare.net/edgar.meij/largescale-data-processing-for-information-retrieval-nlhug. (Retrieved)
37. Miller, S.: How "Big Data" will change your life….. Pew Research Center's Internet & American Life Project, p. 29 (2012)
38. Nambiar, U.: Answering Imprecise Queries Over Autonomous Databases (2005). http://rakaposhi.eas.asu.edu/ullas-thesis.pdf. (Retrieved)

39. Navint Enterprise. Why is BIG Data Important?. A Navint Partners White Paper, 5 (2012). www.navint.com. (Retrieved)
40. Oracle Information Architecture: An Architect's Guide to Big Data. An Oracle White Paper in Enterprise Architecture, 25 (2012)
41. Oracle. Oracle: Big data for Enterprise. Oracle Enterprise, 16 (2012)
42. Oracle. Combining big data tools with traditional data management offers enterprises the complete view. White paper: Integrate for Insight, 4 (2012)
43. Part III: IBM's strategy for big data and analytics. IBM Corporation, 5 (2012)
44. Bennett, P.N., El-Arini, K.: Enriching Information Retrieval. In: SIGIR Workshop Report, p. 6 (2011)
45. Paz-Trillo, C., Wassermann, R., Braga, P.P.: An Information Retrieval application using Ontologies (2005). http://www.ime.usp.br/~rmcobe/onair/files/jsbc_onair.pdf
46. Provost, F., Fawcett, T.: DATA science and its relationship to big data and data-driven decision making. BD51 **1**(1), 9 (2013)
47. Rabinowitz, J.: Indexing arbitrary data with SWISH-E. In: The Proceedings of the 2004 USENIX Technical Conference, p. 7 (2004)
48. Recommender system (2013). http://en.wikipedia.org/wiki/Recommender_system
49. Rouse, M.: What is Graph Search? (2013). http://whatis.techtarget.com/definition/Graph-Search
50. Smith, M., Szongott, S., Henne, B., Voigt, G.: Big Data Privacy Issues in Public Social Media (2013)
51. Sun Yanhou, Y.: Big data in enterprise challenges & opportunities. Software and Service Group, p. 15 (2011)
52. Venkatraman, S., Kamatkar, S.J.: Intelligent information retrieval and recommender system framework. Int. J. Future Comput. Commun. **2**(2), 5 (2013)
53. Zhu, J.: Data Modeling for Big Data (2011)
54. Zhou, B., Yao, Y.: Evaluating Information Retrieval System Performance Based on User Preference. http://www2.cs.uregina.ca/~zhou200b/4-zhou.pdf
55. Zikopoulos, P., Deustch, T.: The big deal about big data. IBM Corporation, 43 (2012)

Adaptive Elitist Differential Evolution Extreme Learning Machines on Big Data: Intelligent Recognition of Invasive Species

Konstantinos Demertzis(✉) and Lazaros Iliadis

Lab of Forest-Environmental Informatics and Computational Intelligence, Democritus University of Thrace, 193 Pandazidoust., 68200 N Orestiada, Greece
{kdemertz, liliadis}@fmenr.duth.gr

Abstract. One of the direct consequences of climate change lies in the spread of invasive species, which constitute a serious and rapidly worsening threat to ecology, preservation of natural biodiversity and protection of flora and fauna. It can even be a potential threat to the health of humans. These species, do not appear to have serious morphological variations, despite their strong biological differences. Due to this fact their identification process is often quite difficult. The need to protect the environment and safeguard public health, requires the development of advanced methods for early and accurate identification of some particularly dangerous invasive species, in order to plan and apply specific and effective management measures. The aim of this study is to create an advanced computer vision system for the automatic recognition of invasive or other unknown species, based on their phenotypes. More specifically, this research proposes an innovative and very effective Extreme Learning Machine (ELM) model, which is optimized by the Adaptive Elitist Differential Evolution algorithm (AEDE). The AEDE is an improved version of the differential evolution (DE) algorithm and it is proper for big data resolution. Feature selection is done by using deep learning Convolutional Neural Networks. A Geo Location system is used to detect the invasive species by Comparing with the local species of the region under research.

Keywords: Big data · Invasive species · Adaptive Elitist Differential Evolution · Extreme Learning Machine · Machine vision · Convolutional Neural Networks

1 Introduction

1.1 Invasive Species

The potential impacts of climate change are evident at various levels of biological organization and especially in disorders found in Biodiversity, in changes done in the level of biocoenosis and in the extinction of organisms with the simultaneous appearance of invasive species [1]. Invasive species are the ones that enter into new unfamiliar habitats, they overwhelm native flora or fauna and they damage the environment. Their impacts are considered as extremely serious. The consequences are related to human health, agriculture, fisheries and food production. The usual reason for moving is searching for colder climate conditions. A typical reason is the fact that their physical environment does not meet the temperature range in which they can survive. Also they follow different plant species or organisms which migrate to cooler habitats. Although not all migrating species are harmful, the preparation of proper invasive species management plans (depending on their risk profile) is imposed preemptively and also institutionally. Their control and potential eradication should be a parallel process with the restoration of ecosystems affected by them.

