

Hujun Yin · David Camacho
Paulo Novais
Antonio J. Tallón-Ballesteros (Eds.)

LNCS 11314

Intelligent Data Engineering and Automated Learning – IDEAL 2018

19th International Conference
Madrid, Spain, November 21–23, 2018
Proceedings, Part I

1
Part I

 Springer

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, Lancaster, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Friedemann Mattern

ETH Zurich, Zurich, Switzerland

John C. Mitchell

Stanford University, Stanford, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

C. Pandu Rangan

Indian Institute of Technology Madras, Chennai, India

Bernhard Steffen

TU Dortmund University, Dortmund, Germany

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Gerhard Weikum

Max Planck Institute for Informatics, Saarbrücken, Germany

More information about this series at <http://www.springer.com/series/7409>

Hujun Yin · David Camacho
Paulo Novais · Antonio J. Tallón-Ballesteros (Eds.)


Intelligent Data Engineering and Automated Learning – IDEAL 2018

19th International Conference
Madrid, Spain, November 21–23, 2018
Proceedings, Part I

Editors

Hujun Yin 
University of Manchester
Manchester, UK

David Camacho 
Autonomous University of Madrid
Madrid, Spain

Paulo Novais 
Campus of Gualtar
University of Minho
Braga, Portugal

Antonio J. Tallón-Ballesteros
University of Seville
Seville, Spain

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-030-03492-4 ISBN 978-3-030-03493-1 (eBook)
<https://doi.org/10.1007/978-3-030-03493-1>

Library of Congress Control Number: 2018960396

LNCS Sublibrary: SL3 – Information Systems and Applications, incl. Internet/Web, and HCI

© Springer Nature Switzerland AG 2018

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

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

Preface

This year saw the 19th edition of the International Conference on Intelligent Data Engineering and Automated Learning (IDEAL), which has been playing an increasingly leading role in the era of big data and deep learning. As an established international forum, it serves the scientific communities and provides a platform for active, new, and leading researchers in the world to exchange the latest results and disseminate new findings. The IDEAL conference has continued to stimulate the communities and to encourage young researchers for cutting-edge solutions and state-of-the-art techniques on real-world problems in this digital age. The IDEAL conference attracts international experts, researchers, academics, practitioners, and industrialists from machine learning, computational intelligence, novel computing paradigms, data mining, knowledge management, biology, neuroscience, bio-inspired systems and agents, distributed systems, and robotics. It also continues to evolve to embrace emerging topics and trends.

This year IDEAL was held in one of most beautiful historic cities in Europe, Madrid. In total 204 submissions were received and subsequently underwent the rigorous peer-review process by the Program Committee members and experts. Only the papers judged to be of the highest quality were accepted and are included in the proceedings. This volume contains 125 papers (88 for the main rack and 37 for workshops and special sessions) accepted and presented at IDEAL 2018, held during November 21–23, 2018, in Madrid, Spain. These papers provided a timely sample of the latest advances in data engineering and automated learning, from methodologies, frameworks, and techniques to applications. In addition to various topics such as evolutionary algorithms, deep learning neural networks, probabilistic modeling, particle swarm intelligence, big data analytics, and applications in image recognition, regression, classification, clustering, medical and biological modeling and prediction, text processing and social media analysis. IDEAL 2018 also enjoyed outstanding keynotes from leaders in the field, Vincenzo Loia, Xin Yao, Alexander Gammerman, as well as stimulating tutorials from Xin-She Yang, Alejandro Martin-Garcia, Raul Lara-Cabrera, and David Camacho.

The 19th edition of the IDEAL conference was hosted by the Polytechnic School at Universidad Autónoma de Madrid (UAM), Spain. With more than 30,000 students, and 2,500 professors and researchers and a staff of over 1,000, the UAM offers a comprehensive range of studies in its eight faculties (including the Polytechnic School). UAM is also proud of its strong research commitment that is reinforced by its six university hospitals and the ten joint institutes with CSIC, Spain's National Research Council.

We would like to thank all the people who devoted so much time and effort to the successful running of the conference, in particular the members of the Program Committee and reviewers, organizers of workshops and special sessions, as well as the authors who contributed to the conference. We are also very grateful to the hard work

by the local organizing team at Universidad Autónoma de Madrid, especially Victor Rodríguez, for the local arrangements, as well as the help from Yao Peng at the University of Manchester for checking through all the camera-ready files. The continued support and collaboration from Springer LNCS are also greatly appreciated.

September 2018

Hujun Yin
David Camacho
Paulo Novais
Antonio J. Tallón-Ballesteros

Organization

Honorary Chairs

Hojjat Adeli
Francisco Herrera

Ohio State University, USA
Granada University, Spain

General Chairs

David Camacho
Hujun Yin
Emilio Corchado

Universidad Autónoma de Madrid, Spain
University of Manchester, UK
University of Salamanca, Spain

Programme Co-chairs

Carlos Cotta
Antonio J. Tallón-Ballesteros
Paulo Novais

Universidad de Málaga, Spain
University of Seville, Spain
Universidade do Minho, Portugal

International Advisory Committee

Lei Xu (Chair)

Chinese University of Hong Kong and Shanghai
Jiaotong University, China

Yaser Abu-Mostafa

CALTECH, USA

Shun-ichi Amari

RIKEN, Japan

Michael Dempster

University of Cambridge, UK

José R. Dorronsoro

Autonomous University of Madrid, Spain

Nick Jennings

University of Southampton, UK

Soo-Young Lee

KAIST, South Korea

Erkki Oja

Helsinki University of Technology, Finland

Latit M. Patnaik

Indian Institute of Science, India

Burkhard Rost

Columbia University, USA

Xin Yao

Southern University of Science and Technology,
China and University of Birmingham, UK

Steering Committee

Hujun Yin (Chair)

University of Manchester, UK

Laiwan Chan (Chair)

Chinese University of Hong Kong, Hong Kong,
SAR China

Guilherme Barreto

Federal University of Ceará, Brazil

Yiu-ming Cheung

Hong Kong Baptist University, Hong Kong,
SAR China

Emilio Corchado	University of Salamanca, Spain
Jose A. Costa	Federal University of Rio Grande do Norte, Brazil
Marc van Hulle	K. U. Leuven, Belgium
Samuel Kaski	Aalto University, Finland
John Keane	University of Manchester, UK
Jimmy Lee	Chinese University of Hong Kong, Hong Kong, SAR China
Malik Magdon-Ismail	Rensselaer Polytechnic Inst., USA
Peter Tino	University of Birmingham, UK
Zheng Rong Yang	University of Exeter, UK
Ning Zhong	Maebashi Institute of Technology, Japan

Publicity Co-chairs/Liaisons

Jose A. Costa	Federal University of Rio Grande do Norte, Brazil
Bin Li	University of Science and Technology of China, China
Yimin Wen	Guilin University of Electronic Technology, China

Local Arrangements Chairs

Antonio González Pardo	Raúl Lara Cabrera
Cristian Ramírez Atencia	Raquel Menéndez Ferreira
Víctor Rodríguez Fernández	F. Javier Torregrosa López
Alejandro Martín García	Ángel Panizo Lledot
Alfonso Ortega de la Puente	Marina de la Cruz

Programme Committee

Paulo Adeodata	Zoran Bosnic
Imtiaj Ahmed	Vicent Botti
Jesus Alcalá-Fdez	Edyta Brzywczy
Richard Aler	Andrea Burattin
Davide Anguita	Robert Burduk
Ángel Arcos-Vargas	José Luis Calvo Rolle
Romis Attux	Heloisa Camargo
Martin Atzmüller	Josep Carmona
Javier Bajo Pérez	Mercedes Carnero
Mahmoud Barhamgi	Carlos Carrascosa
Bruno Baruque	Andre Carvalho
Carmelo Bastos Filho	Pedro Castillo
José Manuel Benitez	Luis Cavique
Szymon Bobek	Darryl Charles
Lordes Borrajo	Francisco Chavez

Richard Chbeir
Songcan Chen
Xiaohong Chen
Sung-Bae Cho
Stelvio Cimato
Manuel Jesus Cobo Martin
Roberto Confalonieri
Rafael Corchuelo
Juan Cordero
Oscar Cordon
Francesco Corona
Luís Correia
Paulo Cortez
Jose Alfredo F. Costa
Carlos Cotta
Raúl Cruz-Barbosa
Ernesto Cuadros-Vargas
Bogusław Cyganek
Ireneusz Czarnowski
Ernesto Damiani
Ajalmar Rêgo Darocha Neto
Javier Del Ser
Boris Delibašić
Fernando Díaz
Juan Manuel Dodero
Bernabe Dorronsoro
Jose Dorronsoro
Gérard Dreyfus
Adrião Duarte
Jochen Einbeck
Florentino Fdez-Riverola
Francisco Fernandez De Vega
Joaquim Filipe
Juan J. Flores
Pawel Forczmanski
Giancarlo Fortino
Felipe M. G. França
Dariusz Frejlichowski
Hamido Fujita
Marcus Gallagher
Ines Galvan
Matiaz Gams
Yang Gao
Jesus Garcia
Salvador Garcia
Pablo García Sánchez
Ana Belén Gil
María José Ginzo Villamayor
Fernando Gomide
Antonio Gonzalez-Pardo
Pedro González Calero
Marcin Gorawski
Juan Manuel Górriz
Manuel Graña
Maciej Grzenda
Jerzy Grzymala-Busse
Juan Manuel Górriz
Barbara Hammer
Richard Hankins
Ioannis Hatzilygeroudis
Francisco Herrera
Álvaro Herrero
J. Michael Herrmann
Ignacio Hidalgo
James Hogan
Jaakko Hollmén
Vasant Honavar
Wei-Chiang Samuelson Hong
Anne Håkansson
Iñaki Inza
Vladimir Ivančević
Dušan Jakovetić
Vahid Jalali
Dariusz Jankowski
Vicente Julian
Rushed Kanawati
Benjamin Klöpper
Mario Koeppen
Ilkka Kosunen
Miklós Krész
Raul Lara-Cabrera
Florin Leon
Bin Li
Clodoaldo Lima
Ivan Lukovic
Wenjian Luo
Mihai Lupu
M. Victoria Luzon
Felix Mannhardt
Alejandro Martin
José F. Martínez-Trinidad
Giancarlo Mauri

Raquel Menéndez Ferreira
José M. Molina
Mati Mottus
Valery Naranjo
Susana Nascimento
Tim Nattkemper
Antonio Neme
Ngoc-Thanh Nguyen
Yusuke Nojima
Fernando Nuñez
Eva Onaindia
Jose Palma
Ángel Panizo Lledot
Juan Pavón
Yao Peng
Carlos Pereira
Sarajane M. Peres
Costin Pribeanu
Paulo Quaresma
Juan Rada-Vilela
Cristian Ramírez-Atencia
Izabela Rejer
Victor Rodriguez Fernandez
Zoila Ruiz
Luis Rus-Pegalajar
Yago Saez

Jaime Salvador
Jose Santos
Matilde Santos
Dragan Simic
Anabela Simões
Marcin Szpyrka
Jesús Sánchez-Oro
Ying Tan
Ricardo Tanscheit
Renato Tinós
Stefania Tomasiello
Pawel Trajdos
Stefan Trausan-Matu
Carlos M. Travieso-González
Milan Tuba
Turki Turki
Eiji Uchino
José Valente de Oliveira
José R. Villar
Lipo Wang
Tzai-Der Wang
Dongqing Wei
Michal Wozniak
Xin-She Yang
Weili Zhang

Additional Reviewers

Mahmoud Barhamgi
Gema Bello
Carlos Camacho
Carlos Casanova
Laura Cornejo
Manuel Dorado-Moreno
Verónica Duarte
Antonio Durán-Rosal
Felix Fuentes
Dušan Gajić
Brunno Goldstein
David Guijo

César Hervás
Antonio López Herrera
José Ricardo López-Robles
José Antonio Moral Muñoz
Eneko Osaba
Zhisong Pan
Pablo Rozas-Larraondo
Sancho Salcedo
Sónia Sousa
Radu-Daniel Vatavu
Fion Wong
Hui Xue

Workshop on RiskTrack: Analyzing Radicalization in Online Social Networks

Organizers

Javier Torregrosa	Universidad Autónoma de Madrid, Spain
Raúl Lara-Cabrera	Universidad Autónoma de Madrid, Spain
Antonio González Pardo	Universidad Autónoma de Madrid, Spain
Mahmoud Barhamgi	Université Claude Bernard Lyon 1, France

Workshop on Methods for Interpretation of Industrial Event Logs

Organizers

Grzegorz J. Nalepa	AGH University of Science and Technology, Poland
David Camacho	Universidad Autónoma de Madrid, Spain
Edyta Brzychczy	AGH University of Science and Technology, Poland
Roberto Confalonieri	Smart Data Factory, Free University of Bozen-Bolzano, Italy
Martin Atzmueller	Tilburg University, The Netherlands

Workshop on the Interplay Between Human–Computer Interaction and Data Science

Organizers

Cristian Mihăescu	University of Craiova, Romania
Ilkka Kosunen	University of Tallinn, Estonia
Ivan Luković	University of Novi Sad, Serbia

Special Session on Intelligent Techniques for the Analysis of Scientific Articles and Patents

Organizers

Manuel J. Cobo	University of Granada, Spain
Pietro Ducange	eCampus University, Italy

Antonio Gabriel López-Herrera University of Granada, Spain
Enrique Herrera-Viedma University of Granada, Spain

Special Session on Machine Learning for Renewable Energy Applications

Organizers

Sancho Salcedo Sanz Universidad de Alcalá, Spain
Pedro Antonio Gutiérrez University of Cordoba, Spain

Special Session on Evolutionary Computing Methods for Data Mining: Theory and Applications

Organizers

Eneko Osaba TECNALIA Research and Innovaton, Spain
Javier Del Ser University of the Basque Country, Spain
Sancho Salcedo-Sanz University of Alcalá, Spain
Antonio D. Masegosa University of Deusto, Spain

Special Session on Data Selection in Machine Learning

Organizers

Antonio J. Tallón-Ballesteros University of Seville, Spain
Ireneusz Czarnowski Gdynia Maritime University, Poland
Simon James Fong University of Macau, SAR China
Raymond Kwok-Kay Wong University of New South Wales, Australia

Special Session on Feature Learning and Transformation in Deep Neural Networks

Organizers

Richard Hankins University of Manchester, UK
Yao Peng University of Manchester, UK
Qing Tian Nanjing University of Information Science
and Technology, China
Hujun Yin University of Manchester, UK

Special Session on New Models of Bio-inspired Computation for Massive Complex Environments

Organizers

Antonio González Pardo

Universidad Autónoma de Madrid, Spain

Pedro Castillo

Universidad de Granada, Spain

Antonio J. Fernández Leiva

Universidad de Málaga, Spain

Francisco J. Rodríguez

Universidad de Extremadura, Spain

Contents – Part I

Compound Local Binary Pattern and Enhanced Jaya Optimized Extreme Learning Machine for Digital Mammogram Classification	1
<i>Figlu Mohanty, Suvendu Rup, and Bodhisattva Dash</i>	
Support Vector Machine Based Method for High Impedance Fault Diagnosis in Power Distribution Networks	9
<i>K. Moloi, J. A. Jordaan, and Y. Hamam</i>	
Extended Min-Hash Focusing on Intersection Cardinality	17
<i>Hisashi Koga, Satoshi Suzuki, Taiki Itabashi, Gibran Fuentes Pineda, and Takahisa Toda</i>	
Deep-Learning-Based Classification of Rat OCT Images After Intravitreal Injection of ET-1 for Glaucoma Understanding	27
<i>Félix Fuentes-Hurtado, Sandra Morales, Jose M. Mossi, Valery Naranjo, Vadim Fedulov, David Woldbye, Kristian Klemp, Marie Torm, and Michael Larsen</i>	
Finding the Importance of Facial Features in Social Trait Perception	35
<i>Félix Fuentes-Hurtado, Jose Antonio Diego-Mas, Valery Naranjo, and Mariano Alcañiz</i>	
Effective Centralized Trust Management Model for Internet of Things	46
<i>Hela Maddar, Wafa Kammoun, and Habib Youssef</i>	
Knowledge-Based Solution Construction for Evolutionary Minimization of Systemic Risk	58
<i>Krzysztof Michalak</i>	
Handwritten Character Recognition Using Active Semi-supervised Learning	69
<i>Papangkorn Inkeaw, Jakramate Bootkrajang, Teresa Gonçalves, and Jeerayut Chaijaruwanich</i>	
Differential Evolution for Association Rule Mining Using Categorical and Numerical Attributes	79
<i>Iztok Fister Jr., Andres Iglesias, Akemi Galvez, Javier Del Ser, Eneko Osaba, and Iztok Fister</i>	
Predicting Wind Energy Generation with Recurrent Neural Networks	89
<i>Jaume Manero, Javier Béjar, and Ulises Cortés</i>	

Improved Architectural Redesign of MTree Clusterer in the Context of Image Segmentation	99
<i>Marius Andrei Ciurez and Marian Cristian Mihaescu</i>	
Exploring Online Novelty Detection Using First Story Detection Models	107
<i>Fei Wang, Robert J. Ross, and John D. Kelleher</i>	
A Fast Metropolis-Hastings Method for Generating Random Correlation Matrices	117
<i>Irene Córdoba, Gherardo Varando, Concha Bielza, and Pedro Larrañaga</i>	
Novel and Classic Metaheuristics for Tuning a Recommender System for Predicting Student Performance in Online Campus	125
<i>Juan A. Gómez-Pulido, Enrique Cortés-Toro, Arturo Durán-Domínguez, Broderick Crawford, and Ricardo Soto</i>	
General Structure Preserving Network Embedding.	134
<i>Sinan Zhu and Caiyan Jia</i>	
Intelligent Rub-Impact Fault Diagnosis Based on Genetic Algorithm-Based IMF Selection in Ensemble Empirical Mode Decomposition and Diverse Features Models	147
<i>Manjurul Islam, Alexander Prosvirin, and Jong-Myon Kim</i>	
Anomaly Detection in Spatial Layer Models of Autonomous Agents	156
<i>Marie Kiermeier, Sebastian Feld, Thomy Phan, and Claudia Linnhoff-Popien</i>	
Deep Learning-Based Approach for the Semantic Segmentation of Bright Retinal Damage	164
<i>Cristiana Silva, Adrián Colomer, and Valery Naranjo</i>	
Comparison of Local Analysis Strategies for Exudate Detection in Fundus Images	174
<i>Joana Pereira, Adrián Colomer, and Valery Naranjo</i>	
MapReduce Model for Random Forest Algorithm: Experimental Studies	184
<i>Barbara Bobowska and Dariusz Jankowski</i>	
Specifics Analysis of Medical Communities in Social Network Services	195
<i>Artem Lobantsev, Aleksandra Vatian, Natalia Dobrenko, Andrei Stankevich, Anna Kaznacheeva, Vladimir Parfenov, Anatoly Shalyto, and Natalia Gusarova</i>	
PostProcessing in Constrained Role Mining	204
<i>Carlo Blundo, Stelvio Cimato, and Luisa Siniscalchi</i>	

<p>Linguistic Features to Identify Extreme Opinions: An Empirical Study</p> <p style="padding-left: 2em;"><i>Sattam Almatarneh and Pablo Gamallo</i></p>	<p>215</p>
<p>Retinal Image Synthesis for Glaucoma Assessment Using DCGAN and VAE Models</p> <p style="padding-left: 2em;"><i>Andres Diaz-Pinto, Adrián Colomer, Valery Naranjo, Sandra Morales, Yanwu Xu, and Alejandro F. Frangi</i></p>	<p>224</p>
<p>Understanding Learner’s Drop-Out in MOOCs</p> <p style="padding-left: 2em;"><i>Alya Itani, Laurent Brisson, and Serge Garlatti</i></p>	<p>233</p>
<p>Categorical Big Data Processing</p> <p style="padding-left: 2em;"><i>Jaime Salvador-Meneses, Zoila Ruiz-Chavez, and Jose Garcia-Rodriguez</i></p>	<p>245</p>
<p>Spatial-Temporal K Nearest Neighbors Model on MapReduce for Traffic Flow Prediction.</p> <p style="padding-left: 2em;"><i>Anton Agafonov and Alexander Yumaganov</i></p>	<p>253</p>
<p>Exploring the Perceived Usefulness and Attitude Towards Using Tesys e-Learning Platform.</p> <p style="padding-left: 2em;"><i>Paul-Stefan Popescu, Costel Ionascu, and Marian Cristian Mihaescu</i></p>	<p>261</p>
<p>An ELM Based Regression Model for ECG Artifact Minimization from Single Channel EEG</p> <p style="padding-left: 2em;"><i>Chinmayee Dora and Pradyut Kumar Biswal</i></p>	<p>269</p>
<p>Suggesting Cooking Recipes Through Simulation and Bayesian Optimization.</p> <p style="padding-left: 2em;"><i>Eduardo C. Garrido-Merchán and Alejandro Albarca-Molina</i></p>	<p>277</p>
<p>Assessment and Adaption of Pattern Discovery Approaches for Time Series Under the Requirement of Time Warping</p> <p style="padding-left: 2em;"><i>Fabian Kai-Dietrich Noering, Konstantin Jonas, and Frank Klawonn</i></p>	<p>285</p>
<p>Machine Learning Methods Based Preprocessing to Improve Categorical Data Classification</p> <p style="padding-left: 2em;"><i>Zoila Ruiz-Chavez, Jaime Salvador-Meneses, and Jose Garcia-Rodriguez</i></p>	<p>297</p>
<p>Crossover Operator Using Knowledge Transfer for the Firefighter Problem.</p> <p style="padding-left: 2em;"><i>Krzysztof Michalak</i></p>	<p>305</p>
<p>Exploring Coclustering for Serendipity Improvement in Content-Based Recommendation</p> <p style="padding-left: 2em;"><i>Andrei Martins Silva, Fernando Henrique da Silva Costa, Alexandra Katiuska Ramos Diaz, and Sarajane Marques Peres</i></p>	<p>317</p>

Weighted Voting and Meta-Learning for Combining Authorship Attribution Methods	328
<i>Smiljana Petrovic, Ivan Petrovic, Ileana Palesi, and Anthony Calise</i>	
On Application of Learning to Rank for Assets Management: Warehouses Ranking	336
<i>Worapol Alex Pongpech</i>	
Single-Class Bankruptcy Prediction Based on the Data from Annual Reports	344
<i>Peter Drotár, Peter Gnip, Martin Zoričák, and Vladimír Gazda</i>	
Multi-dimensional Bayesian Network Classifier Trees	354
<i>Santiago Gil-Begue, Pedro Larrañaga, and Concha Bielza</i>	
Model Selection in Committees of Evolved Convolutional Neural Networks Using Genetic Algorithms	364
<i>Alejandro Baldominos, Yago Saez, and Pedro Isasi</i>	
Chatbot Theory: A Naïve and Elementary Theory for Dialogue Management	374
<i>Francisco S. Marcondes, José João Almeida, and Paulo Novais</i>	
An Adaptive Anomaly Detection Algorithm for Periodic Data Streams	385
<i>Zirije Hasani, Boro Jakimovski, Goran Velinov, and Margita Kon-Popovska</i>	
Semantic WordRank: Generating Finer Single-Document Summarizations	398
<i>Hao Zhang and Jie Wang</i>	
Exploratory Study of the Effects of Cardiac Murmurs on Electrocardiographic-Signal-Based Biometric Systems	410
<i>M. A. Becerra, C. Duque-Mejía, C. Zapata-Hernández, D. H. Peluffo-Ordóñez, L. Serna-Guarín, Edilson Delgado-Trejos, E. J. Revelo-Fuelagán, and X. P. Blanco Valencia</i>	
Improving the Decision Support in Diagnostic Systems Using Classifier Probability Calibration	419
<i>Xiaowei Kortum, Lorenz Grigull, Urs Muecke, Werner Lechner, and Frank Klawonn</i>	
Applying Tree Ensemble to Detect Anomalies in Real-World Water Composition Dataset	429
<i>Minh Nguyen and Doina Logofătu</i>	

A First Approach to Face Dimensionality Reduction
Through Denoising Autoencoders 439
*Francisco J. Pulgar, Francisco Charte, Antonio J. Rivera,
and María J. del Jesus*

An Approximation to Deep Learning Touristic-Related Time
Series Forecasting 448
*Daniel Trujillo Viedma, Antonio Jesús Rivera Rivas,
Francisco Charte Ojeda, and María José del Jesus Díaz*

CCTV Image Sequence Generation and Modeling Method for Video
Anomaly Detection Using Generative Adversarial Network 457
Wonsup Shin and Sung-Bae Cho

Learning Optimal Q-Function Using Deep Boltzmann Machine
for Reliable Trading of Cryptocurrency 468
Seok-Jun Bu and Sung-Bae Cho

Predicting the Household Power Consumption Using CNN-LSTM
Hybrid Networks 481
Tae-Young Kim and Sung-Bae Cho

Thermal Prediction for Immersion Cooling Data Centers Based
on Recurrent Neural Networks 491
Jaime Pérez, Sergio Pérez, José M. Moya, and Patricia Arroba

Detecting Intrusive Malware with a Hybrid Generative Deep
Learning Model 499
Jin-Young Kim and Sung-Bae Cho

Inferring Temporal Structure from Predictability in Bumblebee
Learning Flight 508
*Stefan Meyer, Olivier J. N. Bertrand, Martin Egelhaaf,
and Barbara Hammer*

Intelligent Wristbands for the Automatic Detection of Emotional States
for the Elderly 520
*Jaime A. Rincon, Angelo Costa, Paulo Novais, Vicente Julian,
and Carlos Carrascosa*