The identification and classification of the invasive species, exclusively with the use of phenotypic markers, is an extremely difficult and dangerous process, as in this case there might be neither major morphological differences nor significant similarities capable of reflecting the affinity or not of the organisms (species problem) [2]. However, it is a necessary and an extremely important process in the overall strategy to address alien species. Specifically, the above method is a key preventive mechanism, as it can be used easily in low cost devices e.g. Smart phones without significant expenses, unlike the genetic methods of identification, like DNA barcoding, or methods using comparison with biochemical or molecular markers [3]. Moreover, it can be used by personnel without specialized knowledge, such as farmers or breeders.

1.2 Machine Vision and Big Data

Machine vision [4] is a scientific field of artificial intelligence which algorithmically attempts to reproduce the sense of sight. It is associated with the theoretical background and the design-manufacturing technologies of data analysis systems when data is derived from digital images, video, views from multiple cameras, multi-dimensional shapes and others. In machine vision there is a high probability of information loss, especially when converting from two to three dimensions, which is due to the transformation of the image perspective. Also, the brightness is given by the natural formation of the image, which is quite complicated. This fact makes its representation a quite complex and laborious process [5].

Some other important peculiarities to be taken into account when designing the algorithmic approach to machine vision systems are related to the change of angle, the variation of the optical scale of the objects, the deformation, the intracategorical variance of the displayed objects and the possibility of noise being [6]. Thus, the results obtained from the analysis of an image with different scenes and geometrical properties, but with different lighting conditions or with similar colors could result in misleading the system and towards incorrect categorizations.

Under this light, this is a case of big data processing, as data extracted from images require large storage space (especially if the images are in high resolution) and they should be solved by a deterministic polynomial time automaton [7]. They also require dynamic assigning of computing resources and coordination of complex data analysis procedures.

1.3 Literature Review

The authors in [8] presented a computer vision-based system that reliably classifies different fish species based on length measurement and weight determination. In [9], an ANN-based platform for fish species identification is presented that uses several statistical methods such as discriminate function analysis and principal component analysis. Rova et al. in [10] applied SVM algorithm to fish recognition and constructed a texture-based mechanism that distinguishes between the Striped Trumpeter and the Western Butterfish species. Lee et al. in [11] carried out shape analysis of fish and developed an algorithm for removing edge noise and redundant data point base on nine species with similar shape features. Also in [12] a SVM-based technique for the elimination of the limitations of some existing techniques and improved classification of fish species is proposed. In [13], an ANN and Decision Tree-based platform for fish classification based on feature selection, image segmentation and geometrical parameter techniques is presented. On the other hand, to find the near optimal network parameters of ELM, several methods have been proposed [14–18]. Also, many scholars have combined ELM with other intelligent optimization algorithms, such as particle swarm optimization (PSO) [19], differential evolution (DE) [20], cuckoo search (CS) [21], firefly algorithm (FA) [22], Grey Wolf optimizer (GSO) [23], harmony search (HS) [24], biogeography-based optimization (BBO) [25], animal migration optimization (AMO) [26], gravitational search algorithm [27], ant colony optimization [28] and interior search [29].

1.4 Innovation of the Project

An innovative element of this work is the development of a hybrid model that employs Extreme Learning Machines combined with the optimization Adaptive Elitist Differential Evolution algorithm (AEDE) [30] aiming in the optimal selection of the input layer weights and the bias. This is done in order to achieve a higher level of generalization. Another interesting point is the performance of feature extraction using Convolutional Neural Networks (CNN) [31] (ELM classifier) and the optimization with the AEDE approach. This hybrid method combines two highly effective, biologically inspired, machine learning algorithms, for solving a multidimensional and complex machine vision problem. Another important issue that supports the autonomous operation of the system is the addition of geolocation capability (using global position systems) of the identified species and the performance of comparisons with native ones. The system is pouring artificial intelligence to digital cameras that can easily (quickly and at low cost) identify invasive or rare species on the basis of their phenotypes. This would result in strengthening biosecurity programs.

2 Materials and Methods

2.1 Data

The fish image data set [32] (currently consisting of 3,961 images) is related to 469 species. This data consists of real-world images of fish captured in conditions defined as “controlled”, “out-of-the-water” and “in-situ”. The “controlled”, images consist of fish specimens, with their fins spread, taken against a constant background with controlled illumination. The “in-situ” ones are underwater images of fish in their natural habitat and so there is no control over background or illumination. The “out-of-the-water” are related to fish specimens, taken out of the water with a varying background and limited control over the illumination conditions. A tight red bounding box is annotated around the fish. Sketch images are not used in this paper. There are two main difficulties when performing classification on the fish imagery.

In many cases, different species are visually similar, as shown Fig. 1(a)–(d). Also many times, there is a high degree of variability in the image quality and environmental conditions (see Fig. 2). Approximately half of the images have been captured in the “controlled” condition, where the image of the fish has been captured out-of-the-water with a controlled back-ground. The “in-situ” condition consists of images taken underwater with no control over the background and with significant pose and illumination variations. Approximately one third of the data was captured in this manner. Finally, the remaining images are captured “out-of-the-water”, but without a controlled background and may contain some minor pose variation [32].

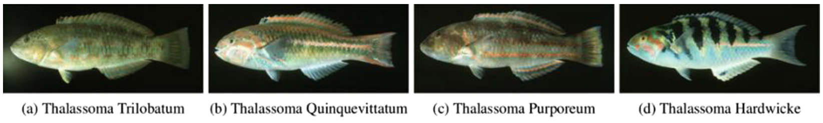


Fig. 1. Example images of four different fish species. All of them have similar visual appearance despite being distinct species. (Images taken by J.E. Randall) [30]

2.2 The Curse of Dimensionality and Feature Simplification

Convolution Neural Networks are a case of Feed Forward Neural Networks (FFNN). Their particularity lies in the neuronal levels preceding the FFNN which are viewed as filters in particular subsets of the available data. The neuronal levels commonly used in CNN networks are divided into the following main categories namely: Convolutional Layers (CL), Max-Pooling Layers (MaxPL), Feed Forward Layer (FFL) [31]. In the Machine Vision applications of Deep Learning architectures, the input parameters are the pixels. Each pixel is considered as a distinct feature. The classification by this method is too complex, since assuming a 200 × 200-pixel image it actually contains 40,000 pixels which correspond to the input data features. In this case 40,000 neurons are required to form a fully connected topology. The final result is the employment of 1,600,000,000 parameters. It is obvious that the computational cost with the described architecture is huge and that the application of an effective feature simplification method would offer invaluable services.

A CNN model is commonly applied in Machine Vision cases for this purpose. In this research effort, CNN has been employed as a method of feature simplification [31] and the extracted data have been fed in the input layer in standard machine learning systems.

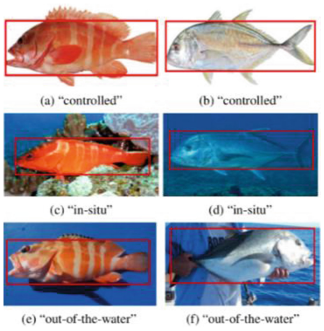


Fig. 2. The three different capture conditions: "controlled", "in-situ" and "out-of-the-water". Significant variation in appearance due to the changed imaging conditions (session variation) is evident. Ground truth bounding boxes are shown in red [32].

The dimensions of the pictures used in this research were 81 × 42 and 96 filters were applied (feature map). The picture was divided in frames of 9 × 7 pixels (in order to avoid the frame of 1 × 1 pixels that would lead to an intangible to handle feature vector. To perform feature extraction, the following procedure was used:

The dimensions of the convolution layer were 81 × 42 × 96 which means that it contained 81 × 42 = 3402 neurons for each one of the 96 used filters, which leads to a total number of neurons equal to 81 × 42 × 96 = 326,592. Each one of these neurons had 9 × 7 × 3 + 1 = 190 weights plus a bias, (the number 3 is used due to the RGB color model used that requires 1 channel for each color), which gives a total number of 326,592 × 190 = 62,052,480 weights and biases values which corresponds to an intangible number of input data to be analyzed.

In order to avoid this huge problem, we accepted that the neurons belonging to the same filter were assigned the same weights. This means that a filter with 3402 neurons can have only 190 weights instead of 3402 × 190 = 646,380. In this way the weights plus bias were reduced significantly to 96 × (9 × 7 × 3 + 1) = 18,240. This assumption is not arbitrary but it is based on the logical statement that the application of a filter can have useful results (obtained characteristics) regardless the position in which it may be applied [33, 34].

A determining factor for delivering high precision and low uncertainty in machine vision metrology is the resolution of the acquired image. As a gauge, the smallest unit of measurement in a machine vision system is the single pixel. As with any measurement system, in order to make a repeatable and reliable measurement one must use a gauge where the smallest measurement unit (as a general rule of thumb) is one tenth of the required measurement tolerance band [6].