Applying Cost-Sensitive Classifiers with Reinforcement Learning to IDS. 531
*Roberto Blanco, Juan J. Cilla, Samira Briongos, Pedro Malagón,
and José M. Moya*

ATM Fraud Detection Using Outlier Detection 539
Roongtawan Laimek, Natsuda Kaothanthong, and Thepchai Supnithi

Machine Learning for Drugs Prescription	548
<i>P. Silva, A. Rivolli, P. Rocha, F. Correia, and C. Soares</i>	
Intrusion Detection Using Transfer Learning in Machine Learning Classifiers Between Non-cloud and Cloud Datasets	556
<i>Roja Ahmadi, Robert D. Macredie, and Allan Tucker</i>	
Concatenating or Averaging? Hybrid Sentences Representations for Sentiment Analysis.	567
<i>Carlotta Orsenigo, Carlo Vercellis, and Claudia Volpetti</i>	
ALoT: A Time-Series Similarity Measure Based on Alignment of Textures. . .	576
<i>Hasan Oğul</i>	
Intelligent Agents in a Blockchain-Based Electronic Voting System	586
<i>Michał Pawlak, Aneta Poniszewska-Marañda, and Jakub Guziur</i>	
Signal Reconstruction Using Evolvable Recurrent Neural Networks.	594
<i>Nadia Masood Khan and Gul Muhammad Khan</i>	
A Cluster-Based Prototype Reduction for Online Classification.	603
<i>Kemilly Dearo Garcia, André C. P. L. F. de Carvalho, and João Mendes-Moreira</i>	
Reusable Big Data System for Industrial Data Mining - A Case Study on Anomaly Detection in Chemical Plants	611
<i>Reuben Borrison, Benjamin Klöpper, Moncef Chioua, Marcel Dix, and Barbara Sprick</i>	
Unsupervised Domain Adaptation for Human Activity Recognition	623
<i>Paulo Barbosa, Kemilly Dearo Garcia, João Mendes-Moreira, and André C. P. L. F. de Carvalho</i>	
Data Set Partitioning in Evolutionary Instance Selection.	631
<i>Miroslaw Kordos, Łukasz Czepielik, and Marcin Blachnik</i>	
Identification of Individual Glandular Regions Using LCWT and Machine Learning Techniques	642
<i>José Gabriel García, Adrián Colomer, Valery Naranjo, Francisco Peñaranda, and M. Á. Sales</i>	
Improving Time Series Prediction via Modification of Dynamic Weighted Majority in Ensemble Learning.	651
<i>Marek Łóderer, Peter Pavlík, and Viera Rozinajová</i>	

Generalized Low-Computational Cost Laplacian Eigenmaps	661
<i>J. A. Salazar-Castro, D. F. Peña, C. Basante, C. Ortega, L. Cruz-Cruz, J. Revelo-Fuelagán, X. P. Blanco-Valencia, G. Castellanos-Domínguez, and D. H. Peluffo-Ordóñez</i>	
Optimally Selected Minimal Learning Machine.	670
<i>Átilla N. Maia, Madson L. D. Dias, João P. P. Gomes, and Ajalmar R. da Rocha Neto</i>	
Neural Collaborative Filtering: Hybrid Recommendation Algorithm with Content Information and Implicit Feedback	679
<i>Li Ji, Guangyan Lin, and Huobin Tan</i>	
Overlap-Based Undersampling for Improving Imbalanced Data Classification	689
<i>Pattaramon Vuttipittayamongkol, Eyad Elyan, Andrei Petrovski, and Chrisina Jayne</i>	
Predicting Online Review Scores Across Reviewer Categories	698
<i>Michela Fazzolari, Marinella Petrocchi, and Angelo Spognardi</i>	
Improving SeNA-CNN by Automating Task Recognition.	711
<i>Abel Zacarias and Luis A. Alexandre</i>	
Communication Skills Personal Trainer Based on Viola-Jones Object Detection Algorithm	722
<i>Álvaro Pardo Pertierra, Ana B. Gil González, Javier Teira Lafuente, and Ana de Luis Reboredo</i>	
Optimizing Meta-heuristics for the Time-Dependent TSP Applied to Air Travels.	730
<i>Diogo Duque, José Aleixo Cruz, Henrique Lopes Cardoso, and Eugénio Oliveira</i>	
Compositional Stochastic Average Gradient for Machine Learning and Related Applications	740
<i>Tsung-Yu Hsieh, Yasser EL-Manzalawy, Yiwei Sun, and Vasant Honavar</i>	
Instance-Based Stacked Generalization for Transfer Learning	753
<i>Yassine Baghoussi and João Mendes-Moreira</i>	
Combined Classifier Based on Quantized Subspace Class Distribution.	761
<i>Paweł Ksieniewicz</i>	
A Framework for Form Applications that Use Machine Learning	773
<i>Guilherme Aguiar and Patrícia Vilain</i>	

CGLAD: Using GLAD in Crowdsourced Large Datasets 783
Enrique G. Rodrigo, Juan A. Aledo, and Jose A. Gamez

Extending Independent Component Analysis for Event Detection
on Online Social Media 792
Hoang Long Nguyen and Jason J. Jung

Framework for the Training of Deep Neural Networks in TensorFlow
Using Metaheuristics 801
*Julián Muñoz-Ordóñez, Carlos Cobos, Martha Mendoza,
Enrique Herrera-Viedma, Francisco Herrera, and Siham Tabik*

New Fuzzy Singleton Distance Measurement by Convolution 812
Rodrigo Naranjo and Matilde Santos

Peak Alpha Based Neurofeedback Training Within Survival Shooter Game. . . 821
*Radu AbuRas, Gabriel Turcu, Ilkka Kosunen,
and Marian Cristian Mihaescu*

Taking e-Assessment Quizzes - A Case Study with an SVD Based
Recommender System 829
*Oana Maria Teodorescu, Paul Stefan Popescu,
and Marian Cristian Mihaescu*

Towards Complex Features: Competitive Receptive Fields
in Unsupervised Deep Networks 838
Richard Hankins, Yao Peng, and Hujun Yin

Deep Neural Networks with Markov Random Field Models
for Image Classification 849
Yao Peng, Menyu Liu, and Hujun Yin

Author Index 861

Contents – Part II

Workshop on RiskTrack: Analyzing Radicalization in Online Social Networks

Ontology Uses for Radicalisation Detection on Social Networks	3
<i>Mahmoud Barhamgi, Raúl Lara-Cabrera, Djamal Benslimane, and David Camacho</i>	
Measuring Extremism: Validating an Alt-Right Twitter Accounts Dataset.	9
<i>Joshua Thorburn, Javier Torregrosa, and Ángel Panizo</i>	
RiskTrack: Assessing the Risk of Jihadi Radicalization on Twitter Using Linguistic Factors	15
<i>Javier Torregrosa and Ángel Panizo</i>	
On Detecting Online Radicalization Using Natural Language Processing	21
<i>Mourad Oussalah, F. Faroughian, and Panos Kostakos</i>	

Workshop on Methods for Interpretation of Industrial Event Logs

Automated, Nomenclature Based Data Point Selection for Industrial Event Log Generation	31
<i>Wolfgang Koehler and Yanguo Jing</i>	
Monitoring Equipment Operation Through Model and Event Discovery.	41
<i>Sławomir Nowaczyk, Anita Sant'Anna, Ece Calikus, and Yuantao Fan</i>	
Creation of an Event Log from a Low-Level Machinery Monitoring System for Process Mining Purposes	54
<i>Edyta Brzychczy and Agnieszka Trzcionkowska</i>	
Causal Rules Detection in Streams of Unlabeled, Mixed Type Values with Finit Domains	64
<i>Szymon Bobek and Kamil Jurek</i>	
On the Opportunities for Using Mobile Devices for Activity Monitoring and Understanding in Mining Applications.	75
<i>Grzegorz J. Nalepa, Edyta Brzychczy, and Szymon Bobek</i>	
A Taxonomy for Combining Activity Recognition and Process Discovery in Industrial Environments	84
<i>Felix Mannhardt, Riccardo Bovo, Manuel Fradinho Oliveira, and Simon Julier</i>	

Mining Attributed Interaction Networks on Industrial Event Logs	94
<i>Martin Atzmueller and Benjamin Kloepper</i>	
Special Session on Intelligent Techniques for the Analysis of Scientific Articles and Patents	
Evidence-Based Systematic Literature Reviews in the Cloud	105
<i>Iván Ruiz-Rube, Tatiana Person, José Miguel Mota, Juan Manuel Dodero, and Ángel Rafael González-Toro</i>	
Bibliometric Network Analysis to Identify the Intellectual Structure and Evolution of the Big Data Research Field	113
<i>J. R. López-Robles, J. R. Otegi-Olaso, I. Porto Gomez, N. K. Gamboa-Rosales, H. Gamboa-Rosales, and H. Robles-Berumen</i>	
A New Approach for Implicit Citation Extraction	121
<i>Chaker Jebari, Manuel Jesús Cobo, and Enrique Herrera-Viedma</i>	
Constructing Bibliometric Networks from Spanish Doctoral Theses	130
<i>V. Duarte-Martínez, A. G. López-Herrera, and M. J. Cobo</i>	
Measuring the Impact of the International Relationships of the Andalusian Universities Using Dimensions Database	138
<i>P. García-Sánchez and M. J. Cobo</i>	
Special Session on Machine Learning for Renewable Energy Applications	
Gaussian Process Kernels for Support Vector Regression in Wind Energy Prediction	147
<i>Victor de la Pompa, Alejandro Catalina, and José R. Dorronsoro</i>	
Studying the Effect of Measured Solar Power on Evolutionary Multi-objective Prediction Intervals	155
<i>R. Martín-Vázquez, J. Huertas-Tato, R. Aler, and I. M. Galván</i>	
Merging ELMs with Satellite Data and Clear-Sky Models for Effective Solar Radiation Estimation	163
<i>L. Cornejo-Bueno, C. Casanova-Mateo, J. Sanz-Justo, and S. Salcedo-Sanz</i>	
Distribution-Based Discretisation and Ordinal Classification Applied to Wave Height Prediction	171
<i>David Guijo-Rubio, Antonio M. Durán-Rosal, Antonio M. Gómez-Orellana, Pedro A. Gutiérrez, and César Hervás-Martínez</i>	

Wind Power Ramp Events Ordinal Prediction Using Minimum Complexity
Echo State Networks 180
*M. Dorado-Moreno, P. A. Gutiérrez, S. Salcedo-Sanz, L. Prieto,
and C. Hervás-Martínez*

**Special Session on Evolutionary Computing Methods for Data Mining:
Theory and Applications**

GELAB - A Matlab Toolbox for Grammatical Evolution 191
Muhammad Adil Raja and Conor Ryan

Bat Algorithm Swarm Robotics Approach for Dual Non-cooperative Search
with Self-centered Mode 201
*Patricia Suárez, Akemi Gálvez, Iztok Fister, Iztok Fister Jr.,
Eneko Osaba, Javier Del Ser, and Andrés Iglesias*

Hospital Admission and Risk Assessment Associated to Exposure
of Fungal Bioaerosols at a Municipal Landfill Using Statistical Models 210
*W. B. Morgado Gamero, Dayana Agudelo-Castañeda,
Margarita Castillo Ramirez, Martha Mendoza Hernandez,
Heidy Posso Mendoza, Alexander Parody, and Amelec Vilorio*

Special Session on Data Selection in Machine Learning

Novelty Detection Using Elliptical Fuzzy Clustering in a Reproducing
Kernel Hilbert Space 221
*Maria Kazachuk, Mikhail Petrovskiy, Igor Mashechkin,
and Oleg Gorohov*

Semi-supervised Learning to Reduce Data Needs of Indoor
Positioning Models 233
Maciej Grzenda

Different Approaches of Data and Attribute Selection
on Headache Disorder 241
Svetlana Simić, Zorana Banković, Dragan Simić, and Svetislav D. Simić

A Study of Fuzzy Clustering to Archetypal Analysis 250
Gonçalo Sousa Mendes and Susana Nascimento

Bare Bones Fireworks Algorithm for Medical Image Compression 262
*Eva Tuba, Raka Jovanovic, Marko Beko, Antonio J. Tallón-Ballesteros,
and Milan Tuba*

EMnGA: Entropy Measure and Genetic Algorithms Based Method
for Heterogeneous Ensembles Selection 271
Souad Taleb Zouggar and Abdelkader Adla

Feature Selection and Interpretable Feature Transformation: A Preliminary Study on Feature Engineering for Classification Algorithms	280
<i>Antonio J. Tallón-Ballesteros, Milan Tuba, Bing Xue, and Takako Hashimoto</i>	
Data Pre-processing to Apply Multiple Imputation Techniques: A Case Study on Real-World Census Data.	288
<i>Zoila Ruiz-Chavez, Jaime Salvador-Meneses, Jose Garcia-Rodriguez, and Antonio J. Tallón-Ballesteros</i>	
Imbalanced Data Classification Based on Feature Selection Techniques	296
<i>Paweł Ksieniewicz and Michał Woźniak</i>	
Special Session on New Models of Bio-inspired Computation for Massive Complex Environments	
Design of Japanese Tree Frog Algorithm for Community Finding Problems	307
<i>Antonio Gonzalez-Pardo and David Camacho</i>	
An Artificial Bee Colony Algorithm for Optimizing the Design of Sensor Networks.	316
<i>Ángel Panizo, Gema Bello-Orgaz, Mercedes Carnero, José Hernández, Mabel Sánchez, and David Camacho</i>	
Community Detection in Weighted Directed Networks Using Nature-Inspired Heuristics	325
<i>Eneko Osaba, Javier Del Ser, David Camacho, Akemi Galvez, Andres Iglesias, Iztok Fister Jr., and Iztok Fister</i>	
A Metaheuristic Approach for the α -separator Problem	336
<i>Sergio Pérez-Peló, Jesús Sánchez-Oro, and Abraham Duarte</i>	
Author Index	345



Compound Local Binary Pattern and Enhanced Jaya Optimized Extreme Learning Machine for Digital Mammogram Classification

Figlu Mohanty^(✉), Suwendu Rup, and Bodhisattva Dash

Image and Video Processing Lab, International Institute of Information Technology,
Bhubaneswar, India

figlu92@gmail.com, suwendu@iiit-bh.ac.in, bdash.fac@gmail.com

Abstract. The fatality rate due to breast cancer still continues to remain high across the world and women are the frequent sufferers of this cancer. Mammography is one of the powerful imaging modalities to detect and diagnose cancer at its early stage effectively. A computer-aided diagnosis (CAD) system is a potential tool which analyses the mammographic images to reach a correct decision. The present work aims at developing a CAD framework which can classify the mammograms accurately. This work has primarily four stages. First, contrast limited adaptive histogram equalization (CLAHE) is used for pre-processing. Second, feature extraction is realized using compound local binary pattern (CLBP) followed by principal component analysis (PCA) for feature reduction. Finally, an enhanced Jaya-based extreme learning machine is utilized to classify the mammograms as normal or abnormal, and further, benign or malignant. The success rate in terms of classification accuracy achieves 100% and 99.48% for MIAS and DDSM datasets, respectively.

Keywords: Mammography · Computer-aided diagnosis
Compound local binary pattern · Jaya algorithm
Extreme learning machine

1 Introduction

According to the statistics on cancer, the incidence and mortality scenario of breast cancer across the world is getting intensified day by day. The world health organization [8] makes an estimation of 21 million cancer cases by the year 2030 which was only 12.7 million in 2008. This high figure of cancer cases can be lessened by detecting cancer at its early stage. So, it is of utmost importance to design an efficient detection and diagnosis tool in order to reduce the mortality rates among women and men. Mammography is the most effective and reliable method to detect the abnormalities in the breast in an early stage. With the advent of digital image processing, pattern recognition, and artificial

intelligence, a CAD system provides a ‘second opinion’ to the radiologists to improve their diagnosis. However, designing a CAD system with high efficiency and automated analysis remains a challenging task. The proposed work aims at designing an efficient CAD system which includes the following contributions: (1) The first step deals with the image pre-processing which includes cropping the mammograms to obtain the desired region of interest (ROI) followed by contrast limited adaptive histogram equalization (CLAHE) to improve the quality of low-contrast images. (2) In the next step, feature extraction using compound local binary pattern (CLBP) is utilized followed by feature reduction using the principal component analysis (PCA). (3) Finally, classification of mammograms into normal and abnormal, and then benign and malignant using enhanced Jaya-based extreme learning machine (EJaya-ELM).

The structure of the article is as follows: Sect. 2 discusses briefly the recent works done to design effective CAD systems. Section 3 elaborates the proposed CAD model. Analyses of the results obtained with the proposed model are presented in Sect. 4. Section 5 finally concludes the article.

2 Related Work

Reyad *et al.* [11] proposed a CAD system which extracts features using discrete wavelet transform (DWT), contourlet transform (CT), and local binary pattern (LBP). A comparison analysis of the features is made in terms of classification accuracy employing support vector machine (SVM) as the classifier and an improved accuracy of 98.63% is obtained on DDSM dataset. Berbar [3] proposed three hybrid methods, namely, Wavelet-CT1, Wavelet-CT2 and ST-GLCM for extracting the features followed by SVM as a classifier on DDSM. Xie *et al.* [14] presented a CAD model using extreme learning machine-based classifier and validated on MIAS and DDSM. With the extracted multidimensional features followed by feature selection using a combination of SVM and ELM. Bajaj *et al.* [2] proposed a novel approach using bi-dimensional empirical mode decomposition (BEMD) and least-square SVM for mammogram classification on MIAS dataset. Chithra *et al.* [5] used wavelet entropy and ensemble classifiers using k-nearest neighbor (k-NN) and SVM. Singh *et al.* [12] proposed a wavelet-based center-symmetric local binary pattern technique to extract the features from the mammograms. Later, they used SVM-recursive feature elimination (SVM-RFE) for feature selection and random forest for classification which yields an accuracy of 97.3% on MIAS dataset.

3 Proposed Methodology

In the present work, initially, the desired ROIs are generated from the original mammograms using simple cropping approach. In case of abnormal mammograms, the cropping is done using the given ground truth information about the position and radius of the abnormal regions. However, in the case of normal

mammograms, cropping is done on any arbitrary location of the breast area of normal mammogram to get the ROI. The further modules are performed on the extracted ROIs. A detailed illustration of the proposed CAD model is given in Fig. 1.

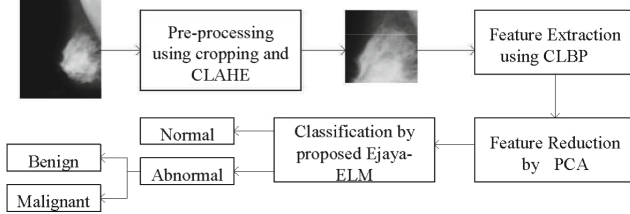


Fig. 1. Block diagram of the proposed CAD model

3.1 Pre-processing Using CLAHE

Pre-processing of the ROIs is considered to be a vital step as it influences the rest of the modules in a CAD system. In most of the observations, the images collected from the datasets are of low contrast, so it is required to enhance the contrast of such images. Hence, in the present work, contrast limited adaptive histogram equalization (CLAHE) [9] is utilized to improve the quality of the low-contrast images.

3.2 Feature Extraction Using Compound Local Binary Pattern (CLBP)

The local binary pattern (LBP) operator ignores the magnitude of the difference between the center pixel value and its neighboring pixel values resulting in inconsistent codes. So, to overcome the issues of LBP, CLBP is introduced. In CLBP, a 2-bit code is utilized to encode the local texture information of an image, where the first bit indicates the sign of the difference between the center pixel and corresponding neighboring pixel values and the second bit represents the magnitude of the difference with respect to a threshold value T_{avg} [1]. The term T_{avg} is the mean value of the difference between the center pixel and its corresponding neighbors in the local neighborhood window. If P_n is one of the neighboring pixels, and P_c is the center pixel, then the mathematical expression of the 2-bit code is presented as follows:

$$s(i_n, i_m) = \begin{cases} 00 & P_n - P_c < 0 \text{ and } |P_n - P_c| \leq T_{avg} \\ 01 & P_n - P_c < 0 \text{ and } |P_n - P_c| > T_{avg} \\ 10 & P_n - P_c \geq 0 \text{ and } |P_n - P_c| \leq T_{avg} \\ 11 & \text{otherwise} \end{cases} \quad (1)$$

This $C(P_n, P_c)$ generates 16-bit codes for the eight neighbors which is again split into two 8-bit codes, one for the diagonal neighbors and another for non-diagonal neighbors. Then, two different histograms are plotted for the two different groups and are combined for generating the CLBP.

3.3 Feature Reduction Using PCA

The number of features obtained from the feature extraction module is quite large and to prevent the ‘curse of dimensionality’ issue, it is needed to lessen the size of the feature vector which also makes the task of classifier simple. In addition, out of all the features, some of them are not relevant. To obtain the set of relevant features, PCA technique is employed. PCA generates a reduced set of features by transforming the high-dimensional data to a set of data having low dimension retaining maximum variance of the original data. This transforms produces a set of linearly uncorrelated data which are called as principal components (PCs). A deep insight about PCA can be referred in [4, 6].

3.4 Classification Using Proposed EJaya-ELM

The two major problems of traditional ELM are: (1) ELM requires a large number of hidden neurons, and (2) ELM causes ill-conditioned hidden layer output matrix which degrades the generalization performance [15, 16]. So to overcome these issues, many investigations have been undertaken using the principles of evolutionary and swarm intelligence algorithms to fine-tune the hidden node parameters of ELM. In order to enhance the generalization performance of ELM, in the present work, a recently proposed optimization algorithm, namely, Jaya algorithm [10] is utilized to optimize the hidden node parameters (input weights and bias) of ELM. Prior to classification module, the whole dataset is divided into training, validation, and testing set using 5-fold stratified cross-validation (SCV) to prevent the over-fitting problem. The flowchart of the working principle of the proposed CAD model is represented in Fig. 2. It is worth specifying here that the EJaya-ELM explores the global optimum solution considering the classification accuracy as the fitness function. The prime objective of EJaya-ELM is to improve the accuracy and to limit the parameters of ELM (input weights and bias) in a definite range to improve the convergence of ELM. The steps involved in the proposed EJaya-ELM are given as follows:

1. Start the process with random initialization of candidate solution in the population so that each solution has a set of input weights and bias as

$$C_j = [W_{11}^i, W_{12}^i, \dots, W_{1l}^i, W_{21}^i, W_{22}^i, \dots, W_{2l}^i, \dots, W_{h1}^i, W_{h2}^i, W_{hl}^i, b_1, b_2, \dots, b_h] \quad (2)$$

The weights W and bias b are initialized in a range of $[-1, 1]$.

2. For each of the candidate solutions, determine the fitness value (classification accuracy) and the output weights. The fitness value is calculated on the validation set in order to prevent the over-fitting issue.

- Find the best and worst solution based on the fitness value and modify them using Jaya algorithm.

$$C'_j(k) = C_j(k) + r_1(k)(C_{best}(k) - |C_j(k)|) - r_2(k)(C_{worst}(k) - |C_j(k)|) \quad (3)$$

where $C'_j(k)$ indicate the updated value of the solution. $C_{best}(k)$ and $C_{worst}(k)$ denote the best and worst solution in k^{th} iteration. $r_1(k)$ and $r_2(k)$ are two randomly generated numbers in a range of $[0, 1]$.

- Generate the new solutions as

$$C_j(k+1) = \begin{cases} C'_j(k) & f(C'_j(k)) > f(C_j(k)) \\ C_j(k) & \text{otherwise} \end{cases} \quad (4)$$

where $f(C'_j(k))$ and $f(C_j(k))$ denote the fitness value and the respective updated value of j^{th} solution in k^{th} iteration.

- Find the out-of-bound cases in the new solution and limit them in a range of $[-1, 1]$ as

$$C_j(k+1) = \begin{cases} -1 & C_j(k+1) < -1 \\ 1 & C_j(k+1) > 1 \end{cases} \quad (5)$$

- Repeat steps (2)–(5) for a specified number of iterations. Finally, the optimized input weights and biases are attained and utilized on the testing data set to obtain the overall performance of the proposed model.

4 Experimental Results and Analysis

The proposed CAD model is experimented on two standard benchmark datasets, namely, MIAS [13] and DDSM [7]. A total of 314 and 1500 images are collected from MIAS and DDSM, respectively. The collected images are first classified into normal or abnormal, and further, benign or malignant using the proposed CAD system. The performance of the proposed model is evaluated in terms of different performance metrics, namely, accuracy, sensitivity, specificity, area under curve (AUC), and receiver operating characteristics (ROC) curve.

Prior to the feature extraction module, ROIs are segmented from the unnecessary background regions using cropping. Using the ground truth information regarding the coordinates of the abnormalities in the images, ROIs of size 256×256 are generated. After cropping, the ROIs are pre-processed using CLAHE to enhance the contrast. Then, CLBP technique is applied on the extracted ROIs to obtain the feature matrix. Applying CLBP, a feature matrix of size $s \times F$ is obtained, where s and F indicate the number of ROIs and the number of generated features, respectively. In this work, 512 number of features (F) are generated from CLBP which is quite large. So, to reduce the size of the

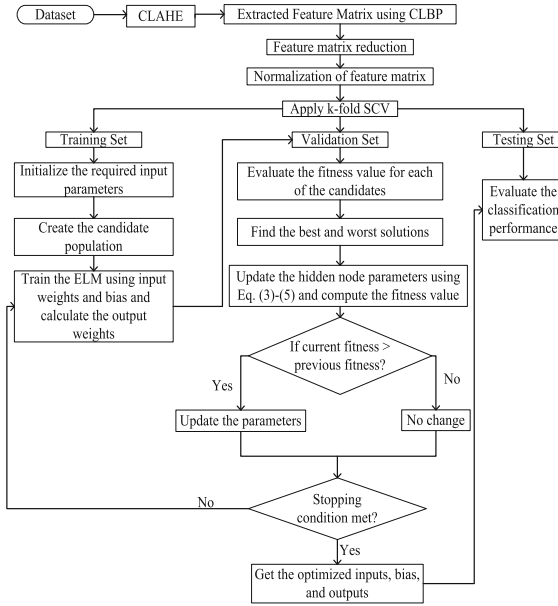


Fig. 2. Flowchart of the proposed CLAHE + CLBP + PCA + EJaya-ELM approach

feature vector and make the classification simpler, PCA is utilized which reduces the number of features from 512 to 14 preserving 99% of the variance of the original data. The reduced features are passed to the proposed enhanced Jaya-ELM classifier for classifying the mammograms as normal or abnormal followed by benign or malignant.