3 The Proposed Algorithm

3.1 Extreme Learning Machines (ELM)

Extreme Learning Machines are a kind of the Single-Hidden-Layer feed forward Neural Networks (SLFNs). ELMs are characterized by the possibility to establish the parameters of hidden nodes at random, before they see the training data vectors. They are extremely fast and effective and they are capable of handling a wide range of activation functions (e.g. stopping criterion, learning rate and learning epochs [35]. The output of an ELM with a training set $\{(x_i, t_i)\}_{i=1}^{N}$ consisting of N discrete samples ($x_i \in R^n$ *and* $t_i \in R^m$) with an activation function $g(x)$ and l hidden nodes is given by the following function 1 [35]:

$$t_i = \sum\nolimits_{j=1}^{l} \beta_j g\left(\omega_j x_i + b_j\right), i = 1, 2, \ldots, N, \tag{1}$$

Where $\beta_j = \left[\beta_{j1}, \beta_{j2}, \ldots, \beta_{jm}\right]^T$ is the weight vector connecting the output neurons and the *j-th* hidden neuron, $\omega_j = \left[\omega_{j1}, \omega_{j2}, \ldots, \omega_{jm}\right]^T$ is the weight vector connecting the input neurons and the *j-th* hidden neuron, b_j is the bias of the *j-th* hidden neuron, and $\omega_j . x_i$ indicates the inner product of ω_j and x_i. Function 1 can also be written as follows:

$$H\beta = T, \tag{2}$$

where

$$H\left(\omega_j, b_j, x_i\right) = \begin{bmatrix} g(\omega_1 x_1 + b_1) & \cdots & g(\omega_l x_1 + b_l) \\ \vdots & \ddots & \vdots \\ g(\omega_1 x_N + b_1) & \cdots & g(\omega_l x_N + b_l) \end{bmatrix}_{N \times l}, \tag{3}$$

Also H is called the hidden layer output matrix of the network. Before training, the input weights matrix ω and the hidden biases vector b are created randomly in the interval [−1, 1], where $\omega_j = \left[\omega_{j1}, \omega_{j2}, \ldots, \omega_{jm}\right]^T$ and $\beta_j = \left[\beta_{j1}, \beta_{j2}, \ldots, \beta_{jm}\right]^T$.

Then the hidden layer output matrix H is calculated by the activation function and by using the training set based on the function $H = g(\omega x + b)$.

Finally, the output weights matrix β is estimated by the equation $\beta = H^{+} T$, where H^{+} is the Moore–Penrose generalized inverse of matrix H and $T = [t_1, t_2, \ldots, t_N]^T$ [35].

3.2 Adaptive Elitist Differential Evolution (AEDE)

In evolutionary computation, Differential Evolution (DE) [36] is a method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. In the DE, the parameters such as mutant factor F and crossover control parameter CR and trial vector generation strategies have significant influence on its performance. To overcome the common limitations of optimization algorithms such as the use of a huge volume of resources (e.g. high computational cost) the Adaptive Elitist Differential Evolution algorithm (AEDE) [30] introduces two alternatives. The first one is applied in the mutation phase and the second one in the selection phase, in order to enhance the search capability as well as the convergence speed of the DE algorithm. The new adaptive mutation scheme of the DE uses two mutation operators. The first one is the "rand/1" which aims to ensure diversity of the population and prohibits the population from getting stuck in a local optimum. The second is the "current-to-best/1" which aims to accelerate convergence speed of the population by guiding the population toward the best individual. On the other hand, the new selection mechanism is performed as follows: Firstly, the children population *C* consisting of trial vectors is combined with the parent population *P* of target vectors to create a combined population *Q*. Then, *NP* best individuals are chosen from the *Q* to construct the population for the next generation. In this way, the best individuals of the whole population are always stored for the next generation. This helps the algorithm to obtain a better convergence rate [30]. The elitist selection operator is presented in the following Algorithm 1.

Algorithm 1. Elitist selection operator [30]

```
1: Input: Children population $C$ and parent population $P$
2: Assign
3: Select $NP$ best individuals from $Q$ and assign to $P$
4: Output: $P$
```

The AEDE method is summarily shown as in Algorithm 2 below [30]:
where *tolerance* is the allowed error; *MaxIter* is the maximum number of iterations; and randint(1, D) is the function which returns a uniformly distributed random integer number between 1 and D.

3.3 Adaptive Elitist Differential Evolution ELM (AEDE-ELM)

Given that ELMs produce the initial weights (weights) and (bias) randomly, the process may not reach the optimal result, which may not imply as high classification accuracy as the desired one. The optimal choice of weights and bias, create the conditions for maximum potential accuracy and of course the best generalization performance of the ELMs. To solve the above problem, we recommend the use of the AEDE optimization method for the optimal selection of weights and bias of the ELMs. Initially, each individual in the first generation is obtained randomly, and it is composed of the input weights and hidden biases: $x = [\omega_1, \omega_2, \ldots, \omega_l, b_1, b_2, \ldots, b_l]$.

Secondly, the corresponding output weights matrix for each individual is calculated in the manner of the ELM algorithm. Then, we apply AEDE to find the fitness for each individual in the population. Finally, when the evolution is over, we can use the optimal parameters of the ELM to perform the classification.

The procedure of AEDE-ELM algorithm is shown by Algorithm 3.

Algorithm 2. The adaptive elitist Differential Evolution (aeDE) algorithm [30]

```
1: Generate the initial population
2: Evaluate the fitness for each individual in the population
//Definition of searching criteria
3: while delta > tolerance or MaxIter is not reached do
//Find the best individuals
4: for i =1 to NP do
//Generate the initial mutation factor
5:      F = rand[0.4, 1]
//Generate the initial crossover control parameter
6:     CR = rand[0.7, 1]
//Select a random integer number between 1 and D
7:      $j_{rand}$ = randint(1, D)
//Find the optimal parameters
8:    for j =1 to D do
//Check the crossover operation
9:   if rand[0, 1] < CR or j == $j_{rand}$ then
//Check the mutation
10:   if delta > threshold then
//Select the optimal parameters
11:               Select randomly r1 ≠r2 ≠r3 ≠i;
                                                    $\forall i \in \{1, \ldots, NP\}$
12:              $u_{ij} = x_{r1j} + F \times (x_{r2j} - x_{r3j})$
13:           else
14:             Select randomly r1 ≠r2 ≠best ≠i;
                                                    $\forall i \in \{1, \ldots, NP\}$
15:              $u_{ij} = x_{ij} + F \times (x_{bestj} - x_{ij}) + F \times (x_{r2j} - x_{r3j})$
16:           end if
17:         else
18:           $u_{ij} = x_{ij}$
19:         end if
20:    end for
21:Evaluate the trial vector $\mathbf{u}_i$
22: end for
23: Do selection phase based on Algorithm 1
24: Define $f_{best}, f_{mean}$
25: delta = $\left|\frac{f_{best}}{f_{mean-1}}\right|$
26: end while
```

3.4 GPS Country Location

The Global Positioning System (GPS) can locate stationary or moving users, based on a "grid" of 24 Earth satellites equipped with special tracking devices, called "GPS transceivers". These transceivers are providing accurate coordinates for the position of a point (Longitude - Latitude). For the accurate determination of the country to which the coordinates (reported by a GPS device) belong, the system follows a process which considers the global borders of the countries as they originally appear in the shapefiles which are free available online at the following link: http://thematicmapping.org/downloads/world_borders.php. In the following script example, the shapefile is imported through the Python platform and the coordinates 39.35230, 24.41232 belonging to Greece are checked [37]:

```
import countries
    cc = countries.CountryChecker('TM_WORLD_BORDERS-0.3.shp')
    print cc.getCountry(countries.Point(39.35230, 24.41232)).iso
print Greece
```

Algorithm 3. aeDE-ELM algorithm

```
Input:
   Training set, testing set;
   aeDE algorithm parameters, NP;
1: Create a random initial population;
2: Evaluate the fitness for each individual with training set;
3: while (stopping criteria not met) do
4:        Randomly generate $F_i$ and $CR_i$
5:  for i=1 to NP do
6:             Call the Algorithm 2;
7:             Use the optimal parameters of ELM;
8:        end for
9:   end while
10: Evaluate the optimized model by testing set;
Output:
   Classification result;
```

4 Algorithmic Steps of the Proposed Hybrid Approach

The algorithmic approach of the proposed hybrid scheme includes (in the first stage) the feature extraction process using CNN, in order to extract features from each photo of the fish dataset. In the second stage, these features are introduced in the proposed ELM, which is optimized by the AEDE approach, to yield the maximum classification accuracy in order to identify the type of fish detected by the machine vision system. Having identified the type, the coordinates are obtained by the GPS and they are mapped to the country where they belong. Then a crosscheck is made to find out whether the type identified is indigenous in this country, or otherwise it is labeled as invasive species. The lists of indigenous and invasive species were exported from the Invasive Species Compendium (http://www.cabi.org/isc/) [38], the most authoritative and comprehensive database on the subject that exists worldwide. The algorithm for identifying as invasive species a species is presented below (Fig. 3):