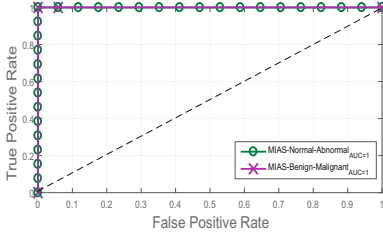
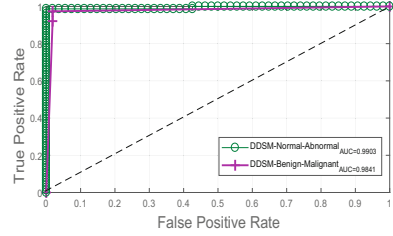
Table 1. Performance measures obtained by the proposed EJaya-based ELM along with SVM and Jaya-ELM classifiers; N-Normal, A-Abnormal, B-Benign, M-Malignant

Dataset	Performance measures	CLAHE + CLBP + PCA + SVM		CLAHE + CLBP + PCA + Jaya-ELM		CLAHE + CLBP + PCA + EJaya-ELM	
		N-A	B-M	N-A	B-M	N-A	B-M
MIAS	Sensitivity	0.9928	0.9891	0.9931	0.9899	1	1
	Specificity	0.9905	0.9844	0.9901	0.9881	1	1
	Accuracy (%)	99.15	98.45	99.05	98.86	100	100
DDSM	Sensitivity	0.9821	0.9733	0.9912	0.9871	0.9945	0.9886
	Specificity	0.9888	0.9812	0.9908	0.9792	0.9912	0.9869
	Accuracy (%)	98.23	97.28	99.17	98.22	99.48	98.61

Table 1 depicts the various performance results attained with the proposed CAD model. For more clarity, the proposed model is also compared with traditional SVM and Jaya-ELM classifier. From the table, it can be noticed that the

Table 2. Comparison with some of the other existing CAD schemes

Reference	Proposed scheme	Classification accuracy (%)
[11]	Statistical Features + LBP + SVM	98.63 (DDSM)
[3]	Wavelet-CT1 + Wavelet-CT2 + ST-GLCM + SVM	98.69 (DDSM)
[14]	Multidimensional features + SVM + ELM	96.02 (MIAS), 95.73 (DDSM)
[2]	BEMD + SVM	95 (MIAS)
[12]	WCS-LBP + SVM-RFE + Random Forest	97.3 (MIAS)
Proposed	CLAHE + CLBP + PCA + EJaya-ELM	100 (MIAS), 99.48 (DDSM)

**Fig. 3.** ROC curve for MIAS dataset**Fig. 4.** ROC curve for DDSM dataset

highest accuracy achieved for MIAS dataset is 100% in both normal-abnormal and benign-malignant classifications. Similarly, for DDSM dataset, an accuracy of 99.48% and 98.61% is achieved for normal-abnormal, and benign-malignant classification, respectively. Additionally, the ROC graphs generated by the proposed classifier are plotted in Figs. 3 and 4 showing the corresponding values of AUC for MIAS and DDSM datasets, respectively. Since the proposed approach gives 100% accuracy on MIAS dataset, it can be noticed that the ROC is a perfect curve having $AUC=1$. Further, to add more justification, the proposed CAD model is compared with five recently developed CAD schemes. The comparison against the other schemes is made in terms of classification accuracy and is depicted in Table 2. From the table, it can be observed that the proposed model gives superior results than that of the other competent schemes.

5 Conclusion

In the present work, an enhanced CAD system has been proposed for breast cancer classification in digital mammograms. Initially, CLAHE is used to enhance the low-contrast images. Then, CLBP is employed to extract the texture features followed by a feature reduction module using PCA. The reduced feature set is then passed through EJaya-ELM-based classifier to classify the mammograms. The proposed model has been experimented on two benchmark datasets, namely, MIAS and DDSM. Furthermore, the performance of the proposed model has been compared with five recent schemes and it has been noticed that the proposed model with only 14 features achieves improved results over the competent

schemes. The high success rate with respect to the accuracy of the proposed scheme helps radiologists to make an accurate diagnosis decision to reduce unnecessary biopsies.

References

1. Ahmed, F., Hossain, E., Bari, A., Hossen, M.S.: Compound local binary pattern (CLBP) for rotation invariant texture classification. *Int. J. Comput. Appl.* **33**(6), 5–10 (2011)
2. Bajaj, V., Pawar, M., Meena, V.K., Kumar, M., Sengur, A., Guo, Y.: Computer-aided diagnosis of breast cancer using bi-dimensional empirical mode decomposition. *Neural Comput. Appl.* 1–9 (2017)
3. Berbar, M.A.: Hybrid methods for feature extraction for breast masses classification. *Egypt. Inf. J.* **19**(1), 63–73 (2018)
4. Bishop, C.: *Pattern Recognition and Machine Learning*. Springer, New York (2006)
5. Chithra Devi, M., Audithan, S.: Analysis of different types of entropy measures for breast cancer diagnosis using ensemble classification. *Biomed. Res.* **28**, 3182–3186 (2017)
6. Duda, R.O., Hart, P.E., Stork, D.G.: *Pattern Classification*. Wiley, Hoboken (2012)
7. Heath, M., Bowyer, K., Kopans, D., Moore, R., Kegelmeyer, W.P.: The digital database for screening mammography. In: *Proceedings of the 5th International Workshop on Digital Mammography*, pp. 212–218. Medical Physics Publishing (2000)
8. World Health Organization: *Burden: mortality, morbidity and risk factors. Global status report on noncommunicable diseases 2011* (2010)
9. Pizer, S.M., Johnston, R.E., Ericksen, J.P., Yankaskas, B.C., Muller, K.E.: Contrast-limited adaptive histogram equalization: speed and effectiveness. In: *1990, Proceedings of the First Conference on Visualization in Biomedical Computing*, pp. 337–345. IEEE (1990)
10. Rao, R.: Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *Int. J. Ind. Eng. Comput.* **7**(1), 19–34 (2016)
11. Reyad, Y.A., Berbar, M.A., Hussain, M.: Comparison of statistical, LBP, and multi-resolution analysis features for breast mass classification. *J. Med. Syst.* **38**(9), 100 (2014)
12. Singh, V.P., Srivastava, S., Srivastava, R.: Effective mammogram classification based on center symmetric-LBP features in wavelet domain using random forests. *Technol. Health Care* **25**(4), 709–727 (2017)
13. Suckling, J., et al.: The mammographic image analysis society digital mammogram database. In: *Excerpta Medica. International Congress Series*, vol. 1069, pp. 375–378 (1994)
14. Xie, W., Li, Y., Ma, Y.: Breast mass classification in digital mammography based on extreme learning machine. *Neurocomputing* **173**, 930–941 (2016)
15. Zhao, G., Shen, Z., Miao, C., Man, Z.: On improving the conditioning of extreme learning machine: a linear case. In: *2009 7th International Conference on Information, Communications and Signal Processing, ICICS 2009*, pp. 1–5. IEEE (2009)
16. Zhu, Q.Y., Qin, A.K., Suganthan, P.N., Huang, G.B.: Evolutionary extreme learning machine. *Pattern Recognit.* **38**(10), 1759–1763 (2005)



Support Vector Machine Based Method for High Impedance Fault Diagnosis in Power Distribution Networks

K. Moloji^(✉), J. A. Jordaan, and Y. Hamam

Tshwane University of Technology, Pretoria, South Africa
moloikt023@gmail.com, {JordaanJA, Hamama}@tut.ac.za

Abstract. The detection of high impedance faults (HIFs) on a power distribution system has been a subject of concern for many decades. This poses a very unique challenge to the protection engineers, as it seems to be invisible to be detected by conventional protection schemes. The major concern about HIFs is that they pose a safety risk, as these faults are associated with arcing which may be dangerous for the surroundings. In this work, we propose a technique, which uses feature extraction, classification and a locating algorithm. Discrete wavelet transform (DWT) is used to extract meaningful information, support vector machine (SVM) is used as a classifier and a support vector regression (SVR) scheme is used as a fault location estimator. The technique is tested on a network of a power utility.

Keywords: High impedance fault · Fault detection · Fault location

1 Introduction

1.1 Background and Motivation

A power system plays an important role in the development of any society. The availability of electricity with high quality and minimum environmental contraventions still remains as one of the main objectives of the power utilities. Over the years conventional protection has shown great results for low impedance faults (LIFs) but has been proven to have limitations in terms of HIFs detection. Generally, a HIF occurs when an energized conductor breaks and falls onto the ground resulting in a restricted path for the fault current, thus making its detection difficult [1]. This makes conventional protection schemes not suitable for this category of faults [2].

A review of many approaches concerning HIF detection from classical to heuristically have been presented in [1] and this is a useful guideline in tracking the progress made in the research space. Amongst other methods is the mathematical morphology (MM) based filter [3], discrete wavelet transform (DWT), fuzzy logic (FL) [4] and neural network (NN) [5]. There has been significant developments made in an attempt to develop a suitable algorithm over the years, however much still needs to be done to improve the current technologies. In this paper, the main goal is to develop an intelligent protection method, which can accurately diagnose HIFs, and distinguish these faults from other power system events.

2 Proposed Method for High Impedance Fault Detection

In the proposed algorithm, the current magnitude is received at various sections along the distribution line and processed using DWT. The features such as the standard deviation and the mean deviation are extracted. Subsequently these features are used to train the SVM for classification and detection, and subsequently support vector regression (SVR) for location. Figure 1 shows the detailed flow chart of the proposed method.

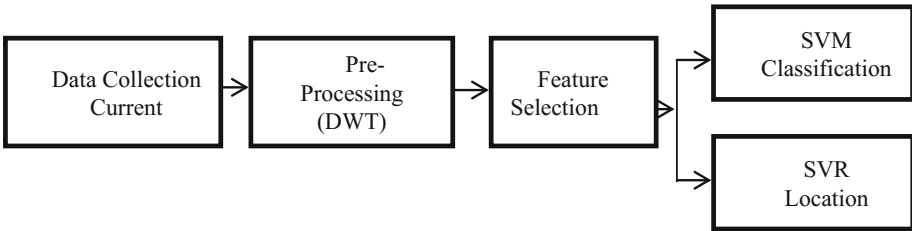


Fig. 1. The proposed method for HIF diagnosis

3 Case Study

3.1 Power Distribution Network

In order to effectively study the proposed method for a HIF detection scheme, a representation of the reduced power distribution network of a power utility is considered. Eskom is a power utility in South Africa. The network is shown in Fig. 2 and it comprises of the Moutse, Siyabuswa T-off and Zoefontein T-off with line lengths of 7.5 km, 6.35 km and 12.5 km respectively at 22 kV. The conductor type has a positive sequence resistance of 0.186 (Ω/km), positive sequence reactance of 0.418 (Ω/km), zero sequence resistance of 0.413 (Ω/km) and zero sequence reactance of 0.413 (Ω/km). The load at Moutse, Siyabuswa and Zoefontein are 6.8 MVA, 7.2 MVA and 4.6 MVA respectively.

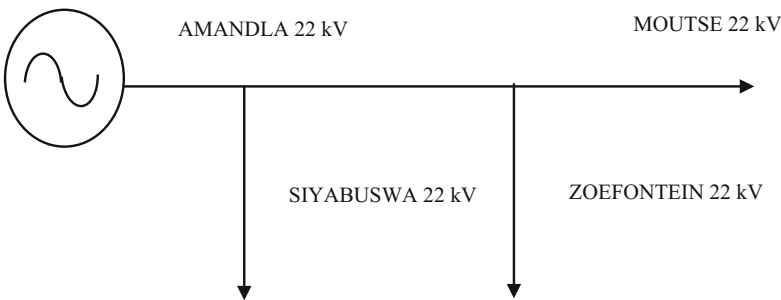


Fig. 2. Eskom's reduced distribution power system

The source has a short circuit power, short circuit current, X/R ratio, X0/X1 ratio and R0/R1 ratio rating of 61.9 MVA, 16.3 kA, 133.1, 5.52, and 12.12 respectively. The Emmanuel arc based model is used in this work [2].

4 Signal Processing and Feature Extraction

4.1 Wavelet Transform (WT)

Wavelet Transform (WT), has in recent years gained the attention of most researchers mainly because of its ability to decompose a signal into both its frequency and time information. WTs have proven to be an appropriate signal processing tool for transient, non-variant and non-stationary signals. There are two most common known wavelets, namely the Discrete Wavelet Transform (DWT) and the Continues Wavelet Transform (CWT), these have shown great advantage over Fourier Transforms (FT) [6, 7]. The mathematical expression of CWT is defined as:

$$CWT(a, b) = \frac{1}{a} \int_{-\infty}^{\infty} x(t) \psi\left(\frac{t-b}{a}\right) dt \quad (1)$$

where a is the scaling factor and b is the translation factor. The DWT is expressed as:

$$x(t) = \sum_k c_j(k) \phi(t-k) + \sum_{j-1} \sum d_j(k) \psi(2^{-j}t-k) \quad (2)$$

where c_j is an approximation coefficient, d_j is a detail coefficient, ϕ is a scaling factor and ψ is a wavelet function. The symlet 4 (sym 4), sym 6 and sym 8 have been widely employed for image processing [8]. In this paper we investigate the efficiency of the symlet mother wavelet for HIFs detection in an electrical power system. The features selected in this work for a decomposed signal using symlet wavelet at level 3 are presented in Table 1. The signals of HIF, and other disturbances such as load switching and capacitor switching are observed using a MATLAB/SIMULINK model on the Eskom power system.

Table 1. DWT accuracy comparison

Mother wavelet	Standard deviation	Mean absolute deviation
Sym 4	1.523	1.025
Sym 6	1.667	1.140
Sym 8	1058	1.062

5 Fault Classification and Detection

5.1 Support Vector Machine (SVM)

Support vector machines (SVM) were initially developed to solve classification problems in statistical learning theory and structural risk minimization (SRM) [8]. In applying SVM algorithms, the input vectors are mapped into a high dimension space to find the optimal hyperplane which separates the two data classes. A hyperplane is said to be optimal if and only if it separates the vectors without error and if the distance between the closest vector to the hyperplane is maximal. Computing the hyperplane is equivalent to solving the optimization problem given as:

$$\min \frac{1}{2} |w|^2 + C \left(\sum_{i=1}^l \zeta_i \right), \quad (3)$$

Subject to $y_i(w \cdot x_i + b) \geq 1 - \zeta_i, \zeta_i \geq 0, \forall i$

where x_i is the i th example and y_i is the class label which is either +1 or -1. The problem is solved using its dual form [8]:

$$\max L_D = \sum_i a_i - \frac{1}{2} \sum_{i,j} a_i a_j y_i y_j (x_i^T x_j) \quad (4)$$

Subject to $0 \leq a_i \leq C, \forall i, \sum_i i a_i y_i = 0$

In a case where the data is not linearly separable, different kernel functions may be employed to transform the input data to a higher dimensional space.

5.2 Support Vector Regression (SVR) for Fault Estimation

The Support vector regression (SVR) algorithm uses the same principles as SVM for classification, with only a few minor differences. The applied SVM for a regression problem is done by introducing a loss function. The loss function is to be modified in such a way that it includes the distance measured. The most commonly known loss functions are, the Quadratic, Huber, and c-insensitive loss function [9]. The SVR problem may be expressed as finding the solution of the quadratic programming problem which can be solved by introducing a dual set of variables $a_i a_i^*$ and constructing a Lagrange function [8]. The kernel functions are consequently used to compute the dot product in the high dimensional space in order to achieve optimal mapping. The SVR solution may be expressed as:

$$\begin{aligned} \max W(a_i a_i^*) &= \frac{1}{2} \sum_{i,j=1}^l (a_i - a_i^*) (a_j - a_j^*) K(x_i - x_j^*) + \sum_{i,j=1}^l (a_i^* - a_i) y_i \\ &\quad - \frac{1}{2C} \sum_{i=1}^l a_i^2 - a_i^{*2} \end{aligned} \quad (5)$$

where $K(x_i - x_j^*)$ is the kernel function [8].

6 Results and Discussion

The results of the statistical comparison of three mother wavelets are shown in Table 1. The sym 4 mother wavelet showed best signal analysis results compared to both sym 6 and sym 8. The Standard deviation and absolute mean deviation were calculated. These two features were used to train the SVM and SVR schemes.

The typical HIF current signal is shown in Fig. 3, the detail coefficient of the current signal at Level 1, 2 and 3 are presented from Figs. 4, 5 and 6 respectively. The original and the approximation signals are presented in Fig. 7. In order to evaluate the performance of the locating scheme, the mean square error (MSE) is determined. The response of the SVR scheme for the fault at 25% of the Moutse line is presented in Fig. 8. The MSE results are presented in Table 2.

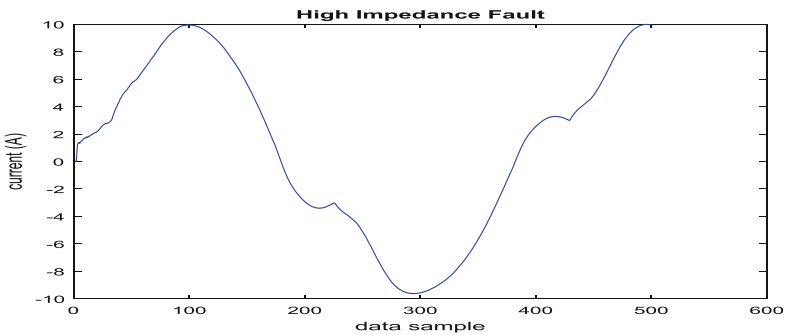


Fig. 3. HIF current signal

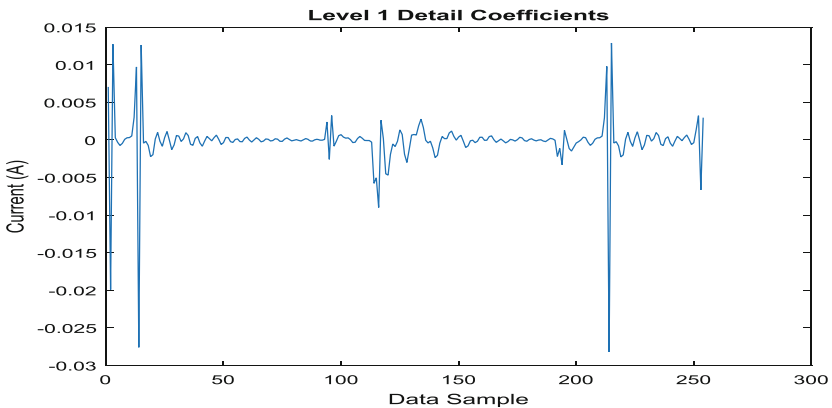


Fig. 4. Level 1 detail coefficient using sym 4

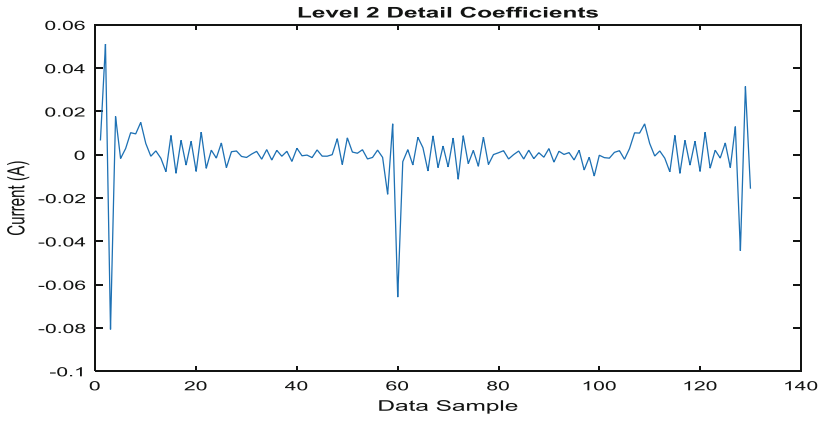


Fig. 5. Level 2 detail coefficient using sym 4

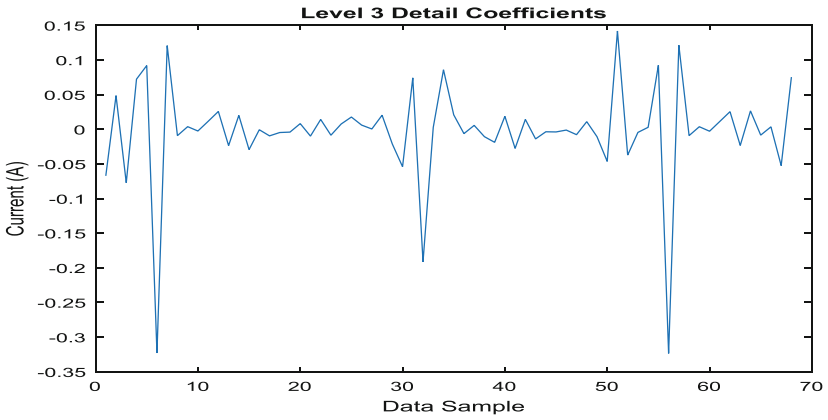


Fig. 6. Level 3 detail coefficient using sym 4

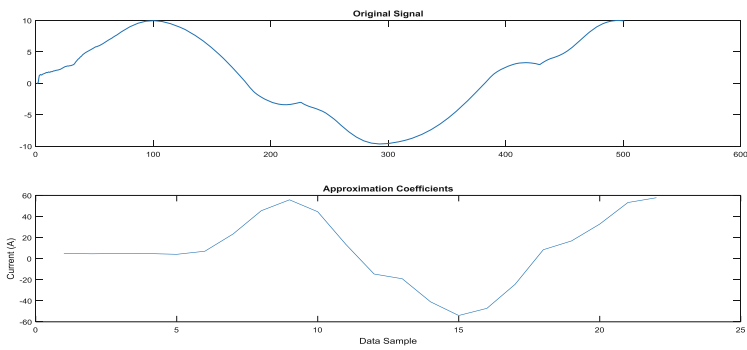


Fig. 7. HIF original and approximated signals at level 3 using sym 4

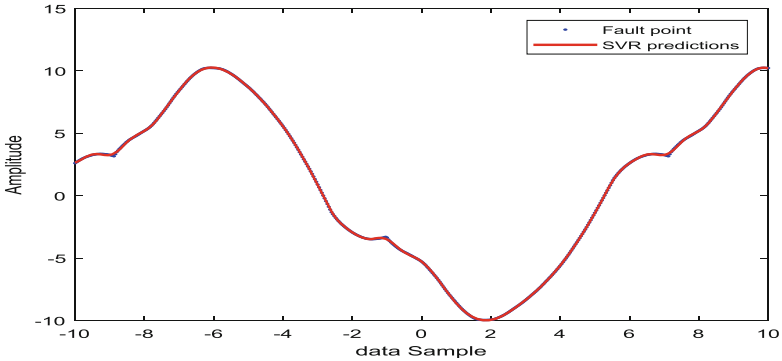


Fig. 8. SVR fault estimation at 25% of the Moutse 22 kV line

Table 2. SVR scheme accuracy for MSE determination

Line	Locator scheme	MSE	Distance in (%)
Moutse	SVR	2.55E-04	25
Siyabuswa	SVR	3.58E-06	25
Zoefontein	SVR	4.25E-3	25

Table 3. SVM fault and event type detection accuracy

Classifier	Event type	Class	Accuracy (%)
SVM	HIF	+1	100
SVM	Capacitor switching	-1	100
SVM	Load switching	-1	100

Table 4. Scheme comparison

Scheme	Method	Accuracy (%)
Proposed scheme	DWT-SVM-SVR	98.8
Scheme in [3]	MM	99
Scheme in [4]	DWT-FFA	94.19

The accuracy of the results detecting various events in a power system is presented in Table 3, +1 indicates the presence of a fault and -1 is an indication of no fault in the system.

A total number 1000 HIF cases were simulated, 500 capacitor switching cases and 500 load switching cases were simulated in MATLAB. The cross validation method was employed to train and test the SVM algorithm. The scheme accuracy is presented in Table 4, comparing the proposed and the cited scheme.

7 Conclusion

In this work, a HIF detection protection scheme is proposed and it is extensively tested on a real Eskom power system with different operating conditions. The proposed method uses DWT for signal decomposition, SVM for fault classification and SVR for fault estimation. The method reveals its suitability for application as it shows promising results. The main contribution of the paper is that this method may be used in any distribution network with any configuration. However the classifier may have to be retrained to suit the application. Although the results are based on simulation work due to the scarcity of actual data, this method can be used and tested on practical environment. The future work will entail experimental data and other machine learning techniques for optimal and suitable application to eradicate this challenge.