Algorithm 4. Geolocation Process

```
Input:
   Recognized_Fish;
   Country;
   Country_Native_Fishes;
1: Read Recognized_Fish, Country, Country_Native_Fishes;
2:    for i=1 to Country_Native_Fishes [max]do
3:          if Country_Native_Fishes[i]= Recognized_Fish then
4:                      Recognized Fish=! Recognized Fish;
5:             Recognized Fish=Native_Fish
6:          else
7:   Recognized Fish=Invasive_Fish
8: end if
9:    end
Output:
Fish Identity;
```

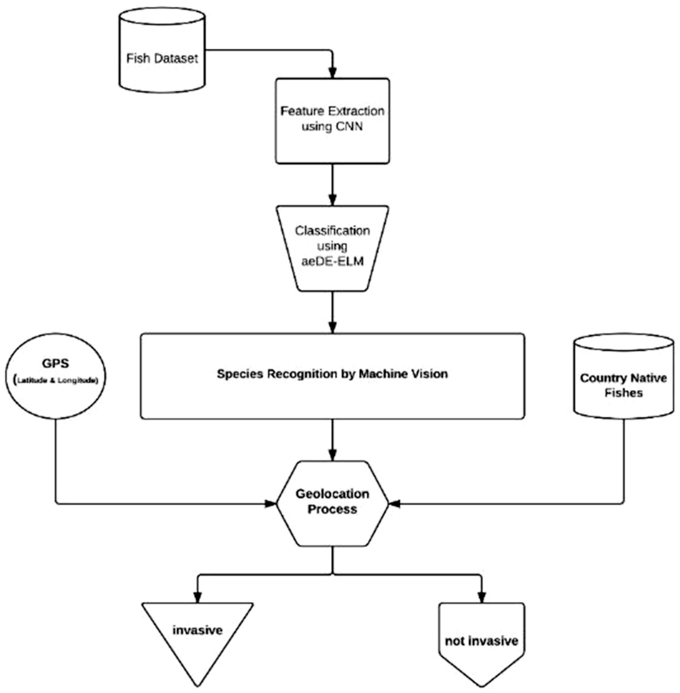

Fish Dataset
Feature Extraction using CNN
Classification using aeDE-ELM
GPS (Latitude & Longitude)
Species Recognition by Machine Vision
Country Native Fishes
Geolocation Process
invasive
not invasive

Fig. 3. The proposed architecture of the proposed machine vision system

5 Results and Comparative Analysis

It is extremely comforting and hopeful, the fact that the proposed hybrid system manages to solve a particularly complex computer vision problem with high accuracy, regardless the fact that the dataset used for training and evaluation of the proposed AEDE-ELM is highly complex, given the similarity between the species tested and the specificities resulting from the feature extraction process using CNN. It is also characteristic that in the process of categorization of the species tested, the accuracy rate in all cases ("controlled", "in-situ", "out-of-the-water") was very high. This fact suggests and confirms the generalization capabilities of the proposed system.

The following Table 1, presents the analytical values of the predictive power of the AEDE-ELM by using a 10 Fold Cross Validation approach (10-FoldCV) and the corresponding results when competitive algorithms were used, namely: Differential Evolution ELM (DE-ELM), Artificial Neural Network (ANN) and Support Vector Machine (SVM).

The Precision measure shows what percentage of positive predictions where correct, whereas Recall measures what percentage of positive events were correctly predicted. The F-Score can be interpreted as a weighted average of the precision and recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F-Score is usually more useful than accuracy and it works best if false positives and false negatives have similar

Table 1. Performance metrics and comparisons

Classifier	Classification accuracy (ACC) & Performance metrics						
	ACC	RMSE	Precision	Recall	F-Score	ROC	Evaluation
AEDE-ELM	**94.81 %**	**0.1673**	**0.948 %**	**0.948**	**0.949 %**	**0.985**	**10-Fold CV**
DE-ELM	93.97 %	0.1711	0.940 %	0.940	0.940 %	0.985	10-Fold CV
ELM	93.13 %	0.1726	0.932 %	0.932	0.932 %	0.982	10-Fold CV
ANN	92.89 %	0.1804	0.930 %	0.929	0.929 %	0.978	10-Fold CV
SVM	91.35 %	0.2031	0.914 %	0.914	0.914 %	0.960	10-Fold CV

Table 2. Performance metrics and comparisons of “Lagocephalus Sceleratus”

Classifier	Classification accuracy (ACC) & Performance metrics						
	ACC	RMSE	Precision	Recall	F-Score	ROC	Evaluation
AEDE-ELM	**97.52 %**	**0.1247**	**0.972 %**	**0.972**	**0.972 %**	**0.993**	**10-Fold CV**
DE-ELM	97.01 %	0.1289	0.970 %	0.970	0.971 %	0.990	10-Fold CV
ELM	95.86 %	0.1461	0.960 %	0.958	0.959 %	0.965	10-Fold CV
ANN	95.70 %	0.1482	0.957 %	0.957	0.957 %	0.957	10-Fold CV
SVM	95.62 %	0.1502	0.956 %	0.956	0.957 %	0.957	10-Fold CV

cost, in this case. Finally, the ROC curve is related in a direct and natural way to cost/benefit analysis of diagnostic decision making. This comparison generates encouraging expectations for the identification of the AEDE-ELM as a robust classification model suitable for difficult problems. This method was tested with great success in the control and automatic recognition of the highly dangerous for the public health Mediterranean invasive species fish “Lagocephalus Sceleratus” (Table 2).

6 Discussion and Conclusions

An innovative biologically inspired hybrid computational intelligence approach suitable for big data was presented in this research paper. It is a machine vision system which can recognize various fish species in order to rank them as to whether they are invasive in the region identified or not. Specifically, the hybrid and innovative AEDE-ELM algorithm was suggested which uses the innovative and highly effective algorithm AEDE in order to optimize the operating parameters of an ELM. The CNN feature extraction method was also used for the training of the model. The system can operate in an autonomous mode, because it estimates the geolocation of the species and it decides if it can be considered local or invasive. The wide application of the proposed method which simplifies and reduces to a minimum the computation cost and the identification time and the wide collection of respective data vectors are prerequisites for the establishment of an effective management and risk prevention system.

A future extension would be the use and incorporation to the system, of other methods for similar characteristics determination like: Representational Similarity Analysis, Local Similarity Analysis, Isoperimetry and Gaussian Analysis. Finally, it would be very important to use and test the performance of the Deep Extreme Learning Machines technology for this specific case.

References

1. Rahel, F., Olden, J.D.: Assessing the effects of climate change on aquatic invasive species. Soc. Conserv. Biol. **22**(3), 521–533 (2008)
2. Miller, W.: The structure of species, outcomes of speciation and the species problem: Ideas for paleobiology. Palaeoclimatol. Palaeoecol. **176**, 1–10 (2001)
3. Demertzis, K., Iliadis, L.: Intelligent bio-inspired detection of food borne pathogen by DNA barcodes: the case of invasive fish species Lagocephalus Sceleratus. Eng. Appl. Neural Netw. **517**, 89–99 (2015). doi:10.1007/978-3-319-23983-5_9
4. Hornberg, A.: Handbook of Machine Vision, p. 709. Wiley, Hoboken (2006). ISBN: 978-3-527-40584-8
5. Graves, M., Batchelor, B.G.: Machine Vision for the Inspection of Natural Products, p. 5. Springer, London (2003). ISBN: 978-1-85233-525-0
6. Carsten, S., Ulrich, M., Wiedemann, C.: Machine Vision Algorithms and Applications, p. 1. Wiley-VCH, Weinheim (2008). ISBN: 978-3-527-40734-7
7. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, vol. 25 (2012)
8. Svellingen, C., Totland, B., White, D., Ovredal, T.: Automatic Species Recognition, length measurement and weight determination using the CatchMeter Computer Vision System (2006)
9. Cabreira, A.G., Tripode, M., Madirolas, A.: Artificial neural networks for fish-species identification. ICES J. Mar. Sci. **66**, 1119–1129 (2009)
10. Rova, A., Mori, G., Dill, L.M.: One fish, two fish, butterfish, trumpeter: recognizing fish in underwater video. In: Conference on Machine Vision Applications, pp. 404–407 (2007)
11. Lee, D.J., Schoenberger, R., Shiozawa, D., Xu, X., Zhan, P.: Contour matching for a fish recognition and migration monitoring system. Stud. Comput. Intell. **122**, 183–207 (2008)
12. Ogunlana, S.O., Olabode, O., Oluwadare, S.A.A., Iwasokun, G.B.: Fish classification using SVM. IEEE Afr. J. Comput. ICT **8**(2), 75–82 (2015)
13. Mutasem, K.A., Khairuddin, B.O., Shahrulazman, N., Ibrahim, A.: Fish recognition based on the combination between robust features selection, image segmentation and geometrical parameters techniques using artificial neural network and decision tree. J. Comput. Sci. Inf. Secur. **6**(2), 215–221 (2009)
14. Zhu, Q.Y., Qin, A.K., Suganthan, P.N., Huang, G.B.: Evolutionary extreme learning machine. Pattern Recogn. **38**, 1759–1763 (2005)
15. Qu, Y., Shen, Q., Parthaláin, N.M., Wu, W.: Extreme learning machine for mammograhic risk analysis. In: UK Workshop on Computational Intelligence, pp. 1–5 (2010)
16. Sridevi, N., Subashini, P.: Combining Zernike moments with regional features for Classification of Handwritten Ancient Tamil Scripts using Extreme Learning Machine. In: IEEE IC Emerging Trends in Computing, Communication and Nanotechnology, pp. 158–162 (2013)
17. Wang, D.D., Wang, R., Yan, H.: Fast prediction of protein-protein interaction sites based on extreme learning machines. Neurocomputing **77**, 258–266 (2014)