References

1. Sedighizadeh, M., Rezazadeh, A., Elkalashy, N.I.: Approches in high impedance fault detection - a chronological review. *Adv. Electr. Comput. Eng.* **10**(3), 114–128 (2010)
2. Sekar, K., Mohanty, N.K.: Combined mathematics morphology and data mining based high impedance fault detection. In: 1st International Conference on Power Engineering, Computing and Control. VIT University, Chennai Campus, 2–4 March 2017
3. Gautam, S., Brahma, S.M.: Detection of high impedance fault in power distribution systems using mathematical morphology. *IEEE Trans. Power Syst.* **28**, 1226–1364 (2013)
4. Banejad, M., Ijadi, H.: High impedance fault detection using discrete wavelet transform and fuzzy function approximation. *J. AI Data Min.* **2**(2) 149–158 (2014)
5. Vijayachandram, G., Mathew, B.K.: Arcing fault detection in feeder networks using discrete wavelet transform and artificial neural network. *Int. J. Emerg. Sci. Eng.* **1**(10), 93–102 (2013)
6. Keerthana, G., Umayal, S.P.: Analysis of faults in transmission lines with the help of discrete wavelet transform. In: The International Conference on Current Research in Engineering Science and Technology (ICCREST) (2016)
7. Jadhav, A., Thakur, K.: Fault detection and classification in transmission lines based on wavelet transform. *Int. J. Sci. Eng. Res.* **3**(5), 14–19 (2015)
8. Magagula, X.G., Hamam, Y., Jordaan, J.A., Yusuff, A.A.: A fault classification and localization method in a power distribution network. In: IEEE Africon, Cape Town, South Africa (2017)
9. Gunn, S.R.: Support Vector Machine For Classification and Regression. University of Southampton, Southampton (1998)



Extended Min-Hash Focusing on Intersection Cardinality

Hisashi Koga^{1(✉)}, Satoshi Suzuki¹, Taiki Itabashi¹, Gibran Fuentes Pineda²,
and Takahisa Toda¹

¹ University of Electro-Communications, Tokyo, Japan
koga@sd.is.uec.ac.jp

² Universidad Nacional Autonoma de Mexico, Mexico City, Mexico

Abstract. Min-Hash is a reputable hashing technique which realizes set similarity search. Min-Hash assumes the Jaccard similarity $\frac{|A \cap B|}{|A \cup B|}$ as the similarity measure between two sets A and B . Accordingly, Min-Hash is not optimal for applications which would like to measure the set similarity with the intersection cardinality $|A \cap B|$, since the Jaccard similarity decreases irrespective of $|A \cap B|$, as the gap between $|A|$ and $|B|$ becomes larger. This paper shows that, by modifying Min-Hash slightly, we can effectively settle the above difficulty inherent to Min-Hash. Our method is shown to be valid both by theoretical analysis and with experiments.

Keywords: Set similarity search · Min-Hash · Intersection

1 Introduction

Nowadays, many pattern recognition tasks are addressed by modeling the data as finite sets. For example, the famous bag-of-words model represents a document as a set of words. Then, similar documents can be retrieved by searching similar word sets. Thus, the set similarity search becomes more and more important.

This paper studies a set similarity search which, given a query set A , seeks such sets B for which the intersection cardinality $|A \cap B|$ is large. We name this type of similarity search as the *intersection search*. This intersection search is significant for applications which need to search sets including a subset similar to A from the database. One example of such applications is the detection of documents which plagiarize a short query document. Here, after representing the short query document as the query word set A , the intersection search can find the documents plagiarizing the query document by searching documents B for which the number of common words $|A \cap B|$ is large. We remark that $|A|$ denotes the cardinality of A , i.e., the number of elements in A .

One natural approach to perform the intersection search fast is to utilize Min-Hash [1]. Min-Hash is a hashing technique that realizes fast set similarity search. Suppose that A and B are two sets. Min-Hash prepares a randomized hash function h for which the probability of hash collision between A and B equals

their Jaccard similarity. That is, $P[h(A) = h(B)] = \frac{|A \cap B|}{|A \cup B|}$. Min-Hash makes it possible to search similar sets on hash tables, when the similarity between two sets is measured with their Jaccard similarity. Min-Hash has been applied to various applications such as near duplicate detection of documents and images [3], object discovery from images [8], graph summarization and so on.

As the numerator of Jaccard similarity is $|A \cap B|$, Min-Hash can process the intersection search well, if every set in the database has almost the same cardinality. By contrast, Min-Hash does not work effectively for the intersection search, when the set cardinalities are diverse in the database, because $P[h(A) = h(B)]$ becomes very small without regard to $|A \cap B|$, when the gap between $|A|$ and $|B|$ is large. Let $\alpha > 1$. If $|B|$ is α times as large as $|A|$, $P[h(A) = h(B)] = \frac{|A \cap B|}{|A \cup B|}$ never exceeds $\frac{1}{\alpha}$ irrespective of $|A \cap B|$, since $|A \cap B| \leq |A|$ and $|A \cup B| \geq |B| = \alpha|A|$.

This paper proposes a new algorithm for the intersection search. Our algorithm is easily derived by modifying Min-Hash slightly. Nonetheless, it successfully settles the above difficulty inherent to Min-Hash. Our algorithm is especially suitable in the situation where the cardinality of query set is less than or as large as those of the sets in the database. Note this condition is usually satisfied for the plagiarism detection mentioned above. Specifically, our algorithm randomly divides a set B in the database into distinct subsets of equal size and register all the subsets of B to the hash table. Then, if the query set A collides with some subset of B , the original set B is considered to collide with A . Some theoretical analysis explains why our method works successfully. The effectiveness of our method is also shown experimentally.

2 Min-Hash and Related Works

We review the original Min-Hash and its recent extensions in the literature. Min-Hash [1] is a randomized hash algorithm for set similarity search based on Jaccard similarity. Let D be the number of element kinds and $E = \{1, 2, \dots, D\}$. Min-Hash constructs a hash function h by choosing a random permutation π of E . For a set A , the hash value $h(A)$ is defined as $\min_{i \in A} \pi(i)$, i.e., the smallest element ID in A after permuting it with π .

Min-Hash uses multiple hash functions and multiple hash tables to execute the similarity search as follows. First, Min-Hash prepares r hash functions h_1, h_2, \dots, h_r which differ in the choice of π and makes a function g which outputs the concatenation of the r hash values. That is, $g(A) = (h_1(A), h_2(A), \dots, h_r(A))$. Then, Min-Hash generates l instances of g : g_1, g_2, \dots, g_l and l hash tables HT_1, HT_2, \dots, HT_l . For $1 \leq i \leq l$, on the i -th hash table HT_i , a set B in the database is stored in a hash bucket whose index value equals $g_i(B)$. We describe this bucket as $b(g_i(B))$.

For a query set A , Min-Hash searches sets similar to A as follows. First, it computes l hash values $g_1(A), g_2(A), \dots, g_l(A)$. Let R_i be the sets kept in the hash bucket $b(g_i(A))$ on HT_i . $R_1 \cup R_2 \cup \dots \cup R_l$ forms the candidate sets similar to A . Then, the candidates are filtered by computing the exact Jaccard similarity.

In the above discussion, r and l are parameters which should be adequately specified by the user.

Recently, Min-Hash has been extended in various ways. Chum *et al.* [3] extended Min-Hash to treat multisets. One permutation hashing [6] generates multiple hash values per hash function to reduce the overhead of hash value computation. b -bit minwise hashing [5] reduces the space complexity of Min-Hash drastically by using only the lowest b bits of hash values. Sim-Min-Hash [10] enables the comparison between two sets of real-valued vectors by considering the similarity between a pair of vectors when their quantized labels match. Christiani *et al.* [2] realized a hash-based set similarity search based on Braun-Blanquet similarity $\frac{|A \cap B|}{\max\{|A|, |B|\}}$.

Min-Hash has evolved uniquely in the area of image processing so as to capture the spatial relation among elements. Partition Min-Hash (PMH) [4] divides an image into rectangular regions and computes a hash value for each region. PMH is similar to our work, since the feature set of an image is partitioned into subsets and the hash values are computed for all the subsets. However, the way to divide a set is totally different. PMH divides any set into the same number of subsets, whereas the number of subsets becomes proportional to the set cardinality in our method. In addition, PMH assumes that the elements are distributed in the two-dimensional space, while our method processes general sets without such restriction.

Shrivastava *et al.* [9] have recently proposed Asymmetric Min-Hash for the intersection search. Though their method is novel, it must enlarge a set represented as a D -dimensional binary vector to a $D + 2M$ dimensional binary vector by inserting some virtual elements, where M is the maximum set cardinality. Asymmetric Min-Hash needs to use different hash functions for the query and for the database. As far as we know, our method is the first approach to solve the intersection search without increasing the set size. Moreover, our method is symmetric and uses the same hash function both for the query and the database.

3 Proposed Method

This section explains our proposed algorithm for the intersection search. Our method is best suited for applications in which the size of query set is smaller than the size of sets in the database.

Min-Hash is not very good at the intersection search, because $P[h(A) = h(B)] = \frac{|A \cap B|}{|A \cup B|}$ decreases without regard to $|A \cap B|$, as the size difference $||A| - |B||$ becomes larger. We mitigate this property via the next simple strategy: We divide a set into disjoint subsets of equal size and consider that all the hash values of the subsets belong to the original set.

We process a set B in the database as follows. Let T be a tuning parameter which prescribes the subset size.

Step 1: If $|B| \geq 2T$, B is randomly separated into disjoint subsets of size T . When $|B|$ is not a multiple of T , the remaining $|B| \% T$ elements are

appended to one of the subsets arbitrarily. As the result, we have $\lfloor \frac{|B|}{T} \rfloor$ subsets $B_1, B_2, \dots, B_{\lfloor \frac{|B|}{T} \rfloor}$.

Step 2: Let h be a hash function for the standard Min-Hash. We compute the hash values $h(B_1), h(B_2), \dots, h(B_{\lfloor \frac{|B|}{T} \rfloor})$ and regard all of them as the hash values of B .

Regarding the query set A , we also cut A into disjoint subsets of size T and compute all the hash values of subsets. On the other hand, if we know from the domain knowledge that the query set size is small, an option not to cut A into subsets should be permitted. This condition holds for many applications of the intersection search, e.g., the problem to find sets containing the query set approximately.

Then, when A and B have at least one common hash value, a hash collision is considered to happen between them.

3.1 Theoretical Analysis

This subsection discusses the difference between our method and Min-Hash in terms of the hash collision probability between A and B . For simplicity, assume that $|A|$ and $|B|$ are both multiples of T . Without loss of generality, suppose that B consists α times as many elements as A , where $\alpha \geq 1$. Thus, $|A| = cT$ and $|B| = \alpha cT$, where c is a constant natural number. Let $k = |A \cap B|$. Note that $0 \leq k \leq cT$.

First, the hash collision probability in the standard Min-Hash is derived as

$$P[h(A) = h(B)] = \frac{|A \cap B|}{|A \cup B|} = \frac{k}{(\alpha + 1)cT - k}, \quad (1)$$

which converges to 0 as α grows large. Therefore, if $|A|$ and $|B|$ are quite different, the hash collision is difficult to occur in Min-Hash.

Next, let us consider our method. Our method divides A and B into their subsets A_1, A_2, \dots, A_c and $B_1, B_2, \dots, B_{\alpha c}$ respectively. Assuming that the elements in $A \cap B$ are uniformly distributed into the subsets, the expected number of common elements between A_i and B_j ($1 \leq i \leq c$ and $1 \leq j \leq \alpha c$) becomes $\frac{k}{\alpha c^2}$. Thus, $P[h(A_i) = h(B_j)] = \frac{|A_i \cap B_j|}{|A_i \cup B_j|} = \frac{\frac{k}{\alpha c^2}}{T + T - \frac{k}{\alpha c^2}} = \frac{k}{2\alpha c^2 T - k}$. Since this formula converges to 0 as α increases, one may consider, at first glance, that the situation is not different from Min-Hash at all.

The hash collision probability in our method is the probability that A and B share at least one common hash value. Hereafter, we describe this probability as $P_{col}(A, B)$. $P_{col}(A, B)$ is derived by subtracting the probability that none of $c \times \alpha c = \alpha c^2$ subset pairs take the same hash value from 1. Therefore,

$$P_{col}(A, B) = 1 - \left(1 - \frac{k}{2\alpha c^2 T - k}\right)^{\alpha c^2}. \quad (2)$$

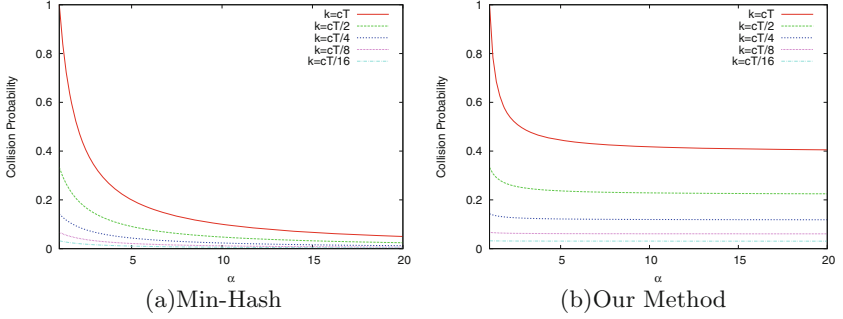


Fig. 1. Collision probability

Although Eq. (2) decreases monotonically regarding α , it does not converge to 0 when α goes to infinity unlike Min-Hash. Interestingly, the convergence value of collision probability depends on k as shown in Eq. (3).

$$\begin{aligned}
 & \lim_{\alpha \rightarrow \infty} 1 - \left(1 - \frac{k}{2\alpha c^2 T - k}\right)^{\alpha c^2} \\
 &= \lim_{\alpha \rightarrow \infty} 1 - \left(\left(1 - \frac{1}{\frac{2T}{k}\alpha c^2 - 1}\right)^{\frac{2T}{k}\alpha c^2 - 1}\right)^{\frac{\alpha c^2}{\frac{2T}{k}\alpha c^2 - 1}} \\
 &= \lim_{\alpha \rightarrow \infty} 1 - \left(\frac{1}{e}\right)^{\frac{\alpha c^2}{\frac{2T}{k}\alpha c^2 - 1}} = 1 - e^{-\frac{k}{2T}}. \tag{3}
 \end{aligned}$$

Moreover, Eq. (3) becomes larger, as k increases. Thus, our method guarantees that, given two sets B and C , the magnitude relation between $P_{col}(A, B)$ and $P_{col}(A, C)$ is consistent with that between $|A \cap B|$ and $|A \cap C|$, when $|B|$ and $|C|$ are both large to some degree. Since this property does not hold for Min-Hash, our method is more suitable for the intersection search than Min-Hash.

From a practical viewpoint, the convergence speed is important and desired to be fast with respect to α . Hence, we examine it by simulation. Figure 1(a) and (b) plot the collision probabilities of Min-Hash in Eq. (1) and of our method in Eq. (2) respectively for various α , when $c = 1$. Each graph consists of multiple curves which correspond to different values of k , i.e., the number of common elements. Figure 1 states that, when k differs, the collision probability surely converges to a different value in our method, whereas it always converges to 0 in Min-Hash. Significantly, the convergence speed of our method is so fast that the probability converges before α reaches 5 for any k . In particular, the curves look like horizontal lines when $k \leq \frac{cT}{4}$, which means that the collision probability is not influenced by α . Note that this is ideal for the intersection search.

3.2 Retrieval System

The retrieval system is implemented just in the same way as Min-Hash mentioned in Sect. 2. We prepare l hash functions from g_1 to g_l each of which consists of r instances of h and create l hash tables.

The time complexity of similarity search becomes proportional to $\left\lfloor \frac{|A|}{T} \right\rfloor$ in our method, since it has to visit $\left\lfloor \frac{|A|}{T} \right\rfloor$ buckets per hash table. Therefore, our method becomes slower relatively to Min-Hash as $|A|$ increases. Thus, our method favors such applications in which the query set is smaller in size than the sets in the database, so that we may cut only the latter into subsets as mentioned at the beginning of Sect. 3.

4 Experiments

This section compares our method with the standard Min-Hash experimentally. The experiments are conducted on one synthetic dataset and on one real dataset.

4.1 Results on Synthetic Dataset

Here, we use a synthetic dataset to evaluate our algorithm quantitatively. Below, we explain how to synthesize the dataset.

1. First, we make a collection of 100 query sets $\{Q_1, Q_2, \dots, Q_{100}\}$. Each query set is made by choosing 200 distinct elements from 20000 kinds of elements. Thus, $|Q_i| = 200$ for $1 \leq i \leq 100$.
2. For each Q_i , we construct a group of 50 similar sets which share many common elements with Q_i . Let this group of sets be S^i and the j -th set in S^i be S_j^i where $1 \leq j \leq 50$. Each set in S^i contains a different number of common elements with Q_i . Specifically, $|Q_i \cap S_j^i| = 30 + 2j$ for $1 \leq j \leq 50$. In addition, to make the set cardinality diverse, $|S_j^i|$ is randomly chosen from the figure pool $\{400, 600, 800, 1000, 1200, 1400\}$. The whole database is derived by merging all S^i for $1 \leq i \leq 100$. Thus, the database keeps $100 \times 50 = 5000$ sets in total.

After mapping the whole database onto the hash tables either by Min-Hash or our method, we operate the similarity search with specifying Q_i as a query set for $1 \leq i \leq 100$. The number of hash tables l is set to 500. The division threshold $T = 200$ in our method, which assures that the query sets are never divided.

To examine how much the intersection size correlates with the collision probability, we create a list which arranges the sets in S^i in descending order of $|Q_i \cap S_j^i|$. We also make another list which places the sets in S^i in descending order of the number of hash collisions. Then, we compute the Spearman's Rank-Order Correlation between these two lists. A high correlation coefficient indicates that the intersection search is managed nicely.

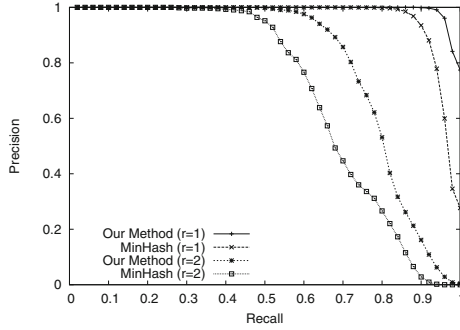


Fig. 2. Precision-recall curve for synthetic dataset

The average correlation coefficient becomes 0.977 for our method and 0.688 for Min-Hash, when $r = 1$. Here, the average is taken over all the query sets. Recall that r denotes a number of hash functions composing g . Thus, our method achieves a higher correlation coefficient than Min-Hash and, therefore, is more suitable for the intersection search than Min-Hash.

We also show the precision-recall curve for $r = 1$ and $r = 2$ in Fig. 2. This curve is depicted in the following manner. For the query set Q_i , the sets in S^{Q_i} serves as the correct answer. Then, we generate an ordered list in which all the sets in the database are arranged in descending order of the number of hash collisions with Q_i . We can acquire a pair of (Recall, Precision), if we fix some front part of this list and examine how many correct answers occupy it. Figure 2 plots the average precision values against specific recall values, where the average is taken over all the query sets. This figure concludes that our method surely outperforms Min-Hash.

4.2 Results on Real Dataset

Next, we evaluate the two algorithms by apply them to the task of automatic annotation to visual objects discovered from real image dataset. We first retrieve tagged images from Flickr by specifying 10 query keywords of famous landmarks including “Taj Mahal” and “Tokyo Skytree”. By gathering 200 images per query, the dataset consists of 2000 tagged images. After scanning the whole dataset, we grasp a set of tags that appear at least once. By removing tags that appear in more than 30% or in less than 0.5% of the images, there remain 394 kinds of tags.

Then, we extract visual objects from the image dataset with the object discovery method in [8] and each of them is represented as a group of visual words, i.e., quantized local features. For instance, see Fig. 3. This figure depicts a toy example consisting of 3 images I_1, I_2, I_3 . Here, the “house” object is modeled by 3 visual words v_1, v_2, v_3 which correspond to a roof, a wall and a chimney.

As for visual words, we extract the MSER local feature points [7] expressed as 128 dimensional vectors out of all the images. The visual words are determined

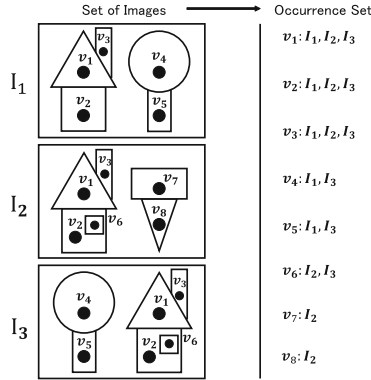


Fig. 3. Occurrence image sets of visual words

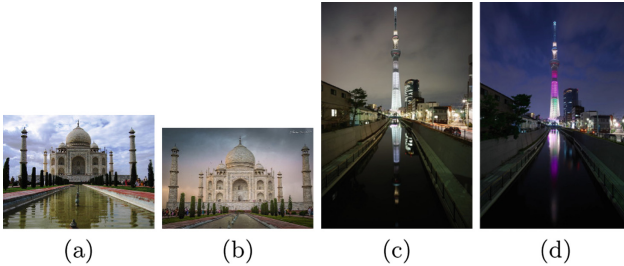


Fig. 4. Images including visual objects

by clustering them with k -means, where $k = 32768$. For a visual word v , a set of images containing v is called the “occurrence image set” of v and described as OI_v . Hereafter, the term “occurrence image set” is abbreviated as “OI”. The right half of Fig. 3 shows the OIs of 8 visual words. We can define the OI of a tag t in the same way and describe it as OI_t . Usually, the OI size becomes larger for tags than visual words, since tags express semantics and correspond to multiple visual appearances. In our setting, the average OI size for visual words is about 10, whereas the OI size for a tag often goes beyond 100.

Next, we annotate a discovered visual object O with tags which tend to co-occur in the same image with some visual words in O . We can find such tags by seeking tags t for which $|OI_v \cap OI_t|$ becomes large for some $v \in O$. Hence, the intersection search needs to be performed by specifying each OI_v as a query, where $r = 3$ and $l = 250$. Since the OIs for visual words are sufficiently smaller than that for tags, it is hard for Min-Hash to manage this intersection search. In our method, we divide only the OIs for tags which serve as the sets in the database. We do not segment the OIs for visual words which play as queries, because their sizes are small. The division threshold T is set to 11, the average OI size for visual words.

We evaluate the result of annotation to visual objects qualitatively. Table 1 shows the annotation results to two visual objects corresponding to the Taj Mahal and the Tokyo Skytree. This table summarizes the tags discovered by Min-Hash and our method. One row lists the tag name, its OI size, the total number of hash collisions and the total number of common elements between the tag and the visual words in the object model.

The first visual object consists of 3 visual words whose OI sizes are 6,7 and 12. Figure 4(a) and (b) show 2 images containing this object. Min-Hash discovers two tags “life” and “love” which seem unrelated to the object. Though the OIs of these two tags have a moderate number of common elements with the visual words only, they are favored by Min-Hash, since their OIs are small. By contrast, our method could discover useful tags such as “tajmahal”, “taj”, “India” and “agra”, the city name where the tajmahal is located. They have many common elements with the visual words, so that our method tailored to the intersection search could retrieve them. Note that Min-Hash misses them, since their OIs are too large. Thus, our method helps the annotation task effectively. We remark that all the tags discovered by Min-Hash are also discovered by our method.

Table 1. Annotation to objects “tajmahal” and “Tokyo Skytree”

Object	Tajmahal				Tokyo Skytree			
	Tag	OI	#colli	#common	Tag	OI	#colli	#common
Min-Hash	Love	17	4	4	Alpha	10	5	11
	Life	10	2	2	Fuji	16	3	2
Proposed	Tajmahal	200	6	25	Tokyo	488	1	19
	India	170	4	25	Skytree	144	6	15
	Agra	155	2	23	Sony	58	6	12
	Travel	157	1	10	Night	202	1	11
	Taj	47	3	5	Zeiss	22	3	10
	Sky	66	2	5	Landmark	60	1	3
	Landscape	67	1	3	Fujifilm	58	1	3
	Street	40	1	2	Cloud	31	3	2
	Tourist	24	1	1				

The second visual object consists of 5 visual words whose OI sizes are 4, 4, 6, 10 and 10. Figure 4(c) and (d) show 2 images which contain this object. Our method succeeds in discovering the tags such as “tokyo” “skytree”, “night” and “landmark” which describe the object well. Since their OIs are too large, they are missed by Min-Hash, despite the intersection cardinality is not small.

5 Conclusion

This paper studied the intersection search which, given a query set A , purposes to seek sets B from the database such that $|A \cap B|$ is large. Min-Hash cannot

handle the intersection search well, since the hash collision probability comes close to 0 irrespective of $|A \cap B|$, as the gap between $|A|$ and $|B|$ enlarges. By contrast, we show that a slight modification of Min-Hash makes it suitable for the intersection search. Our modification is indeed simple: We have only to divide a set into subsets of equal length T and associate their hash values with the original set. Theoretically, our method achieves that the collision probability converges to a constant value determined by $|A \cap B|$, which is a nice property for the intersection search. The experiment on a synthetic dataset confirmed that $|A \cap B|$ correlates with the number of hash collisions surely in our method. Moreover, our method performed more effectively than Min-Hash in the task of automatic annotation to visual objects discovered from images.

Acknowledgments. This work was supported by JSPS KAKENHI Grant Number JP18K11311, 2018.