18. Bazi, Y., Alajlan, N., Melgani, F., AlHichri, H., Malek, S., Yager, R.R.: Differential Evolution Extreme Learning Machine for the Classification of Hyperspectral Images. IEEE Geosci. Remote Sens. Lett. **11**, 1066–1070 (2014)
19. Zhao, X.: A perturbed particle swarm algorithm for numerical optimization. Appl. Soft Comput. **10**(1), 119–124 (2010). doi:10.1016/j.asoc.2009.06.010
20. Li, X., Yin, M.: Application of differential evolution algorithm on self-potential data. PLoS ONE **7**(12), e51199 (2012). doi:10.1371/journal.pone.0051199
21. Gandomi, A.H., Yang, X.-S., Alavi, A.H.: Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng. Comput. **29**(1), 17–35 (2013)
22. Wang, G.-G., Guo, L., Duan, H., Wang, H.: A new improved firefly algorithm for global numerical optimization. J. Comput. Theor. Nanosci. **11**(2), 477–485 (2014). doi:10.1166/jctn.2014.3383
23. Mirjalili, S., Mirjalili, S.M., Lewis, A.: Grey wolf optimizer. Adv. Eng. Softw. **69**, 46–61 (2014). doi:10.1016/j.advengsoft.2013.12.007
24. Geem, Z.W., Kim, J.H., Loganathan, G.V.: A new heuristic optimization algorithm: harmony search. Simulation **76**(2), 60–68 (2001). doi:10.1177/003754970107600201
25. Simon, D.: Biogeography-based optimization. IEEE Trans. Evolut. Comput. **12**(6), 702–713 (2008). doi:10.1109/TEVC.2008.919004
26. Li, X., Zhang, J., Yin, M.: Animal migration optimization: an optimization algorithm inspired by animal migration behavior. Neural Comput. Appl. **24**(7–8), 1867–1877 (2014). doi:10.1007/s00521-013-1433-8
27. Mirjalili, S., Mohd Hashim, S.Z., Moradian Sardroudi, H.: Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm. Appl. Math. Comput. **218**(22), 11125–11137 (2012). doi:10.1016/j.amc.2012.04.069
28. Zhang, Z., Zhang, N., Feng, Z.: Multi-satellite control resource scheduling based on ant colony optimization. Expert Syst. Appl. **41**(6), 2816–2823 (2014). doi:10.1016/j.eswa.2013.10.014
29. Gandomi, A.H.: Interior search algorithm (ISA): a novel approach for global optimization. ISA Trans. **53**(4), 1168–1183 (2014). doi:10.1016/j.isatra.2014.03.018
30. Ho-Huu, V., Nguyen-Thoi, T., Vo-Duy, T., Nguyen-Trang, T.: An adaptive elitist differential evolution for optimization of truss structures with discrete design variables. Comput. Struct. **165**, 59–75 (2016)
31. Bluche, T., Ney, H., Kermorvant, C.: Feature extraction with convolutional neural networks for handwritten word recognition. In: 12th International Conference on Document Analysis and Recognition, pp. 285–289. IEEE (2013)
32. Anantharajah, K., Ge, Z., McCool, C., Denman, S., Fookes, C., Corke, P., Tjondronegoro, D., Sridharan, S.: Local inter-session variability modelling for object classification. In: IEEE Winter Conference on Applications of Computer Vision (WACV 2014). Steamboat Springs, Co., 24–26 March 2014
33. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. IJCV (2015). arXiv:1409.0575
34. Jia, D.J., Vinyals, Y., Hoffman, O., Zhang, J., Tzeng, N., Darrell, E.T.: Decaf: a deep convolutional activation feature for generic visual recognition. CoRR, abs/1310.1531 (2013)
35. Cambria, E., Huang, G.-B.: Extreme learning machines. IEEE Intell. Syst. **28**, 37–134 (2013)
36. Price, K., Storn, M., Lampinen, A.: Differential Evolution: A Practical Approach to Global Optimization. Springer, Heidelberg (2005). ISBN: 978-3-540-20950-8
37. https://github.com/che0/countries
38. http://www.cabi.org/isc/

Author Index