References

1. Broder, A.Z.: On the resemblance and containment of documents. In: Proceedings of Compression and Complexity of Sequences 1997, pp. 21–29 (1997). <https://doi.org/10.1109/SEQUEN.1997.666900>
2. Christiani, T., Pagh, R.: Set similarity search beyond minhash. In: Proceedings of ACM STOC, pp. 1094–1107 (2017)
3. Chum, O., Philbin, J., Zisserman, A.: Near duplicate image detection: min-hash and tf-idf weighting. In: Proceedings of BMVC, pp. 50.1–50.10 (2008)
4. Lee, D.C., Ke, Q., Isard, M.: Partition min-hash for partial duplicate image discovery. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6311, pp. 648–662. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15549-9_47
5. Li, P., König, C.: b-bit minwise hashing. In: Proceedings of WWW, pp. 671–680 (2010). <https://doi.org/10.1145/1772690.1772759>
6. Li, P., Owen, A.B., Zhang, C.: One permutation hashing. In: Proceedings of 26th NIPS 2012, pp. 3122–3130 (2012)
7. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide baseline stereo from maximally stable extremal regions. In: Proceedings of BMVC, pp. 36.1–36.10 (2002)
8. Pineda, G.F., Koga, H., Watanabe, T.: Scalable object discovery: a hash-based approach to clustering co-occurring visual words. *IEICE Trans.* **94–D**(10), 2024–2035 (2011)
9. Shrivastava, A., Li, P.: Asymmetric minwise hashing for indexing binary inner products and set containment. In: Proceedings of WWW, pp. 981–991 (2015). <https://doi.org/10.1145/2736277.2741285>
10. Zhao, W.L., Jégou, H., Gravier, G.: Sim-Min-Hash: an efficient matching technique for linking large image collections. In: Proceedings of ACM MM, pp. 577–580 (2013). <https://doi.org/10.1145/2502081.2502152>



Deep-Learning-Based Classification of Rat OCT Images After Intravitreal Injection of ET-1 for Glaucoma Understanding

Félix Fuentes-Hurtado¹(✉), Sandra Morales¹, Jose M. Mossi¹, Valery Naranjo¹, Vadim Fedulov², David Woldbye³, Kristian Klemp³, Marie Torm⁴, and Michael Larsen⁴

¹ Instituto de Investigación e Innovación en Bioingeniería, I3B, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain
flifuehu@upv.es

² Department of Neuroscience and Pharmacology, University of Copenhagen, Copenhagen, Denmark

³ Laboratory of Neural Plasticity, Department of Neuroscience, University of Copenhagen, Copenhagen, Denmark

⁴ Department of Ophthalmology, Rigshospitalet-Glostrup, Glostrup, Copenhagen, Denmark

Abstract. Optical coherence tomography (OCT) is a useful technique to monitor retinal damage. We present an automatic method to accurately classify rodent OCT images in healthy and pathological (before and after 14 days of intravitreal injection of Endothelin-1, respectively) making use of the DenseNet-201 architecture fine-tuned and a customized top-model. We validated the performance of the method on 1912 OCT images yielding promising results ($AUC = 0.99 \pm 0.01$ in a $P = 15$ leave-P-out cross-validation). Besides, we also compared the results of the fine-tuned network with those achieved training the network from scratch, obtaining some interesting insights. The presented method poses a step forward in understanding pathological rodent OCT retinal images, as at the moment there is no known discriminating characteristic which allows classifying this type of images accurately. The result of this work is a very accurate and robust automatic method to distinguish between healthy and a rodent model of glaucoma, which is the backbone of future works dealing with human OCT images.

Keywords: Optical coherence tomography · Deep-learning
Glaucoma

1 Introduction

Optical coherence tomography (OCT) has become the most important retinal imaging instrument in ophthalmology for the early diagnosis of pathologies as it is high-resolution and non-invasive. The retina is organized in layers, and it

has been demonstrated that changes in this structure may imply ophtalmic, neurodegenerative and vascular disorders [1]. Examples of these diseases are age-related macular degeneration (AMD), diabetic macular edema (DME) or glaucoma. Furthermore, as part of the central nervous system (CNS), the retina is subject to a variety of specialized immune responses similar to those happening in the spinal cord and the brain. CNS disorders such as Alzheimer’s disease, Parkinson’s disease, multiple sclerosis or stroke have been associated to changes in the retinal structure [2].

Glaucoma is the second leading cause of blindness globally. The disease is caused by increased intra-ocular pressure resulting in irreversible damage to the optic nerve head (ONH). Systematic in depth ONH diagnostics after a positive glaucoma screening would save 4M cases of blindness yearly worldwide but a cost-effective test is not possible with existing technology. A leading candidate technology is optical coherence tomography (OCT). OCT is label free and non-invasive and can be implemented in compact and easy to handle systems with the potential to be used worldwide as diagnostic approach. Currently even high quality commercial systems typically achieve about 3–5 μm . However glaucoma screening requires a very high axial resolution for the posterior segment of the eye; perhaps as small as 1 μm . This paper is framed within the European project GALAHAD, which aims at building a new ultra-high resolution (UHR-) OCT system in a very cost-effective manner and developing new machine learning algorithms to analyse both existing data and new UHR- and polarisation sensitive (PS-) OCT images to identify the early stages of glaucoma.

Although the purpose of OCT is human retina imaging, rodents are sometimes utilized in research studies of ophthalmologic diseases and for treatment evaluation which would be unfeasible in humans. However, human and rodent retinas present significant differences mainly due to the difference in size. For example, rodents lenses are relatively larger, they have no macula or fovea. In addition, the thickness of rodent retinal layers is close to the resolution of standard OCT devices, which makes image processing of rodent OCT images even more challenging. For example, while the ganglion cell layer (GCL) in human eyes has a thickness of 20–60 μm , it is not visibly distinguishable in rodent OCT images (about 2 μm of thickness). The contrary occurs with the inner plexiform layer (IPL), which is relatively thick in rodent retinas (about 60 μm).

Endothelin-1 (ET-1) is a potent vasoactive peptide that causes vasoconstriction of retinal vessels and subsequent ischemia, which contributes to the degeneration of the retinal layers. The ischemia effects are similar to those present in glaucoma patients, which makes rat model of intravitreal injection of ET-1 suitable for research studies of the glaucoma disease and for testing the effectiveness of new treatments to slow down the layer degeneration process.

Some works have already attempted to classify retinal OCT images from different databases in healthy and a variety of pathologies. For example, in [3] the authors accurately detect AMD and DME using multiscale histograms of oriented gradient (HOG) descriptors as feature vectors of a support vector machine (SVM) in a database of more than 3.000 OCT images. [4] employed the VGG16

architecture to classify more than 80,000 OCT images in healthy or AMD with an area under the curve (AUC) of 92.77%. [1] achieved 99%, 89% and 86% in classifying healthy, AMD and DME OCT images from 45 patients by fine-tuning the GoogLeNet architecture. As for glaucoma detection, [5] achieved 93.1% of accuracy on 102 patients feeding a random forest with the features extracted by a convolutional neural network (CNN). Finally, [6] obtained an $AUC = 0.88$ in discriminating glaucoma from healthy in 899 patients by applying a logistic regression on different features, such as retinal nerve fiber layer thickness, inter-ocular difference or age. At the light of these results, it remains unknown which features are enough discriminating to accurately classify glaucoma in OCT images [7].

In this paper, an automatic method to distinguish between healthy and a rodent model of glaucoma where damage to the inner retina is induced by intravitreal injection of endothelin-1 is presented. Its aim is to develop a robust method for glaucoma modelling in rat OCT images to then adapt it to be used with humans. Furthermore, this is a first step to better understand which features are the most discriminating at characterizing glaucoma in OCT images.

2 Methods

2.1 Data

A set of 89 Sprague-Dawley rats were used in the study. Rat OCT images were taken with the Micron IV equipment (Phoenix Research Labs, Pleasanton (USA), <http://www.phoenixreslabs.com/products/oct2/>). On average, 16 images per rat were recorded giving place to a total of 1912 rat OCT images (Fig. 1). Rats were anesthetized previously to image acquisition. Several OCT images were acquired before and after intravitreal injection of ET-1. In particular, a follow-up 14 days after injection was performed, when the maximum expression of retinal layer modification is produced according to the rodent model of glaucoma induced by ET-1 injection [8].

2.2 Deep Learning

Deep learning techniques have become the dominant machine learning approach for a wide variety of problems. Differently to traditional machine learning approaches, where features employed to solve the problem are hand-crafted by a human (feature engineering) and thus require a large amount of time and resources, deep learning methods learn these features by themselves through high amounts of data. Although it was first introduced over 20 years ago [9], it has been over the last few years when deep learning has lead to an improvement of the state-of-the-art in several fields of artificial intelligence and machine-learning, such as image processing, voice processing and language processing applications. This was possible due to the increase of computational power achieved by GPUs and the availability of huge amounts of labeled data. Within the scope of image processing, CNNs are the most used approach [1].

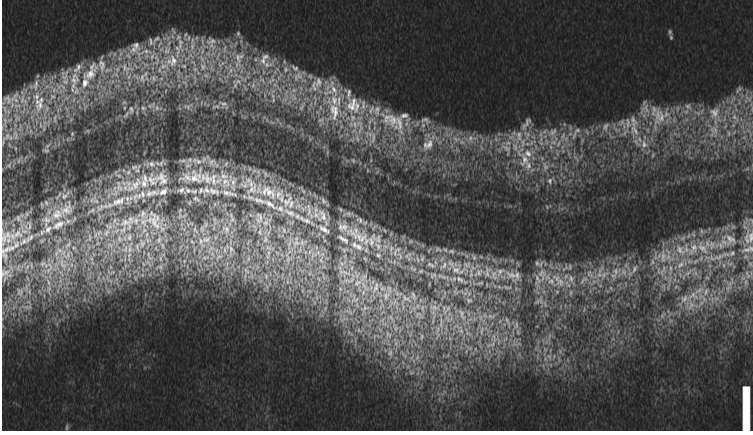


Fig. 1. Example of a healthy rat OCT acquired with the Micron IV equipment.

In contrast to other deep neural networks which base their learning in updating the weight of their neurons, CNNs learn matrices (kernels or filters) to convolve across inputs to extract features from 3D tensors, normally images, and feed them to the following layer. Thus, first layers of CNNs learn to detect basic patterns, such as edges or color blobs of different hue, while layers towards the end of the model learn to detect more complex and problem-specific patterns. While convolutional layers are the heart of CNNs, other kind of layers use to be interspersed with them. Some of these layers are activation layers, such as rectified linear units layer (ReLU), which applies a fixed activation function to their input, pooling layers, which down-sample the spatial dimensions of data, or batch normalization layers, which normalize each input channel across a batch in order to speed up training and reduce the sensitivity to network initialization. A full explanation of the layers employed in deep learning is out of the scope of this paper.

DenseNet Architecture. Many CNNs process data serially, that is, each layer operates only upon the output of the previous layer. In contrast, the DenseNet architecture allows each layer of the network to directly process the outputs of all previous layers. This schema permits features extracted in the early layers of the network to propagate across the network without being perturbed by subsequent layers, so the later layers can use this knowledge. Moreover, the availability of unaltered feature maps from initial layers in the final ones prevents the gradient “vanishing” problem and makes training via back-propagation more efficient [9].

The DenseNet architecture is composed by so called Dense Blocks and Transitions. Figure 2 shows its global architecture.

Transitions consists of a batch normalization layer and an 1×1 convolutional layer followed by a 2×2 average polling layer [9]. On the other hand, the

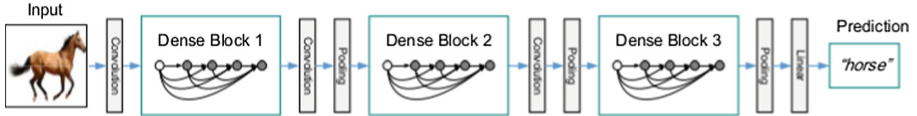


Fig. 2. A deep DenseNet with three Dense Blocks and two transitions (convolution + pooling layers).

Dense Blocks are composed by sequences of Batch Normalization, ReLU and Convolutional (BN-ReLU-Conv) layers.

In this work, we use the DenseNet-201 architecture, composed by four Dense Blocks consisting of two BN-ReLU-Conv sequences of 1×1 and 3×3 kernel sizes. The number of filters of each Dense Block is 6, 12, 48 and 32, respectively. Furthermore, we added a top model consisting of 5 layers: a dense layer of 512 units with ReLU activation followed by a 30% dropout layer, a dense layer of 128 units with ReLU activation followed by a 30% dropout layer and a final dense layer of 2 units with softmax activation, which is the one giving the predictions.

Fine Tuning of Existent Architectures. While deep learning has revolutionized the field of computer vision, it is sometimes difficult to employ it due to the lack of large datasets, as it requires tens of thousands of examples work effectively. To overcome this limitation, transfer learning [1] has been introduced, in which a pre-trained model is employed for a different task than the one for which it was trained.

2.3 Classification Approach

Our approach for classifying the OCT images in healthy (Day 0 or before injection) and pathological (Day 14 after injection) is based on the application of the DenseNet-201 with a customized top model (explained above). Concretely, we first train the network from scratch and then perform fine tuning to compare their behaviour. We implement no pre-processing step, just a resizing of OCT images to 224×224 pixels. The employed hardware is an nVidia Titan Xp GPU with 12 GB of GDDR5X memory and 3840 CUDA cores. The network was configured to use the Adam optimizer [10] with a learning rate of 2×10^{-4} , a batch size of 16 images per batch and 40 epochs.

2.4 Evaluation Methods and Metrics

We use a leave-P-out cross-validation, where P refers to sets of rats mutually exclusive (that is, a single rat appears only in one set). The reason for this kind of cross-validation is to ensure that the training set and the test set contained images from mutually exclusive groups (i.e. there is no single rat which contributed both the training and test set). This decision was made to prevent the network to learn all type of images. Rats are distributed in cages and groups.

Thus, a rat labeled as C1G1 identifies the rat 1 in cage 1, the one labeled C1G5 identifies rat 5 in cage 1, and so on. There are a total of 30 sets of rats which vary in the number of OCT images recorded. Then, to perform the validation, 15 folds of 2 sets of rats were made, trying to keep the folds balanced (i.e. with a similar number of healthy and pathological images). Table 1 shows the distribution of the 15 folds performed.

Table 1. Distribution of rat OCT images for the leave-P-out cross-validation. C stands for Cage, G for group, h for number of healthy images, p for number of pathological images and % for the percentage of healthy images.

	F1		F2		F3		F4		F5		F6		F7		F8		F9		F10		F11		F12		F13		F14		F15	
C	19	27	20	29	19	30	23	27	19	29	23	28	20	30	19	27	23	30	20	28	20	29	24	28	24	30	24	23	20	19
G	5	4	2	1	2	1	1	2	1	2	3	3	3	2	4	1	4	4	1	1	5	4	2	2	1	3	3	2	4	3
h	8	32	8	25	11	25	14	24	11	21	19	21	9	20	10	22	12	18	7	27	8	16	18	18	20	13	19	18	10	14
p	37	11	33	10	46	11	54	11	38	10	53	10	25	10	26	11	30	10	17	16	18	10	39	12	41	10	38	30	18	25
%	45.45		43.42		40.00		36.89		40.00		38.83		45.31		46.38		42.86		50.75		46.15		41.38		39.29		35.24		35.82	

3 Results

In this section, we show and compare the results obtained with the network trained from scratch and fine-tuned. All parameters were exactly the same for both experiments, the only difference was that first the experiment were performed randomly initializing the weights of the network, and then we trained the network pre-initializing it with the ImageNet weights (i.e. we fine-tuned it). Figure 3 shows the progress of the training of both networks in terms of validation accuracy. As can be observed, the behaviour of the fine-tuned network is better than the one trained from scratch. The fine-tuned network achieves a validation accuracy in the range of its final accuracy at epoch 30, whereas the network trained from scratch does so at epoch 55 and its variability is higher.

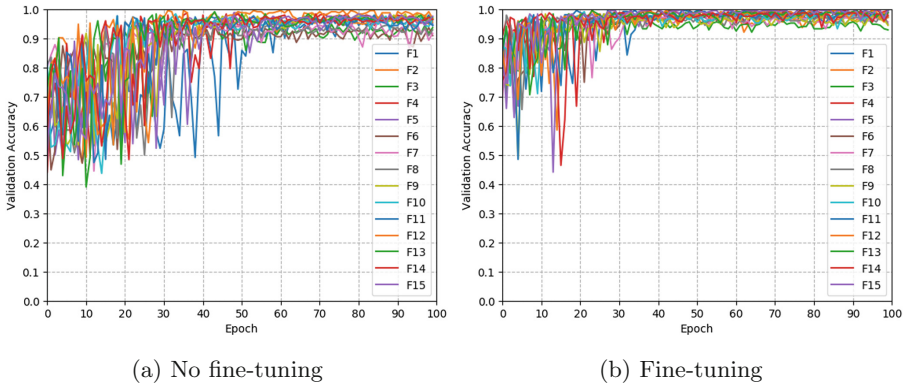


Fig. 3. Progress of the accuracy obtained in the validation set during training for each of the 15 folds.

As for the test set, Fig. 4 shows the ROC curve and Table 2 presents the sensitivity, specificity, accuracy and area under the curve (AUC) of the 15 folds performed for the test set.

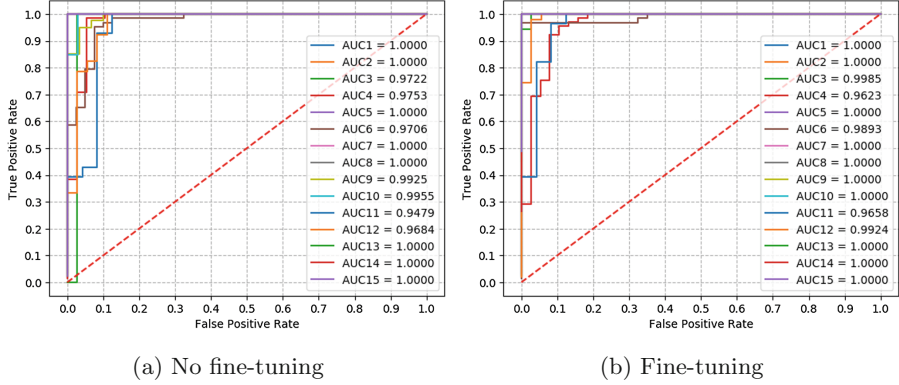


Fig. 4. ROC curve for each of the 15 folds.

Table 2. Metrics of the test set for the 15 folds performed, mean and standard deviation. The metrics presented include: area under the curve (AUC), accuracy (Acc.), sensitivity (Sn), specificity (Sp) and F1-score (F1).

	No fine-tuning					Fine-tuning				
	AUC	Acc.	Sn	Sp	F1	AUC	Acc.	Sn	Sp	F1
F1	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
F2	1.0000	0.9342	1.0000	0.8958	0.9180	1.0000	0.9737	1.0000	0.9556	0.9688
F3	0.9722	0.9889	1.0000	0.9818	0.9859	0.9985	0.9778	0.9722	0.9815	0.9722
F4	0.9753	0.9709	0.9730	0.9697	0.9600	0.9623	0.7476	0.5968	0.9756	0.7400
F5	1.0000	0.9875	0.9697	1.0000	0.9846	1.0000	1.0000	1.0000	1.0000	1.0000
F6	0.9706	0.7670	0.6333	0.9535	0.7600	0.9893	0.9320	0.8511	1.0000	0.9195
F7	1.0000	0.9062	1.0000	0.8537	0.8846	1.0000	0.9062	1.0000	0.8537	0.8846
F8	1.0000	0.9710	0.9412	1.0000	0.9697	1.0000	0.9710	0.9412	1.000	0.9697
F9	0.9925	0.9571	0.9655	0.9515	0.9492	1.0000	0.9714	0.9375	1.0000	0.9677
F10	0.9955	0.9104	1.0000	0.8462	0.9032	1.0000	1.0000	1.0000	1.0000	1.0000
F11	0.9479	0.9423	1.0000	0.9032	0.9333	0.9658	0.9423	1.0000	0.9032	0.9333
F12	0.9684	0.8506	0.7447	0.9750	0.8434	0.9924	0.9080	0.8333	0.9778	0.8974
F13	1.0000	0.9524	0.8919	1.0000	0.9429	1.0000	1.0000	1.0000	1.0000	1.0000
F14	1.0000	0.9905	0.9737	1.0000	0.9867	1.0000	0.9810	0.9487	1.0000	0.9737
F15	1.0000	1.0000	1.0000	1.000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
μ	0.9882	0.9419	0.9395	0.9553	0.9348	0.9939	0.9541	0.9387	0.9765	0.9485
σ	0.0162	0.0615	0.1044	0.0528	0.0638	0.0121	0.0635	0.1055	0.0415	0.0668

4 Conclusions

We have presented a method to accurately classify rodent OCT images in healthy or glaucomatous by using the DenseNet-201 architecture and a customized top-model. As can be observed in Fig. 3, the fine-tuned approach is faster to converge and has less variability. We achieve a mean $AUC = 0.9939$, with 0.9387 of sensitivity and 0.9765 of specificity. Although this work has been performed on rodent OCT images, this is a first step to a very promising human OCT screening system, since rodent OCT images are more challenging than human's due to their characteristics and resolution. In future works, we will study the activation of the layers in the network to better understand what characteristics allow it to be accurate, which will suppose important insights in glaucoma understanding. Furthermore, we will try to improve the system by applying some pre-processing steps to the images. Finally, the system will be modified to work with humans.

Acknowledgments. Animal experiment permission was granted by the Danish Animal Experimentation Council (license number: 2017-15-0201-01213). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. This work was supported by the Project GALAHAD [H2020-ICT-2016-2017, 732613].

References

1. Karri, S., Chakraborty, D., Chatterjee, J.: Transfer learning based classification of optical coherence tomography images with diabetic macular edema and dry age-related macular degeneration. *Biomed. Opt. Express* **8**(2), 579–592 (2017)
2. Pekala, M., Joshi, N., Freund, D.E., Bressler, N.M., et al.: Deep learning based retinal OCT segmentation. arXiv preprint [arXiv:1801.09749](https://arxiv.org/abs/1801.09749) (2018)
3. Srinivasan, P.P., Kim, L.A., Mettu, P.S., et al.: Fully automated detection of diabetic macular edema and dry age-related macular degeneration from optical coherence tomography images. *Biomed. Opt. Express* **5**(10), 3568–3577 (2014)
4. Lee, C.S., Baughman, D.M., Lee, A.Y.: Deep learning is effective for the classification of OCT images of normal versus age-related macular degeneration. arXiv preprint [arXiv:1612.04891](https://arxiv.org/abs/1612.04891) (2016)
5. Muhammad, H., Fuchs, T.J., De Cuir, N., De Moraes, C.G., et al.: Hybrid deep learning on single wide-field optical coherence tomography scans accurately classifies glaucoma suspects. *J. Glaucoma* **26**(12), 1086–1094 (2017)
6. Virgili, G., Michelessi, M., Cook, J., Boachie, C., et al.: Diagnostic accuracy of optical coherence tomography for diagnosing glaucoma: secondary analyses of the gate study. *Br. J. Ophthalmol.* **102**(5), 604–610 (2017). [bjophthalmol-2017](https://doi.org/10.1136/bjophthalmol-2017-025000)
7. Hood, D.C.: Improving our understanding, and detection, of glaucomatous damage: an approach based upon OCT. *Prog. Retin. eye res.* **57**, 46–75 (2017)
8. Nagata, A., Omachi, K., Higashide, T., et al.: OCT evaluation of neuroprotective effects of tafluprost on retinal injury after intravitreal injection of endothelin-1 in the rat eye. *Invest. Ophthalmol. Vis. Sci.* **55**(2), 1040–1047 (2014)
9. Huang, G., Liu, Z., Weinberger, K.Q., van der Maaten, L.: Densely connected convolutional networks. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, p. 3 (2017)
10. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)



Finding the Importance of Facial Features in Social Trait Perception

Félix Fuentes-Hurtado^(✉), Jose Antonio Diego-Mas, Valery Naranjo,
and Mariano Alcañiz

Instituto de Investigación e Innovación en Bioingeniería, I3B,
Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain
flifuehu@upv.es

Abstract. We are constantly making very fast attributions from faces, such as whether a person is trustworthy or threatening, that influence our behavior towards people. In this work, we present a method to automatically tell the importance of facial features on social trait perception. We employ an unsupervised clustering method to group the facial features by similarity and then create a model which explains the contribution of each facial feature to each social trait by means of a Genetic Algorithm. Our model deals with the difficulties associated to quantifying social impression using judgments from human observers (low inter- and intra-observer agreement) and obtains significant correlations greater than 0.7 for all social impressions, which justifies the method developed. Finally, the weights of the eyebrows, eyes, nose, mouth, jawline and facial feature distances are shown and discussed. This work poses a step forward in social trait impression understanding, as to the date, there is no other work quantifying the effects of facial features on social trait perception.

Keywords: Clustering · Genetic algorithms · Social trait impressions

1 Introduction

People are constantly making attributions from faces, such as whether a person is trustworthy or threatening. It is with good reason that the face is the primary source of visual information for identifying people and reading their emotional and mental states [1]. In fact, these attributions are formed very fast in our brains: an exposure of as little as 34 ms is enough to form an impression, and they do not change with exposures longer than 200 ms. Furthermore, attributions made from faces are very important, as they are very likely to influence our behaviour towards people, such as whom we help, whom we hire, or whom we ask for a date [2, 3].

As Kaheman et al. [4] suggested, impressions from faces are natural assessments that have more to do with perception than with thinking. The face has been considered as a window to a person's true nature ever since ancient cultures. However, it was in the nineteenth century when this assumption reached

his heyday. Lavater et al. [5], a Swiss pastor, spread the ideas of physiognomy, the “art” of reading personality in faces. Lombroso et al. [6], the founding father of criminal anthropology, wrote about how it could be possible to identify criminals by external, physical characteristics. Galton et al. [7] developed the first morphing techniques in his work to identify specific human types ranging from the ideal English man to the criminal. Their ideas were discarded in the twentieth century, however, they were onto something.

A large body of research shows that facial appearances influence significant social outcomes in very diverse domains, such as politics, law, business, and the military. Many of the performed studies find that particular facial characteristics can help to experience desirable outcomes (e.g., winning an election) or avoid undesirable outcomes (e.g., being convicted of a crime) [8]. Indeed, Kramer et al. [9] show how internal facial features are signals of personality and health. So, what differences in facial structure and appearance lead to these social inferences? For example, what information is used by people to decide if a face looks trustworthy or untrustworthy? To the best of our knowledge, there is no quantitative information about the importance of each facial feature contribution to social trait perception.

A big problem when trying to model the possible relationships among facial characteristics and social traits is that the space of possible variables driving the perceptions of these traits is infinitely large, so it results almost impossible to solve this problem with conventional approaches. According to Kramer et al. [9], many personality traits can be accurately judged from static facial features. In fact, internal features can by themselves carry much of the information used for personality judgments. Therefore, in this work, we propose a complete pipeline to find the relationships among facial features and social traits in an automatic, quantitative manner. To do so, we implement a clustering method to group all similar facial features by appearance, with which we use as input together with the assessments available in the employed database to a pseudo-heuristic search method (Genetic Algorithm) in order to find the sought relationships.

The rest of the paper is organized as follows: Sect. 2 shows the methods implemented. It details the procedure followed to build a facial feature taxonomy and how the relationships among the different types of facial features and social traits were obtained. Section 3 shows the results and the validation of the proposed procedure. Finally, Sect. 4 provides the discussion of the method and conclusions.

2 Methods

The procedure followed in this work is as follows. First, we extracted the facial features (eyebrows, eyes, noses, mouths and jawlines) from a number of White faces with available assessments in some social traits. As the number of facial features was high and there were similarities among them, we then performed a clustering to group all the similar facial traits together. We used two facial landmark detectors [10,11] to locate the facial features, extract and align them

(one for the internal facial features and the another one for the jawline), and then employed the eigenfaces approach [12] to obtain the basis in which the facial feature images could be decomposed. The clustering was performed using as input the weights characterizing each of the available facial feature images. Finally, we employed a Genetic Algorithm to optimize a model able to tell the weight of each facial feature on each social trait.

2.1 Data

The database employed in this work is the Chicago Face Database [13]. Concretely, we used the 93 white male faces of ages ranging from 16 to 43 present in the database. Each target in the database was represented with a normalized neutral expression photo. According to Kramer et al. [9], static properties of the face have been associated with enduring behavioural biases in the form of personality. This means that it is possible to identify certain personality traits of strangers at a level significantly above chance, based only on a photograph of the face with a neutral expression. We chose to use this database as it has ratings on 15 different social traits (Afraid, Angry, Attractive, Babyface, Disgusted, Dominant, Feminine, Happy, Masculine, Prototypic, Sad, Surprised, Threatening, Trustworthy and Unusual).

2.2 Grouping of Facial Features by Appearance

Facial Feature Extraction. We employed a facial landmark detector to locate and extract six different facial features: eyebrows, eyes, nose, mouth, jawlines and distances. These can be classified into internal facial features (eyebrows, eyes, nose and mouth) and external (jawline and distances). The jawline was defined by its landmarks and the centroids of all the internal facial features, while the distances were defined as the distance from the centroid of the polygon formed by the landmarks belonging to each facial feature to the lowermost point of the jawline. All the internal facial features belonging to the same group were aligned using their centroid and their image's size normalized. Jawlines were not normalized as the CFD images were already normalized and aligned.

Eigenfaces Approach. With the facial feature images aligned and normalized, the next step is their clustering. To do so, we implemented the *eigenfaces* [12] approach to obtain a relatively low-dimensional vector of characteristics which characterized the features instead of using the raw images (very sensitive to noise and difficult to cluster due to the high dimensionality). This method performs a PCA analysis over an ensemble of face images to form a set of basis features [14]. These basis images, known as eigenpictures, can be linearly combined (multiplied by a weight and added up together) to reconstruct images in the original set. This procedure allows for automatic, robust, fast and objective holistic characterization of the facial features considering their appearance while summarizing the central information. Internal features were characterized

using just 45 eigenvalues. The same value was chosen for all of them in order to facilitate the subsequent clustering process, bearing in mind that the explained variances were about 85% or higher in all cases. On the other hand, jawlines were clustered using the previously extracted coordinates, and distances making use of intervals.

K-means Clustering. The clustering or classification by appearance of the extracted facial features is necessary in order to create taxonomies, which are needed to create the face model, in turn necessary to obtain the facial feature weights for each social trait. The reason for this is explained using only internal facial features for the sake of simplicity.

Let's consider a list of faces with their internal features (eyebrows, eyes, noses and mouths). Although it might seem enough to find a function of similarity able to tell how similar two features are and then take the score of the face where the most similar feature is found, by doing this, the assumption that features are independent among them is made, which is incorrect. When looking a face, people see a gestalt of features, meaning that features interfere one with another in order to form the impressions people experience when seeing faces. One way to gather this interferences or relationships among features is to group features by similarity. In this way, when a new face is processed, the eye is compared with all the groups available, and one is chosen. Then, inside this group, several eyes extracted from several different faces are present. Thus, this group has a mean score computed having into account every present face. The key point is that being together in the same group means that all these eyes are similar while the group score has been computed using the face score of the faces they belong and not some sort of eye score, because there is no available score for just the eyes. This links the score obtained by the cluster to the aforementioned type of eye, and it can be considered that this score will be the same or very similar for any possible combination of the rest of facial features, because the eye cluster itself has been created accounting for some variance of these other features, as shows Fig. 1a. However, if all the eyes were considered without any clustering, a similarity function would tell which is the most similar eye, but this eye would be inside a face with certain eyebrows, nose and mouth (reference face). Therefore, this score would be valid only if the rest of facial features of the queried face also belonged to the same clusters as the reference face (Fig. 1b). Although the ideal situation would be to account for 2nd, 3rd, 4th order (and so on) relationships among features, thus modeling relationships of every feature with each other, this would require a huge amount of rated faces unavailable at the moment. Then, the implemented solution tries to overcome this problem to the extent possible, accounting for the feature variability present on the faces within a certain cluster.

Let's understand it with the example of Fig. 1. All these eyes represent groups of different faces, which further have their eyebrows, noses and mouths. Thus, these eyes cluster scores are the average of these faces scores for Afraid. With clustering, the score of eye E1 is computed accounting for the scores of three

faces whose eyes fall within cluster E1, but have different eyebrows (EB7, EB5, EB1), noses (N8, N1, N7) and mouths (M10, M2, M5). Without clustering, the score of eye E1 is the score of the face with E1 when combined with EB7, N8 and M10. But if E1 was to be combined with another eyebrow (EB3, for example), this score (2, 56) would no longer be valid, as it was not computed taking into account that it could be used with other different features.

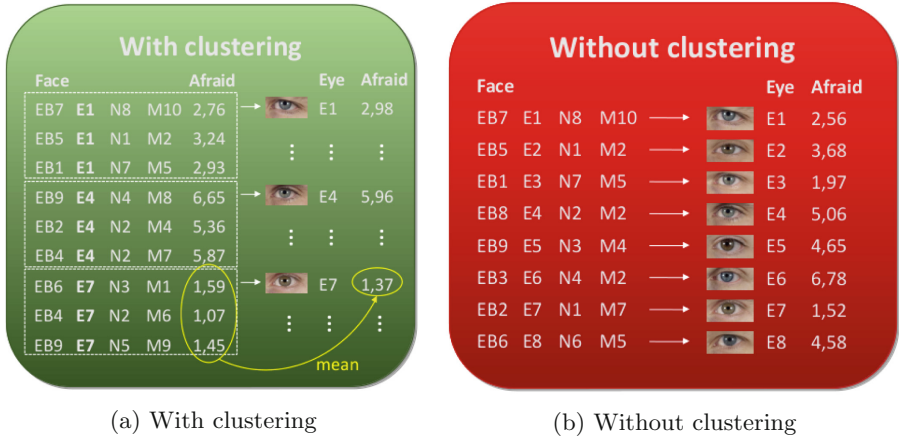


Fig. 1. Why clustering is needed.

We selected the K-means algorithm to perform the clustering. The input were the 45 first weights which multiplied by the basis images recovered the original facial feature images (this kept at least the 85% of the explained variance). As the number of clusters was unknown *a priori*, we performed clusterings with $k = 5, 6, 7 \dots 25$ clusters. Then, for each clustering, we computed the Dunn's Index [15] and the number of clusters with only 1 instance (*mono-clusters*), and established a decision criterion to chose the most suitable clustering as the one with the highest Dunn's Index and less than 2 mono-clusters. This decision was made to promote clusterings whose clusters had a similar number of instances. We then validated the chosen clustering with humans.

2.3 Finding Relationships Among Facial Features and Social Traits

As was previously commented, the employed database provides ratings in a series of social traits (i.e. each face has a score for a given social trait). Then, after performing the clustering, each cluster score was computed as the average of all faces' score belonging to it (i.e. all the faces which feature was within the given cluster). In this manner, we computed a score for each type of facial feature which we call *cluster mean score*.

As literature shows, not all facial features have the same importance when inferring social attributes [16]. Then, we implemented a weighted sum of the

facial features scores. This allowed us to recompute the face score of the existing faces by using our function and compare it with the real one. The weight optimization was performed using a Genetic Algorithm to optimize the following function (i denotes the social impression for which the score is being calculated):

$$GFS^i = \begin{pmatrix} score_{EB}^i \\ score_E^i \\ score_N^i \\ score_M^i \\ score_J^i \\ score_{DEB}^i \\ score_{DE}^i \\ score_{DN}^i \\ score_{DM}^i \\ score_{DEE}^i \end{pmatrix} * \begin{pmatrix} weight_{EB}^i \\ weight_E^i \\ weight_N^i \\ weight_M^i \\ weight_J^i \\ weight_{DEB}^i \\ weight_{DE}^i \\ weight_{DN}^i \\ weight_{DM}^i \\ weight_{DEE}^i \end{pmatrix}^T \quad (1)$$

However, as the values employed in order to compute this calculation are the means of the facial features belonging to each cluster, the variance is diminished in excess. Therefore, this prediction function cannot reproduce extreme values present in the CFD database, as the means have flattened them out. To solve this situation, the predicted values are transformed so they have the same mean and standard deviation as the original CFD scores. As each social trait has a different mean and standard deviation, this process is performed separately for each of them. Equation 2 illustrates this operation.

$$GFS_{expanded} = \frac{GFS - \mu_{GFS}}{\sigma_{GFS}} \cdot \sigma_{CFD} + \mu_{CFD} \quad (2)$$

Where GFS is the global face score for a given impression, μ_{GFS} and σ_{GFS} are defined as the mean and the standard deviation of the global face score obtained with the evaluation function, and μ_{CFD} and σ_{CFD} are the mean and the standard deviation of the CFD scores.

As aforementioned, the goal of the developed model is to achieve a good fitting of the evaluating function to the data available on the CFD. The better this fitting is, the better the weights represent the relationships between the facial features and the social traits. The GA employed to perform this optimization was configured to perform single-point crossover and uniform mutation with a probability of $P_{crossover} = 0.6$ and $P_{mutation} = 0.4$ respectively on a population of 50 individuals. The permitted range for the weights was set to the interval $[0, 1]$. The number of iterations was established at 200 000, however, it was never reached due to the early stopping condition implemented. This condition allowed for a maximum of 100 consecutive iterations without a change higher than 0.0001 in the fitness function solution. The selection method employed was Stochastic Universal Sampling, and the Survivor Selection Policy was fitness-based with *elitism*, that is, the best individual was always selected for the next iteration. Finally, the fitness function was defined as the mean squared error between the predictions made with a given combination of weights and the actual face

scores of the CFD faces. With this configuration, the optimization was performed individually for each impression, resulting in a total of 15 set of weights, one for each trait impression.

3 Results

3.1 Clustering Facial Features by Appearance

Table 1 shows the cluster metrics obtained for each facial feature using the clustering procedure explained in methods. k refers to the original number of clusters and k_{final} to the final number of clusters after removing the mono-clusters (referred to as mc in the table). Furthermore, the Dunn’s Index and the Silhouette Index are also given for each clustering for descriptive purposes.

Table 1. Clusters obtained for each facial feature.

	k	k_{final}	# of mc	DI	SI
Eyebrows	12	10	2	0.57	0.21
Eyes	19	19	0	0.86	0.12
Noses	12	12	0	0.62	0.17
Mouths	11	9	2	0.55	0.21
Jawlines	12	11	1	1.00	0.22

3.2 Weights of the Facial Features in the Different Social Traits

As aforementioned, the validity of the computed weights is subject to how well our implemented model is able to reproduce the CFD scores. Then, in this section, we show the fitting achieved for the CFD faces. Three metrics are computed, the correlation (with its corresponding p-value), the coefficient of determination (r^2), and the mean squared error (MSE). As was explained in Sect. 2.3, the global face scores (GFS) obtained with the evaluating function need to be transformed in order to better reproduce the dispersion present in CFD scores. Table 2 shows these results without and with dispersion. Correlations are the same in both cases due to the linearity of the operations performed.

Figure 2 presents the weight of each facial feature averaged for all social traits. Eyes and nose are the most important facial features regarding social trait perception, with just eyes and nose accounting for 45.40% of the importance, and 67.65% if we also consider their corresponding distances (Fig. 2). The distance between eyes is also very important, with 12.34%. Jaw (12.14%), eyebrows (11.21%), and mouth (9.00%) complete the distribution of weights of facial features for social trait impression.

Table 2. Results of the implemented social trait model for each social trait.

	Correlation	p-value	r ²	MSE	
				W/o disp.	W/ disp.
Afraid	0.7018	3.28e-15	0.4925	0.1088	0.1013
Angry	0.7274	1.01e-16	0.5292	0.2745	0.1872
Attractive	0.7661	2.35e-19	0.5869	0.2540	0.1923
Babyface	0.8124	2.86e-23	0.6600	0.3486	0.1661
Disgusted	0.7008	3.71e-15	0.4912	0.1303	0.0841
Dominant	0.7429	1.01e-17	0.5519	0.3461	0.2480
Feminine	0.8086	6.56e-23	0.6538	0.1014	0.0528
Happy	0.7551	1.47e-18	0.5702	0.1966	0.1222
Masculine	0.7927	1.76e-21	0.6283	0.2023	0.1051
Prototypical	0.8208	4.26e-24	0.6737	0.4139	0.7067
Sad	0.7273	1.02e-16	0.5290	0.2486	0.2183
Surprised	0.7755	4.43e-20	0.6015	0.0403	0.0220
Threatening	0.7559	1.30e-18	0.5713	0.2684	0.1730
Trustworthy	0.7492	3.83e-18	0.5612	0.0901	0.0633
Unusual	0.7536	1.87e-18	0.5680	0.2260	0.1896
Mean	0.7593	4.82e-16	0.5779	0.2167	0.1755

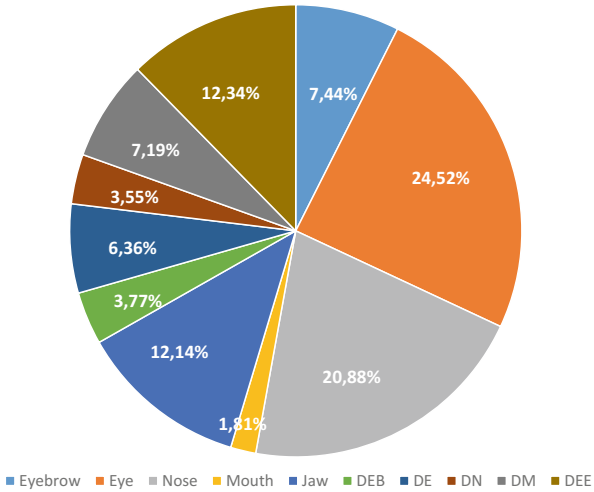


Fig. 2. Average importance of each facial feature in the perception of social traits.

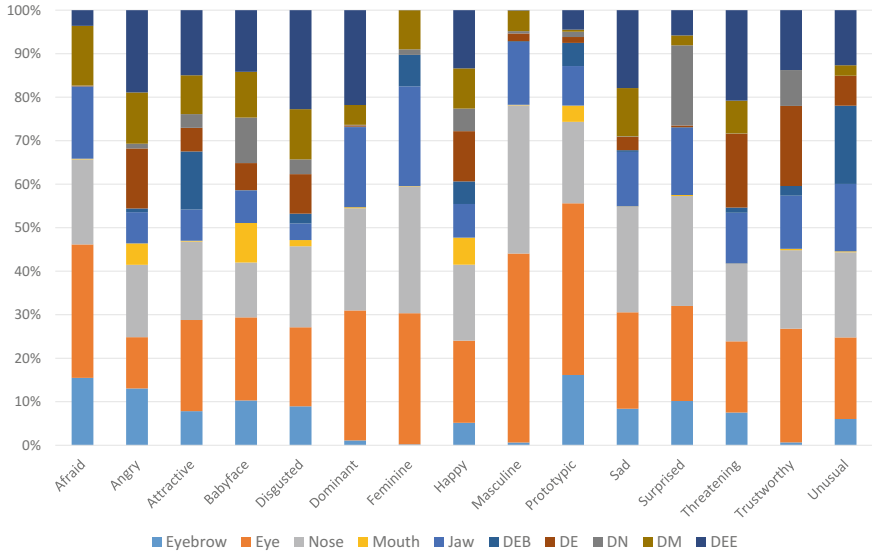


Fig. 3. Importance of each facial feature in the perception of each social trait.

Figure 3 presents the weight of each facial feature for each social trait. For example, eyes alone are very important for the definition of Afraid, Dominant, Masculine and Prototypical; but very few for Angry. Nevertheless, when distances are considered as well, eyes and their related distances obtain the maximum importance for Disgusted, Dominant, Threatening and Trustworthy. This implies a high importance of the eye distances for Disgusted, Threatening and Trustworthy. On the other hand, nose’s importance is higher for Masculine, Sad and Surprised, and lower for Babyface and Unusual. Mouth is very few important for every impression, obtaining its maximum for Babyface, with less than 10%. Jaw reaches its highest importance for Afraid, Dominant, Feminine and Surprised. Finally, it is also important to note how distances are very important, obtaining near 50% of importance for several impressions.

4 Conclusions

In this work, we presented an automatic method for assessing the importance of facial features in social trait perception. We computed the weights for fifteen social traits obtaining a fit with a correlation greater than 0.75 and a $r^2 > 0.55$ for 9 of them, and close values for the rest. This means that our model is accurate if we think about the within- and between-person variability when assessing social traits [17]. This poses a very interesting tool for forensic computing, psychologists, marketing, etc., as it allows to compute the *average* social traits associated with a face automatically as opposed to performing a trial with a group

of persons. Moreover, due to the general design of our pipeline, it is possible to perform this same procedure with any database of interest.

Nevertheless, the model implemented to compute the facial feature weights has some important limitations, as the absence of hair modeling or the assumption of face symmetry. In addition, interactions between features are disregarded, which might vastly improve the performance of the model. However, we did not dispose of enough photographs as to consider these inter-relationships. Thus, the low quantity of faces available has been a limiting factor as well. In future works, more images will be gathered and labeled, and the face model will include hair, asymmetry and features inter-relationships. Furthermore, ethnicity independent models will also be created and the female gender included.

To sum up, considering the great subjectivity and variability present in social trait perception, we consider that the developed model is a step forward in the understanding of how these impressions are produced.

References

1. Todorov, A., Dotsch, R., Wigboldus, D.H., Said, C.P.: Data-driven methods for modeling social perception. *Soc. Pers. Psychol. Compass* **5**(10), 775–791 (2011)
2. Zebrowitz, L.A., Montepare, J.M.: Social psychological face perception: why appearance matters. *Soc. Pers. Psychol. Compass* **2**(3), 1497–1517 (2008)
3. Rule, N., Ambady, N.: First impressions of the face: predicting success. *Soc. Pers. Psychol. Compass* **4**(8), 506–516 (2010)
4. Kahneman, D.: A perspective on judgment and choice: mapping bounded rationality. *Am. Psychol.* **58**(9), 697 (2003)
5. Lavater, J.C.: *Essays on physiognomy: for the promotion of the knowledge and the love of mankind; written in the German language by JC Lavater, abridged from Mr. Holcrofts translation. Printed for GGJ & J, Robinson* (1800)
6. Lombroso, C.: *Criminal man, translated and with a new introduction by Mary Gibson and Nicole Hahn Raftar. Duke University Press, Durham, NC* (2006). (Original work published 1876 and 1897)
7. Galton, F.: *Inquiries into the Human Faculty & its Development. JM Dent and Company, Darlington* (1883)
8. Olivola, C.Y., Funk, F., Todorov, A.: Social attributions from faces bias human choices. *Trends Cogn. Sci.* **18**(11), 566–570 (2014)
9. Kramer, R.S., Ward, R.: Internal facial features are signals of personality and health. *Q. J. Exp. Psychol.* **63**(11), 2273–2287 (2010)
10. Asthana, A., Zafeiriou, S., Cheng, S., Pantic, M.: Incremental face alignment in the wild. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1859–1866 (2014)
11. Thomas, P.B., Baltrušaitis, T., Robinson, P., Vivian, A.J.: The cambridge face tracker: accurate, low cost measurement of head posture using computer vision and face recognition software. *Transl. Vis. Sci. Technol.* **5**(5), 8 (2016)
12. Turk, M., Pentland, A.: Eigenfaces for recognition. *J. Cogn. Neurosci.* **3**(1), 71–86 (1991)
13. Ma, D.S., Correll, J., Wittenbrink, B.: The Chicago face database: a free stimulus set of faces and norming data. *Behav. Res. Methods* **47**(4), 1122–1135 (2015)

14. Sirovich, L., Kirby, M.: Low-dimensional procedure for the characterization of human faces. *Josa a* **4**(3), 519–524 (1987)
15. Dunn, J.C.: Well-separated clusters and optimal fuzzy partitions. *J. Cybern.* **4**(1), 95–104 (1974)
16. Santos, I.M., Young, A.W.: Inferring social attributes from different face regions: evidence for holistic processing. *Q. J. Exp. Psychol.* **64**(4), 751–766 (2011)
17. Sutherland, C.A., Young, A.W., Rhodes, G.: Facial first impressions from another angle: how social judgements are influenced by changeable and invariant facial properties. *Br. J. Psychol.* **108**(2), 397–415 (2017)



Effective Centralized Trust Management Model for Internet of Things

Hela Maddar^(✉), Wafa Kammoun^(✉), and Habib Youssef^(✉)

PRINCE Research Laboratory ISITCom, Sousse University, Hammam Sousse,
Sousse, Tunisia

maddar.hela.si@gmail.com, wafa_kammoun@yahoo.fr,
habib.youssef@fsm.rnu.tn

Abstract. The emergence of the Internet of Things (IoT) is a result of convergence between multiple technologies, like Internet, wireless communication, embedded systems, microelectronic systems and nanotechnology. In 2016, 5.5 million objects are connected every day in the world. A number that could quickly reach billions by 2020 [1]. Gartner predicts that 26 billion objects will be installed in 2020. The market for connected objects could range from a few tens of billions to up to several thousand billion units. Among the vital components of IoT, we find wireless sensor networks (WSNs). Wireless sensor networks as a vital component of the IoT, allow the representation of dynamic characteristics of the real world in the virtual world of the Internet. Nevertheless, the opening of these types of network to the Internet presents a serious problem stand point security. For that, the implementation of intrusion detection mechanisms is essential to limit the internal and external attacks that threaten the smooth running of the network. In this paper, we propose an efficient trust management model which seeks deeply through the nodes to detect attacks that threaten wireless sensor networks. Our model includes a geographical localization system used to identify the nodes location. Although, it includes a set of rules detection attacks based on different parameter analysis. Furthermore, we propose a mathematical model for trust establishment and its update on the network. During the simulations we observe an improvement of the efficiency of the implemented geo-location model and also a reasonable energy consumption. Similarly, we have been able to evince the efficiency of our model in terms of attacks detection rate.

Keywords: IoT · WSNs · Trust · Security · Direct trust · Indirect trust
TDOA

1 Introduction

Recently, we assist to the emergence of the Internet of Things (IoT), which represents a new paradigm that is upsetting the field of networks and telecommunications. Thanks to Internet of Things, we had the opportunity to connect anywhere and anytime. We just need to have an internet connection. The IoT allow us to connect different devices such as Radio Frequency Identification (RFID). Their role consist on tracking the traceability of objects followed.

Other components of the IoT are drones. These are miniature unmanned aircraft. Drones are now elected to make an important part of the future Internet either as an intelligent object reporting control data or as a particular mobile data router flying between different parties connected to the internet. Smartphones and tablets are already connected to the internet by various technologies such as Wi-Fi, 3G and 4G. These allow users to supervise and order their connected objects remotely via their smartphones or tablets.

Wireless sensor networks are an important technology of Internet of Things. They represent the state of objects or environment in which they are deployed. The data collected will be communicated to the various components of the Internet of Things and subsequently to the user who can view, check and monitor the data collected through his smartphone without any need to move. All his need is an internet connection. Nevertheless, the opening of wireless sensor networks to the internet presents a serious problem stand-point safety.

In fact, security solutions already approved for WSNs are not enough. The term security is usually related to cryptography. Cryptographic mechanism only cannot be an effective solution to manage the various vulnerabilities of WSNs. An attacker which was able to access to the network and have valid cryptographic keys will be seen as normal node and cannot be detected as malicious. In this case, Trust Management is seen primordial to detect intrusions and denials of service that may occur on the network.

In this work, we propose a new lightweight model of trust management for Internet of Things. Specifically for WSNs. In our model, a verification of node identity is always checked with TDOA (Time Difference Of Arrival) [2] which is a location technique used to detect any Sybil or identity theft attack. Thereafter, we have proposed a set of detection rules based on nodes behavioral analysis to detect any malicious event occurring on the network. Subsequently, we proposed a trust calculation model to classify node behavior as trusted or untrusted. Simulations were performed to prove the performance of our model.

This paper is organized as follows: Sect. 2 presents the state of the art, Sect. 3 describes our proposed model, Sect. 4 presents the trust establishment model proposed, Sect. 5 describes inter and intra cluster intrusion detection model, Sect. 6 presents the experimentation results, and finally Sect. 7 concludes the paper.

2 State of the Art

Several researchers were done on Trust and intrusion detection, which is seen as an essential attribute in relationship building between entities. It has been studied for a long time by scientists from different scientific fields [3–6]. The general idea of trust system for sensor networks is to observe the behavior of the sensors, their conformity with respect to what is expected, to calculate and to assign values of trust to the different participating nodes. The first trusted management model used for WSN was proposed by Josang & Ismail in 2002 [3]. It is a probabilistic mechanism model developed for e-commerce system and applied to WSNs. Ganeriwal and Srivatava in Reputation-based Framework for High Integrity Sensor Networks [4] used the work of

Josang & Ismail based on the beta probability function to create their trust model for wireless sensors Networks. This work is the first model of trust and reputation designated and developed exclusively for wireless sensor networks. In PLUS (Parameterized and Localized Trust Management Scheme) [7], the authors assume that all important control packets generated by the base station must contain a hash sequence number. The inclusion of hash in the control packets increases the size of packet and the transmission, reception power consumption. Each time the judge node receive a packet from another neighboring node, it always checks its integrity. If the integrity of the packet fails, the trust value of the sender node will be decreased.

Other recent trust models based on IDS ‘Intrusion Detection System’ to differentiate between honest and dishonest nodes have been developed. On [8], the authors have created detective rules implemented in nodes agents which are charged by observation. This work presents much more performance compared to existing models like [4, 7]. This detection model is based on two evaluation criteria: the number of packets deleted and the radio power value received. Four attacks are managed by this model. Another more recent model created by the same authors in [9], took into account more attacks to detect and increase the evaluation criteria. Given the large amount of energy dissipated by trusted management models and the vital importance of energy for the survival of this type of network, the authors in [10] have created a lightweight trust management model that allow to limit attacks by proposing some light criteria of behaviors evaluations without setting precise rules for each attack. Another recent model of trust management based on IDS agent presented in [11]. This model is a powerful model stand point detection of several attacks. Nevertheless it is very costly stand point energy consumption as the most trust management model where all nodes are charged by the observation of all neighbors nodes present in their radio range.

In [12], the authors propose an intrusion detection system which makes possible to classify the behavior of sensor nodes as trusted or not. This model manage three types of attacks, jamming attack, selective attack and the hello flood attack. This detection is based on the evaluation of three parameters. The received radio power, the ratio between the received packets and the sent packets and the rate of the control packets generated.

The authors in [13], have proposed a trust model based on energy calculation to detect Sybil attack. According to the authors, each node in the network is identified by a triplet <identity, location, energy>. Each time a new message is received, a verification of the validity of the identity received with the received location is performed. Thus, a verification of the energy consumed must take place.

In [14], the authors propose four rules on which the verification of the behavior of the sensor nodes is based. This model of detection is established using a multi-agent system where each detection rules are observed by specific agents.

In our work, we seek to create a new trust management model based on different evaluation parameters. An <identity, location> conformance check is always performed to ensure the validity of the source of message received. This verification is based on the TDOA geographic location technique. To remind, GPS (Global Position System) cannot be an effective solution when we are in the indoor, and, given the heterogeneous nature of IoT, GPS system is not sufficient for our needs. For this we have implemented a localization technique that works at the indoors and outdoors to be sufficient for IoT

needs. A node behavior analysis rules set has been fixed to detect any malicious event that has survived on the network. Subsequently, we propose a mathematical model to update the trust values of network nodes. A node with trust value below the minimum threshold will be considered as malicious and will be removed from the network. Finally, different simulations were performed to prove the performance of our model.

3 Our Proposed Model

In this section, we present our network model and the various proposed intrusion detection rules based on different network settings.

3.1 Network Model

Our work is based on a hierarchical network whose resources between nodes are equivalent. The network is made by normal nodes SN, cluster heads CH and a base station BS which is supposed trusted. Communication between SN nodes and CH is accomplished in one-hop. Similarly for the communication between the CHs and the base station. Nodes of the same type (SN or CH) are loaded by the same process. They are not randomly deployed. The network is divided into zones. Each zone has an equivalent number of nodes. The election of a new CH for a given area will always be made from the same area. The routing protocol used for communication is the LEACH Protocol [14].

3.2 Proposed Detection Rules

Our evaluation model is based on six rules to evaluate the behavior of nodes that will be detailed in the following. Initial thresholds for the measured network parameters will be set during an initialization period that is assumed trusted. The observation of the behavior in the network is carried out by cluster heads and the base station. Each zone leader is responsible for assessing the behavior of nodes of its zone. A periodic report of the measured parameters will be sent to the base station after each period of time T_{period} . The base station is responsible for evaluating the behavior of the different CH based on parameter calculated described below for each CH. The trust update is only performed by the base station.

Location Compliance Verification Rule

In order to verify <identity, location> compliance, we used the TDOA (Time difference of Arrival) localization technique. The TDOA technique consists to calculate the arrival time of signal at different receivers, then, calculate the difference in arrival time between at least three receivers. Between a pair of nodes i and j , the TDOA is expressed by the formula below:

$$TDOA_{ij} = t_i - t_j = \frac{1}{c}(d_i - d_j) = \frac{d_{ij}}{c} \quad (1)$$

This position corresponds to the intersection of the hyperbolas obtained from TDOA estimated by each node. To determinate nodes location, anchor nodes are deployed specifically to determine the location of nodes, which contribute only to the geographic computation and are not limited in resources. These nodes are assumed trusted. If no conformity <identity, location> is detected, a Sybil or identity theft attack is detected. A location conformance check is performed by the base station to determine the malicious nodes.

Radio Signal Strength Indication RSSI

The RSSI is the power indicator contained in a received radio signal. Radio power may decrease over time but never increase. From this, we set a rule to check the compatibility between the periodic received average radio power at each time period T_{period} with the initial average radio power received in an initial time $T_{initial}$. The average initial radio power is expressed by:

$$RSSI_{avg_initial} = \sum_{i=1}^{t_{initial}} rssi_i / t_{initial} \quad (2)$$

The periodic average power calculated in a period T_{period} is expressed by the following mathematical formula:

$$RSSI_{avg} = \sum_{i=1}^t RSSI_i / t \quad (3)$$

However if $RSSI_{avg} > RSSI_{avg_initial}$, a sinkhole, blackhole or wormhole attack is detected.

Number of Packets Received NPR

In our network, the nodes are pre-loaded by the same processing. Thus, the number of packets sent by nodes of same type (SN or CH) is the same. In order to verify that the number of packets received does not exceed the normal state, a check of the average number of packets received in an initialization period $T_{initial}$ is performed with the average number of packets received is expressed by:

$$NPR_{avg_initial} = \sum_{i=1}^{t_initial} NPR_i / t_initial \quad (4)$$

Subsequently, a periodic check of the number of packets received in each period of time T_{period} is performed too with the average number of packets received is expressed by the following mathematical formula:

$$NPR_{avg} = \sum_{i=1}^t NPR_i / t \quad (5)$$

If the number of packets received in a T_{period} time excels the average number of packets received in the initialization period, an abnormal behavior will be detected. Note that the number of packets received may decrease over time but never increase. This decrease is usually due to the failure of nodes, loss of packets due to collusion, or jamming, which can survive on network.

If $NPR_{avg} > NPR_{avg_initial}$, denial of service or sinkhole attack is detected.

Energy Consuming Amount ECA

In our detection model, we periodically conduct a verification of the amount of energy consumed ECA at each time period. The amount of energy dissipated is almost the same at each time period for nodes of same type as these nodes are loaded by the same processing.

The amount of energy consumed in each period is determined by the following mathematical formula:

$$ECA_{avg} = ECA_{avg+t} - ECA_{avg}/t \quad (6)$$

Where ECA_{avg+t} is the average value of energy calculated at time t minus the old average value determined at the previous time t.

The average energy dissipated in an initialization period $T_{initial}$ is expressed by:

$$ECA_{avg_initial} = \sum_{i=1}^{t_{initial}} ECA_i/t_{initial} \quad (7)$$

If $ECA_{avg} > ECA_{avg_initial}$, denial of service or sinkhole attack is detected.

Jitter

It is the variation of latency over time. Transmission delays can be affected by network conditions such as imperfect connections, node failures, and collusion. A decrease in the latency time strongly indicates the presence of attack on the network.

To detect any abnormal packet delay events, we periodically calculate an average value of the received jitter value. This value will be compared with the initial average value of jitter determined in an initialization period $T_{initial}$.

With,

$$Jitter_{avg} = \sum_{i=1}^t Jitter_i/t \quad (8)$$

And

$$Jitter_{avg_initial} = \sum_{i=1}^{t_initial} Jitter_i/t_initial \quad (9)$$

If $Jitter_{avg} < Jitter_{avg_initial}$, denial of service attack is detected.

4 Establishment of Trust

In our model, the establishment of trust is based on direct observation, indirect observation, updating and aging. The calculation of trust is based on direct and indirect observation. Aging and updating are factors that guarantee the efficiency and robustness of our model.

4.1 Trusted Values

Initially each node is trusty, and has a trust value equal to 1. In our model, a trust threshold is set to 0.5. A node is considered trusted if its trust value is greater or equal than 0.5.

4.2 Trust Calculation Process

Trust calculation is done in a centralized way where only the base station is loaded by this task. The calculation formula proposed is as follows:

$$T_{SB}[i, t] = T_{SB}[i, t - 1] - \left[\sum_{R=1}^{Rn} (\alpha_R * \beta_R) \right] \quad (10)$$

With $T_{SB}[i, t - 1]$ represents the old trust value pre-loaded in the base station data base related to i node from network in a preceding time $t-1$. The variable Rn represent the set of rules with R represent the current rule to check. The variable α_R is a Boolean variable, if the rule is applied, α_R takes 1 otherwise it takes 0. The variable β_R takes 0.1 in case of noncompliance of RSSI or Jitter or ECA or NPR detected. In case of no conformity location detected, β_R takes $T_{SB}[ch, t - 1]$, and the trust of the current node will be reset to 0.

5 Inter and Intra Cluster Intrusion Detection Model

Within an area of the network, the observation of nodes behavior is carried out by CH. A periodic calculation of different parameters to evaluate is performed for each node in the zone. A report containing the result will be sent to the base station after each evaluation period.

Clusters heads are a very attractive target for attackers because they represent the center of data collection of their zone. After data collection, they are charged by the aggregation of data received, then by transmitting it to the sink. CHs behaviors are evaluated by the base station basing on the different rules already cited. Only the base station has the right to update nodes trust and eliminate untrusted nodes.

6 Experimentation

To evaluate the features of our model, we used MATLAB. In order to test the performances of our model we applied our detection model on the LEACH routing protocol. The choice of LEACH protocol is based on its popularity and widespread use in recent research. Indeed, our model can be applied on any protocol with hierarchical architecture. The purpose of our simulations is to first assess the effectiveness of the applied localization model. Subsequently, we are interested to evaluate the amount of energy consumed. Thirdly, we study the performance in terms of attacks detection in comparison with existing models. Below are the different parameter values taken during our experimentation (Table 1).

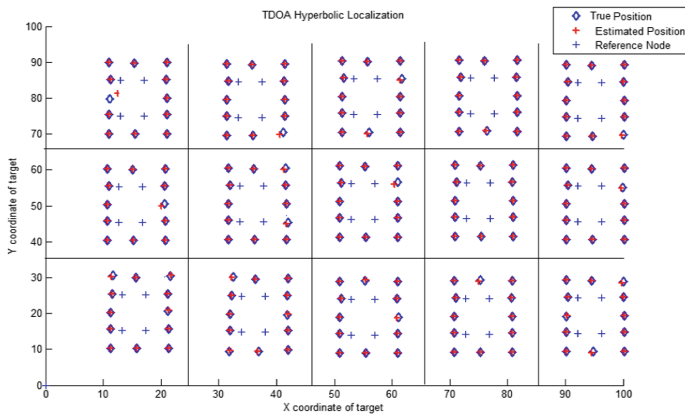
Table 1. Parameters of the experimental model.

Parameter	Value
Routing protocol used	LEACH
Area	100*100 m ²
Simulation round	300
Number of node	91, 181
Sink location	50*50
Radio range	50 m
MAC protocol used	IEEE 802.15.4
Initial Energy	0.22 j
Transmitting and receiving energy Elec = ETX = ERX	50 nJ
Aggregation energy	5 nJ
T initial	300 s
T period	60 s

In order to evaluate the performance of our model, we firstly begin with effectiveness evaluation of the geographic location which is performed according to the number of reference nodes present in each zone. To note, reference nodes are fixed, trusted and with unlimited resources, they only contribute to the geographical calculation.

6.1 Location Detection Using the TDOA Geo-Location Algorithm

As we know, in order to be able to determinate the location with TDOA measurement, it takes at least three reference nodes. In our test, we firstly evaluate location determination with four reference nodes present in each area (Fig. 1).

**Fig. 1.** Location detection with four reference nodes for 181 sensors network.

From the simulation result shown, we can see clearly the good rate of correct estimation of true positions using four reference node in each zone. Nevertheless, the

result found does not satisfy our need. For this, we have increased the number of reference nodes to five. The simulation result are shown below (Fig. 2).

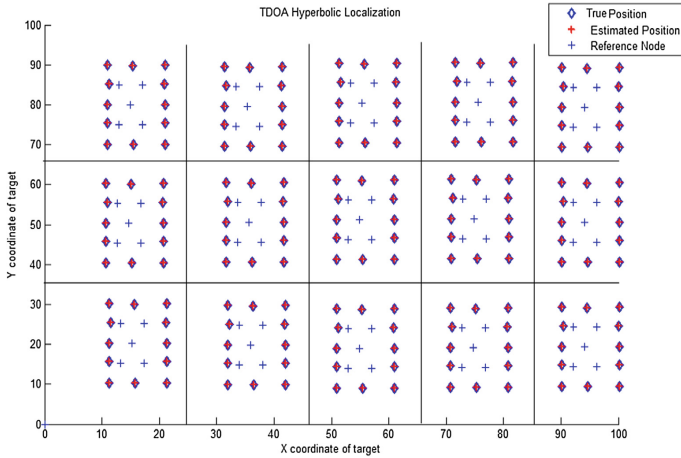


Fig. 2. Location detection with five reference nodes for 181 sensors network.

Despite the good results obtained with four reference nodes, we opt to have better results and better performance. To do this, we increased the number of reference nodes to five nodes per zone. We have acquired a correct estimation rate equal to 100%.

From what precedes and based on various tests performed, in our work, we set the number of reference nodes to five. We proceed now to evaluate the amount of energy dissipated by our model.

6.2 Evaluation of Dissipated Energy

The curve below represents the energy dissipation rate by LEACH and trust-LEACH protocols on network composed by 91 nodes (Fig. 3).

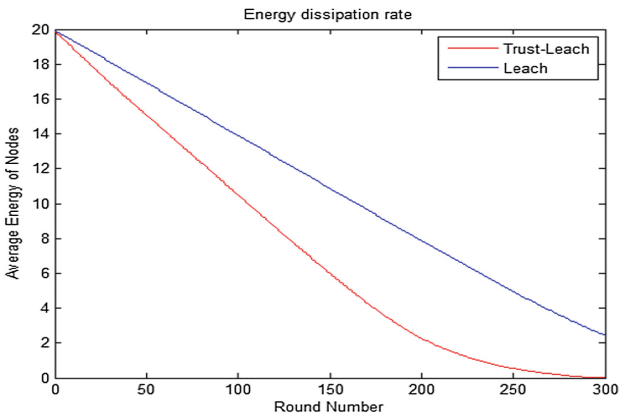


Fig. 3. Energy Dissipation for LEACH and Trust-LEACH.

The curve above represents energy consumption rate of LEACH and trust-LEACH protocols. We can clearly determinate the very reasonable overconsumption of energy produced by the trust-LEACH protocol. This over consumption is due to the calculation of the thresholds during the initialization period, the periodic calculation of the averages of different parameters at each T_{period} , as well as the periodic sending of the evaluation reports made by the different cluster heads to the base station to maintain network trust.

6.3 Detection Rate of Malicious Behavior

We move now to evaluate the performance of our proposal with existing one [13, 18–20]. For that, a new test is provided. This test involves exposing the network to 15 malicious nodes for different network sizes. Experimentation result are presented by the curve below (Fig. 4).

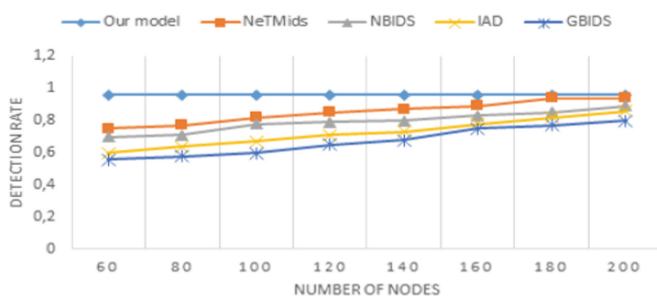


Fig. 4. Comparative study of attacks detection rate for different detection models.

From the curve presented above, we clearly note that the best performances are obtained by our model with constant average detection rate around 96% whatever is the network size. If we compare our model with Muhamed Sayed and al. model named NetMids [13], we notice that in this model the detection rate increases while increasing the number of network nodes. It reaches 94% in the case of network consisting of 200 sensor nodes. For NBIDS model [18], the best detection rate obtained is 89% for network composed by 200 sensor nodes. At this stage, the IAD model [19] has a detection rate equivalent to 80%. In these different models, we notice that while we increase the number nodes of network, the rate of attacks detection increases. In reality, this is very significant since the attacks detection in these different models is based on the observation of nodes behaviors by their direct neighboring nodes. Then, while we increase the neighborhoods number, the rate of attacks detection increase. Such method can cause false diagnosis, and very low detection rate in case of network with small number of nodes. The resilience rate will be much affected because of neighborhood nodes number that will minimize over time (energy exhaustion). In our model, this problem does not exist since each node is evaluated by its successor basing on the different proposed detection rules. Therefore, whatever the number of nodes, our network keeps the same performance.

7 Conclusion

In this work, we propose a model that presents some improvements by the early detection of the abnormal behavior of a node within internet of things. Our model resist against several attacks that threaten the reliability, safety and the good operational mode of WSN. To detect intrusions, we developed a set of detection rules. The results of simulations have shown the effectiveness of attacks detection comparing to the existing models presented on literature. Thanks to our location model we can verify the transmitter node localization basing on ‘TDOA’ technique, which make us able to detect any sybil or identity theft attack.

In spite of the effectiveness of our model, in future works we seek to improve the detection rules based on more criteria and take into account the imperfections of networks to minimize the confusions that can take place on networks. Similarly, we seek to improve the lifetime of the network by the integration of Cloud technology [17] to retain energy consumed and to take into account more IoT components.

References

1. Insecurity of connected objects: “how to combine IoT and security”. Net Journal. Accessed 4 May 2016
2. Steinberg, J.: Contributor on cybersecurity and entrepreneurship, “these devices may be spying on you”. Net Journal, 27 January 2014
3. Kaune, R., Hörst, J., Koch, W.: Accuracy analysis for TDOA localization in sensor networks. In: Information Fusion. IEEE (2011)
4. Jøsang, A., Ismail, R.: The beta reputation system. In: The 15th Bled Electronic Commerce Conference, Bled, Slovenia (2002)
5. Ganeriwal, S., Balzano, L.K., Srivastava, M.B.: Reputation-based frame work for high integrity sensor networks. In: Proceedings of the 2nd ACM workshop on Security of Adhoc and Sensor Networks (2004)
6. Han, G., Jiang, J., Shu, L., Niu, J., Chao, H.C.: Management and applications of trust in wireless sensor networks: a survey. J. Comput. Syst. Sci. (2013)
7. Rodrigo, R., Carmen, F., Javier, L., Hwa, C.: Trust and reputation systems for wireless sensor networks. Secur. Priv. Mob. Wirel. Netw. (2009)
8. Yao, Z., Kimand, D., Doh, Y.: PLUS: parameterized and localized trust management scheme for sensor networks security. In: International Conference on Mobile Adhoc and Sensor Systems. IEEE (2008)
9. Sedjelmaci, H., Senouci, S.M., Feham, M.: An efficient intrusion detection framework in cluster-based wireless sensor networks. Secur. Commun. J. **6**(10), 1211–1224 (2013)
10. Sedjelmaci, H., Senouci, S.M.: Efficient and lightweight intrusion detection based on nodes behaviors in wireless sensor networks. In: Global Information Infrastructure Symposium. IEEE (2013)
11. Maddar, H., Kammoun, W., Cheikhrouhou, O., Youssef, H.: Lightweight trust model with high longevity for wireless sensor networks. In: International Conference on Information Systems Security and Privacy. Springer, Heidelberg (2016)

12. Maddar, H., Kammoun, W., Youssef, H.: Trust intrusion detection system based on location for wireless sensor network. In: Madureira, A.M., Abraham, A., Gamboa, D., Novais, P. (eds.) ISDA 2016. AISC, vol. 557, pp. 831–840. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-53480-0_82
13. Sajjada, S.M., Boukb, S.H., Yousaf, M.: Neighbor node trust based intrusion detection system for WSN. In: 6th International Conference on Emerging Ubiquitous Systems and Pervasive Networks, EUSPN. Elsevier (2015)
14. Alsaedia, N., Hashima, F., et al.: Detecting sybil attacks in clustered wireless sensor networks based on energy trust system (ETS). *Comput. Commun. J.* **110**, 75–82 (2017)
15. Jin, X., Liang, J., et al.: Multi-agent trust-based intrusion detection scheme for wireless sensor networks. *Comput. Electr. Eng.* **59**, 262–273 (2017)
16. Tandel, R.I.: Leach protocol in wireless sensor network: a survey. (*IJCSIT*) *Int. J. Comput. Sci. Inf. Technol.* **7**(4), 1894–1896 (2016)
17. Maddar, H., Kammoun, W., Youssef, H.: Cloudlets architecture for wireless sensor network. In: Madureira, A.M., Abraham, A., Gamboa, D., Novais, P. (eds.) ISDA 2016. AISC, vol. 557, pp. 852–862. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-53480-0_84
18. Stetsko, A., Folkman, L., Matyayas, V.: Neighbor-based intrusion detection for wireless sensor networks. In: International Conference on Wireless and Mobile Communications, Los Alamitos, CA, USA, pp. 420–425 (2010)
19. Liu, F., Cheng, X., Chen, D.: Insider attacker detection in wireless sensor networks. In: Proceedings of IEEE INFOCOM, pp. 1937–1945 (2007)
20. Li, G., He, J., Fu, Y.: A group-based intrusion detection scheme in wireless sensor networks. In: Proceedings of GPS - Workshops, pp. 286–291. IEEE (2008)



Knowledge-Based Solution Construction for Evolutionary Minimization of Systemic Risk

Krzysztof Michalak^(✉)

Department of Information Technologies, Institute of Business Informatics,
Wrocław University of Economics, Wrocław, Poland
krzysztof.michalak@ue.wroc.pl

Abstract. This paper concerns a problem of minimizing systemic risk in a system composed of interconnected entities such as companies on the market. Systemic risk arises, when, because of an initial failure of a limited number of elements, a significant part of the system fails. The system is modelled as a graph, with some nodes in the graph initially failing. The spreading of failures can be stopped by protecting nodes in the graph, which in case of companies can be achieved by setting aside reserve funds. The goal of the optimization problem is to reduce the number of nodes that eventually fail due to connections in the system. This paper studies the possibility of utilizing external knowledge for solution construction in this problem. Rules representing reusable information are extracted from solutions of problem instances and are used when solving new instances.

Experiments presented in the paper show that using rule-based knowledge representation for constructing initial population allows the evolutionary algorithm to attain better results during the optimization run.

Keywords: Knowledge-based optimization
Rule-based knowledge representation · Graph problems · REDS graphs

1 Introduction

Many real-life phenomena such as bankruptcies, epidemics, viruses in computer networks, failures in power grids and wildfires can be modelled in terms of failures of interconnected nodes in a network. Once a node has failed (e.g. a person fell ill or a company has gone bankrupt, etc.) it incurs an additional load on surrounding nodes (e.g. exposing acquaintances to germs, delaying payments to related companies, etc.). Because of that increased load, surrounding nodes may also fail giving rise to a cascading failure which can be a catastrophic event destroying the entire system.

The problem tackled in this paper is a problem of reduction of systemic risk [1]. Systemic risk is often analyzed using network models [2, 3]. As in other works on spreading of threats in networks, in this paper the system is modelled as an

undirected graph $G = \langle V, E \rangle$ with N_v nodes. The nodes in the graph represent the entities subject to failures and they can be in one of the states ‘F’ - failed, ‘P’ - protected, or ‘U’ - unprotected. In the initial state S_0 some nodes from a given non-empty set $\emptyset \neq S \subset V$ are set to the state ‘F’ and the remaining ones are set to the state ‘U’. In the problem studied here each node in the graph can be protected or not, but the total resources that can be used for protecting the graph are limited and we assume that at most N_t nodes can be protected. Therefore, a problem instance is an ordered triple $\langle G, S_0, N_t \rangle$, where N_t is the total number of nodes that can be protected in the graph. The search space is $\{0, 1\}^{N_v}$ with the constraint that the number of ones in any solution cannot exceed N_t . The value of the objective function for a solution $x \in \{0, 1\}^{N_v}$ is calculated by simulating the spreading of failures in the graph. Initially, nodes corresponding to the elements $x[i] = 1$ are protected by setting their state to ‘P’. In each time step $t > 0$ failures progress from nodes in the ‘F’ state to adjacent nodes in the ‘U’ state. Nodes in the ‘P’ state are immune to failures, and this protection lasts to the end of the simulation. When failures stop spreading, the solution x is evaluated by calculating how many nodes in the graph survived (i.e. are in the state ‘U’ or ‘P’).

In this paper an optimization algorithm finds solutions that are vectors, which indicate which nodes in the graph to protect. The proposed approach takes advantage of the fact that a solution for the entire graph can be decomposed to decisions for individual nodes (*protect/do not protect*). The presented method improves solutions in an initial population of an evolutionary algorithm by making decisions for individual nodes using a machine learning algorithm. Initial populations constructed in this way lead to better optimization results than randomly initialized populations.

2 Proposed Approach

The approach proposed in this paper is to use rules for building better initial solutions for an evolutionary optimizer. The rules used in this paper were obtained from solutions of the studied problem found for instances with a low number of nodes in the graph. The proposed approach is summarized below.

1. Run an optimization algorithm on a training set containing several (e.g. $N_{runs} = 30$) instances of the problem
2. Build a training data set in the following manner:
 - For each node $v \in V$ calculate its attributes based on the structure of the graph and on the initial state S_0 . For the list of attributes used in this paper see Table 1.
 - From each of the 30 runs of the optimizer get the best solution found and for each node create one object in the training data set with the attributes of the node and a class indicating if the node is protected in the analyzed solution. This analysis results in N_v objects generated for the training sample, where N_v is the number of nodes in the graph.

- Agglomerate the objects generated in all N_{runs} runs of the test, resulting in a training sample containing $N_{runs} \cdot N_v$ objects partitioned into two classes (*protect* and *do not protect*).
3. Optionally, perform a feature selection step. The goal of this step is twofold. First, with a correctly selected set of attributes classification algorithms may perform better than on the full attribute set. Second, feature selection methods attempt to assess the importance of the attributes and, in some cases, provide a ranking of the attributes which can be used to tell which attributes of the graph nodes are more and which less important.
 4. Generate a set of rules using a chosen machine learning algorithm.
 5. Use the obtained rules for generating initial solutions when solving other, possibly larger, problem instances.

3 Experiments and Results

The experiments performed in this paper were conducted following the steps presented in Sect. 2.

3.1 Test Instances

The graphs used in the experiments were REDS graphs [4] generated on a $[0, 1] \times [0, 1]$ square. The REDS graphs combine spatial relationships with social synergy effects. The edges consume “energy” of the nodes proportionally to the edge length D_{ij} , but the edge cost is discounted by the factor of $\frac{1}{1+S k_{ij}}$ for nodes v_i and v_j that have k_{ij} neighbours in common. This method of generating graphs results in a distribution of edges with denser cliques separated by relatively sparse areas, which is a distribution that resembles that of a real-life social network [4]. The first round of experiments was focused on generating decision rules and in this round graphs with $N_v = 1000$ were generated with the maximum distance between connected nodes $R = 0.1$, the social energy $E = 0.15$ and the synergy parameter $S = 0.5$. In the second round of experiments in which the universality of the generated rules was tested, problem instances with $N_v = 1000, \dots, 3000$ generated with parameters presented in Table 4 were used. Note, that all the parameters shown in Table 4 are adjustable, except for the average degree \bar{k} which is calculated from the obtained graphs. The initial state S_0 was generated for each instance by setting $N_b = 50$ randomly selected nodes to the ‘F’ state. The choice of the initially failed nodes was kept constant between different runs of the tested methods on the same graph in order to allow comparing the results. The above-described settings were confirmed in preliminary experiments to generate problem instances neither too easy (with failures immediately contained) nor too difficult (with almost the entire graph failing).

3.2 The Optimization Algorithm

The solution space of the systemic risk minimization problem presented in this paper is $\{0, 1\}^{N_v}$ with a constraint limiting the number of the elements equal

to 1 to at most N_t (representing the limitation of the available resources used for protecting graph nodes). To solve this problem without additional repair operators or constraint handling techniques an evolutionary algorithm was used with the chromosomes being elements of $[0, 1]^{N_v}$. In order to calculate the objective function value, each solution $s \in [0, 1]^{N_v}$ was decoded by selecting N_t largest elements and setting them to 1 and the remaining elements to 0, obtaining a vector $s' \in \{0, 1\}^{N_v}$. Subsequently, a simulation of the spreading of failures in the graph was performed starting with the initial state S_0 . Nodes corresponding to positions equal to 1 in s' were set to the ‘P’ state, excluding the nodes already failed in the initial state S_0 . This exclusion was imposed in order to prevent the algorithm from finding an easy solution of protecting all the nodes set to ‘F’ in S_0 thereby preventing the failures from occurring at all.

The population size was set to $N_{pop} = 1000$ and the algorithm was allowed to run for a maximum of $N_{gen} = 1000$ generations. In this paper multiple crossover and mutation operators were used, because the effectiveness of different operators may vary during the optimization run [5]. Multi-operator evolutionary algorithms have been found to work well for real-valued [6] and combinatorial optimization problems [7]. This approach also worked well for the Firefighter Problem (FFP) [8], which is a related optimization problem concerning the spreading of a threat (“fire”) in a graph. Three classical crossover operators working on the entire chromosomes were used: single point, two-point and uniform crossover. Also, the Simulated Binary Crossover (SBX) [9] was used with the distribution index $\eta_c = 20$, working at each position of the chromosomes separately. The probability of performing the crossover operation on a pair of parent solutions was set to $P_{cross} = 1.0$. Five mutation operators working on the entire chromosomes were used: displacement, insertion, inversion, scramble and transpose with the probability of applying the mutation $P_{mut/s} = 0.05$ per specimen. Also, two mutation operators were used working at each position of the chromosomes separately: polynomial mutation [10] with the distribution index $\eta_m = 20$ and uniform mutation with the probability of applying the mutation $P_{mut/p} = \frac{1}{N_v}$ per position in the chromosome. For deciding which operator to select, an auto-adaptation mechanism was used, selecting the operators with probabilities proportional to operator success rates (i.e. the number of improved solutions generated by the operator, divided by the number of times the operator was used) [8]. In order not to exclude any operator completely from the operator selection process the minimum probability of selecting an operator using the auto-adaptation mechanism was set to $P_{min} = 0.05$. Parent selection was performed using a binary tournament.

3.3 Rule Extraction and New Solutions Construction

For generating rules a training data set has to be prepared, based on the best solutions obtained in a number of runs of the optimization algorithm. In this data set the nodes of the graph are described by attributes and for each node a desired class is assigned (either *protect* or *do not protect* depending on whether the node has actually been protected in a given solution). Using the training data

set a machine learning algorithm builds rules which can subsequently be used for deciding if a given node in the graph should be protected or not. In the experiments the attributes presented in Table 1 were used.

Table 1. Attributes of a graph node $v \in V$ used in the experiments

Name	Description
<i>failed</i>	Indicates whether the node is set to the ‘F’ state at the beginning of the simulation (<i>failed</i> = 1) or not (<i>failed</i> = 0)
<i>deg</i>	Node degree. The number of edges that are adjacent to the node
<i>clos_centr</i>	Closeness centrality. An inverse of the mean shortest path length from v to every other node in the graph
<i>betw_centr</i>	Betweenness centrality. A measure based on the number of the shortest paths between other nodes that go through v
<i>fail_min_dist</i>	The length of the shortest path between the given node and any node that is initially in the state ‘F’
<i>fail_avg_dist</i>	The average length of the shortest paths between the given node and all the nodes that are initially in the state ‘F’

Based on these attributes, rules were generated using the PART rule discovery algorithm [11] implemented in Weka 3.8.1 [12] using the default parameter settings. Rules discovered using all the attributes are presented in Fig. 1. Numbers in parentheses denote how many training examples matched the rule and how many among those were misclassified. For classifying a node the rules are checked from top to bottom and the first matching rule is invoked generating a decision to either *protect* the node or *do not protect*. The last rule which does not have any conditions before the \implies sign is the default classification rule invoked when no other rules matched a given node.

The rules described above can be used to improve initial populations used by a population-based optimizer containing solutions represented as $[0, 1]^{N_v}$ vectors. Each solution in the initial population is modified with a probability $P_{mod/s}$. If a given solution is selected for modification, then each of the nodes in the solution is modified with a probability $P_{mod/p}$. A classification is performed using the rules, and if the classification result is *do not protect* the corresponding element in the solution is set to 0 and if the classification result is *protect* the element is set to 1. Note, that solution vectors contain values in the $[0, 1]$ range and they are decoded to values from the set $\{0, 1\}$ as described in Sect. 3.2. Therefore, even a node classified as *protect* by the classifier may not actually be protected if the N_t limit is exceeded.

To verify if the discovered rules allow generating good solutions, the same evolutionary algorithm was run again, but on a new set of 30 test instances generated separately from the training ones. In these test runs the algorithm started with the same initial populations without modifications and with initial populations improved using the procedure described above with both the probability

R01: $failed \leq 0$ and $fail_min_dist > 2$	\implies <i>do not protect</i>	(11686/833)
R02: $failed > 0$	\implies <i>do not protect</i>	(1500/0)
R03: $betw_centr \leq 0.005532$ and $clos_centr \leq 0.066908$ and $deg > 22$	\implies <i>do not protect</i>	(445/35)
R04: $fail_min_dist > 1$	\implies <i>do not protect</i>	(8185/1989)
R05: $deg > 18$	\implies <i>do not protect</i>	(3562/884)
R06: $betw_centr > 0.001892$ and $deg \leq 10$	\implies <i>protect</i>	(1603/628)
R07: $betw_centr \leq 0.00003$ and $deg > 6$	\implies <i>do not protect</i>	(90/9)
R08: $betw_centr \leq 0.003689$ and $deg > 6$ and $deg \leq 16$	\implies <i>do not protect</i>	(109/17)
R09: $betw_centr > 0.004271$ and $deg > 14$	\implies <i>do not protect</i>	(1106/479)
R10: $betw_centr > 0.004271$ and $deg > 12$	\implies <i>do not protect</i>	(598/291)
R11: $betw_centr > 0.004087$ and $deg \leq 12$	\implies <i>protect</i>	(527/249)
R12: # the default rule	\implies <i>do not protect</i>	(589/210)

Fig. 1. Rules discovered using all the attributes.

$P_{mod/s}$ and the probability $P_{mod/p}$ varied in the range $\{0.2, 0.4, 0.6, 0.8\}$. From each of the 30 runs the objective function value of the best solution found was recorded. Table 2 presents medians from 30 runs obtained for each pair of values of $P_{mod/s}$ and $P_{mod/p}$ and obtained when no improvement of initial populations was performed (denoted “None” in the table). In order to verify significance of the results, statistical testing was performed using the Wilcoxon signed rank test [13]. In cases when a low p-value was obtained it could be concluded that the difference between median results produced by two compared methods was significant. In Table 2 the results significantly better than those obtained without rule-based improvement (at the significance level $\alpha = 0.05$) are marked in bold.

Clearly, when high probabilities of modifying a solution $P_{mod/s}$ and modifying individual positions $P_{mod/p}$ were set, good initial populations were obtained. In particular for $P_{mod/s} = 0.8$ and $P_{mod/p} = 0.8$ the largest median objective function value was obtained and the result was statistically significant.

3.4 Experiments with Feature Selection

A round of experiments described in this section was carried out using Weka in order to verify if the results could be further improved by selecting a good attribute set. Two methods of feature selection were tested. The first was to follow the filter approach and rank the attributes with respect to the Information

Table 2. Medians from 30 runs obtained for each pair of values of $P_{mod/s}$ and $P_{mod/p}$. The best parameter setting is $P_{mod/s} = 0.8$ and $P_{mod/p} = 0.8$.

Modification probabilities	Median obj. func. value	p-value	Modification probabilities	Median obj. func. value	p-value
$P_{mod/s} = 0.2$			$P_{mod/s} = 0.6$		
$P_{mod/p} = 0.2$	648.5	0.9836	$P_{mod/p} = 0.2$	663.5	0.3440
$P_{mod/p} = 0.4$	653.0	0.7189	$P_{mod/p} = 0.4$	632.5	0.7971
$P_{mod/p} = 0.6$	651.5	0.8797	$P_{mod/p} = 0.6$	649.5	0.4555
$P_{mod/p} = 0.8$	648.5	0.0587	$P_{mod/p} = 0.8$	636.5	0.2755
$P_{mod/s} = 0.4$			$P_{mod/s} = 0.8$		
$P_{mod/p} = 0.2$	643.5	0.7132	$P_{mod/p} = 0.2$	655.5	0.5439
$P_{mod/p} = 0.4$	651.5	0.5440	$P_{mod/p} = 0.4$	655.0	0.0878
$P_{mod/p} = 0.6$	637.0	0.5739	$P_{mod/p} = 0.6$	667.5	0.0042
$P_{mod/p} = 0.8$	659.0	0.0359	$P_{mod/p} = 0.8$	668.0	0.0015
None	647.5	-			

Gain Ratio (IGR) measure [14]. Feature ranking obtained using this method is presented in Table 3. In order to select attributes one has to select those that are located on the top of the ranking. Building of rules based on three or fewer attributes from the ranking resulted in a rule set containing only one rule unconditionally returning *do not protect* for all nodes and the entire data set has six attributes. Therefore, two feature sets were considered: $FS_{rank,4} = \{ failed, fail_min_dist, betw_centr, deg \}$ and $FS_{rank,5} = FS_{rank,4} \cup \{ clos_centr \}$.

Table 3. Ranking of features by the Information Gain Ratio (IGR) measure

Rank	Information Gain Ratio	Feature name	Rank	Information Gain Ratio	Feature name
1	0.05784	<i>failed</i>	4	0.00913	<i>deg</i>
2	0.03992	<i>fail_min_dist</i>	5	0.00271	<i>clos_centr</i>
3	0.00927	<i>betw_centr</i>	6	0.00231	<i>fail_avg_dist</i>

Apart from the filter approach a wrapper feature selection was performed which selected the features: $FS_{wrapper} = \{ deg, betw_centr, fail_min_dist \}$. Attribute sets selected by the algorithms seem in line with what is known about the role of various node attributes in prevention of the spreading of threats. The attributes *deg* and *fail_min_dist* are used, for example, in heuristics described in the literature on the Firefighter Problem (FFP), which suggest defending nodes with high degrees and located close to fire first [15]. The *betw_centr* attribute value depends on how many shortest paths go through a node. Protecting nodes with high betweenness centrality should help preventing failures from spreading quickly by following short paths in the graph. The *failed* feature is also

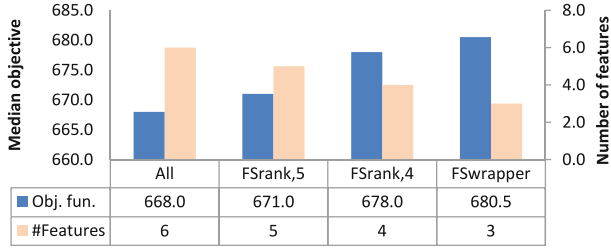


Fig. 2. Median best solutions obtained when initial population was improved using rules built on a set of attributes reduced using feature selection methods.

important, because when $failed = 1$ the node is marked as failed in the initial state S_0 which makes it futile to try to protect that node.

Using the three feature sets $FS_{wrapper}$, $FS_{rank,4}$ and $FS_{rank,5}$ rule sets were built using the PART algorithm. These rule sets were used for improving initial populations with the modification probabilities set to the best found values $P_{mod/s} = 0.8$ and $P_{mod/m} = 0.8$. In Fig. 2 the results obtained using feature selection are presented.

Attribute selection improved the results, with smaller attribute sets performing better than larger ones. The wrapper feature selection produced the best results, which is not very surprising, because wrappers tend to select attribute sets better suited for a particular learning algorithm than filters. On the other hand, wrappers usually require more computations, however, in presented experiments the wrapper feature selection took no more than several seconds to complete. Figure 3 presents rules discovered using the attributes selected by the wrapper feature selection method.

3.5 Testing the Universality of the Obtained Rules

Experiments presented in this section were aimed at testing the universality of the rules discovered for $N_v = 1000$, by using them to generate initial solutions for instances with varying number of nodes. The tests were performed with $P_{mod/s} = 0.8$ and $P_{mod/p} = 0.8$, because these values turned out to be the best in the tests described in previous sections. Parameters of test instances used in this round of experiments are shown in Table 4.

Results presented in Table 5 are median values calculated over 30 runs on test instances of a given size obtained without improving the initial population (denoted “None” in the table) and obtained using rules presented in Figs. 1 (denoted “All”) and 3 (denoted “Wrapper”). For each test the medians obtained using rules are better than when the rules were not used. Results in the “All” and “Wrapper” columns significantly better than those in the “None” column (as verified using the Wilcoxon test at $\alpha = 0.05$) are marked in bold. Results presented in Table 5 show, that the rules generated for $N_v = 1000$ constructed improved solutions for graphs with the number of nodes $N_v \in \{1000, 3000\}$.

R01: $fail_min_dist > 2$	\implies do not protect	(11686/833)
R02: $fail_min_dist > 0$ and $deg \leq 22$ and $fail_min_dist > 1$	\implies do not protect	(7191/1790)
R03: $fail_min_dist > 0$ and $deg > 18$	\implies do not protect	(5001/1118)
R04: $fail_min_dist > 0$ and $deg \leq 10$ and $betw_centr > 0.001634$	\implies protect	(1625/634)
R05: $fail_min_dist \leq 0$	\implies do not protect	(1500/0)
R06: $betw_centr \leq 0.004271$ and $deg > 6$ and $betw_centr \leq 0.000121$	\implies do not protect	(159/25)
R07: $betw_centr \leq 0.004271$ and $deg > 6$ and $deg > 14$ and $deg \leq 16$ and $betw_centr \leq 0.003689$	\implies do not protect	(101/16)
R08: $betw_centr > 0.004271$ and $deg > 14$	\implies do not protect	(1106/479)
R09: $betw_centr > 0.004271$ and $deg > 12$	\implies do not protect	(598/291)
R10: $deg > 12$ and $deg > 16$	\implies do not protect	(101/27)
R11: $betw_centr > 0.003456$ and $deg \leq 12$	\implies protect	(536/255)
R12: $deg > 6$ and $deg \leq 14$	\implies do not protect	(250/80)
R13: # the default rule	\implies do not protect	(146/69)

Fig. 3. Rules discovered using the attributes selected by the wrapper feature selection method ($FS_{wrapper} = \{ deg, betw_centr, fail_min_dist \}$).

3.6 Tests on Erdős-Renyi Graphs

Another round of experiments was performed on graphs constructed according to the Erdős-Renyi model $G(N_v, P_{edge})$ introduced in [16]. The methodology proposed in Sect. 2 was followed, but the graphs used for training and testing were not REDS, but E-R graphs with $N_v = 1000$ nodes and $P_{edge} = 3.0/N_v = 0.003$. The best performing algorithm in these experiments was the algorithm using rule-based population initialization with attributes selected using $FS_{rank,4}$ method and $P_{mod/s} = 0.8$ and $P_{mod/p} = 0.6$. The median number of saved nodes from 30 runs was 343 for the rule-based population initialization and 334 when no improvement of initial populations was performed. The difference between these results was statistically significant, as verified using the Wilcoxon test (obtained p-value: 0.0253).

Table 4. Parameters of problem instances used in the experiments aimed at testing the universality of the rules.

Test instance parameters						
N_v	R	E	S	\bar{k}	N_b	N_t
1000	0.100	0.15	0.500	7.35	50	200
1250	0.089	0.15	0.447	7.97	62	250
1500	0.082	0.15	0.408	8.11	75	300
1750	0.076	0.15	0.378	8.57	88	350
2000	0.070	0.15	0.350	9.16	100	400
2250	0.067	0.15	0.333	9.43	112	450
2500	0.063	0.15	0.316	10.09	125	500
3000	0.060	0.15	0.290	10.06	150	600

Table 5. Results obtained in the experiments aimed at testing the universality of the rules (median value of the best solution found, calculated from 30 runs).

N_v	None	All	Wrapper	N_v	None	All	Wrapper
1000	647.5	668.0	680.5	2000	623.0	641.0	651.0
1250	629.5	653.5	671.0	2250	656.0	665.0	664.5
1500	648.5	666.5	661.5	2500	658.0	670.0	666.0
1750	627.5	643.0	628.5	3000	761.5	764.0	765.0

4 Conclusions

In this paper a method was presented for discovering rules that allow deciding if particular nodes in a graph should be protected in the systemic risk minimization problem. The proposed method works by analyzing solutions of the problem produced by an optimization algorithm, and builds a rule set using a machine learning algorithm. In this paper the PART algorithm was used for generating the rules, because in preliminary experiments it shown promising performance. However, the proposed method is not limited to using PART, and it could also use other classifiers. Presented results suggest that better final results are obtained when the optimization algorithm starts from the initial population improved using the rule-based classifier. Because the testing was performed on a set of problem instances separate from those used for building the rules, it can be stated that the obtained rule sets have the generalization property.

Acknowledgment. This work was supported by the Polish National Science Centre under grant no. 2015/19/D/HS4/02574. Calculations have been carried out using resources provided by Wroclaw Centre for Networking and Supercomputing (<http://wcss.pl>), grant No. 407.

References

1. Burkhholz, R., Leduc, M., Garas, A., Schweitzer, F.: Systemic risk in multiplex networks with asymmetric coupling and threshold feedback. *Phys. D* **323–324**, 64–72 (2016)
2. Battiston, S., Puliga, M., Kaushik, R., Tasca, P., Caldarelli, G.: DebtRank: too central to fail? Financial networks, the FED and systemic risk. *Sci. Rep.* **2**, 541 (2012)
3. Thurner, S., Poledna, S.: DebtRank-transparency: controlling systemic risk in financial networks. *Sci. Rep.* **3**, 1888 (2013)
4. Antonioni, A., Bullock, S., Tomassini, M.: REDS: an energy-constrained spatial social network model. In: Lipson, H., Sayama, H., Rieffel, J., Risi, S., Doursat, R. (eds.) *ALIFE 14: The Fourteenth International Conference on the Synthesis and Simulation of Living Systems*. MIT Press (2014)
5. Esquivel, S.C., Leiva, A., Gallard, R.H.: Multiple crossover per couple in genetic algorithms. In: *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC 1997)*, pp. 103–106 (1997)
6. Elsayed, S.M., Sarker, R.A., Essam, D.L.: Multi-operator based evolutionary algorithms for solving constrained optimization problems. *Comput. Oper. Res.* **38**(12), 1877–1896 (2011)
7. Zhang, L., Wang, L., Zheng, D.Z.: An adaptive genetic algorithm with multiple operators for flowshop scheduling. *Int. J. Adv. Manuf. Technol.* **27**(5), 580–587 (2006)
8. Michalak, K.: The Sim-EA algorithm with operator autoadaptation for the multiobjective firefighter problem. In: Ochoa, G., Chicano, F. (eds.) *EvoCOP 2015*. LNCS, vol. 9026, pp. 184–196. Springer, Cham (2015). <https://doi.org/10.1007/978-3-319-16468-7-16>
9. Deb, K., Agarwal, R.: Simulated binary crossover for continuous search space. *Complex Syst.* **9**(2), 115–148 (1995)
10. Deb, K., Goyal, M.: A combined genetic adaptive search (GeneAS) for engineering design. *Comput. Sci. Inform.* **26**, 30–45 (1996)
11. Vijayarani, S., Divya, M.: An efficient algorithm for generating classification rules. *Int. J. Comput. Sci. Technol.* **2**, 512–515 (2011)
12. Machine Learning Group at the University of Waikato: Weka 3 - Data Mining Software in Java (2017). <https://www.cs.waikato.ac.nz/ml/weka/>
13. Ott, L., Longnecker, M.: *An Introduction to Statistical Methods and Data Analysis*. Brooks/Cole Cengage Learning, Boston (2010)
14. Sharma, A., Dey, S.: Performance investigation of feature selection methods. *CoRR abs/1309.3949* (2013)
15. Garcia-Martinez, C., Blum, C., Rodriguez, F., Lozano, M.: The firefighter problem: empirical results on random graphs. *Comput. Oper. Res.* **60**, 55–66 (2015)
16. Erdős, P.: Some remarks on the theory of graphs. *Bull. Am. Math. Soc.* **53**(4), 292–294 (1947)



Handwritten Character Recognition Using Active Semi-supervised Learning

Papangkorn Inkeaw¹, Jakramate Bootkrajang¹, Teresa Gonçalves²,
and Jeerayut Chaijaruwanich¹(✉)

¹ Department of Computer Science, Faculty of Science, Chiang Mai University,
Chiang Mai 50200, Thailand

{papangkorn.i, jakramate.b, jeerayut.c}@cmu.ac.th

² Department of Informatics, University of Évora, 7000-671 Évora, Portugal
tcg@uevora.pt

Abstract. Constructing a handwritten character recognition model is considered challenging partly due to the high variety of handwriting styles and the limited amount of training data. In practice, only a handful of labeled examples from limited number of writers are provided during the training of the model. Still, a large collection of already available unlabeled handwritten character data from several sources are often left unused. To alleviate the problem of small training sample size, we propose a graph-based active semi-supervised learning approach for handwritten character recognizer construction. The method iteratively builds a neighborhood graph of all examples including the unlabeled ones, assigns pseudo labels to the unlabeled data and retrain the model. Additionally, the label of the least confident pseudo label according to a newly proposed uncertainty measure is to be requested from the oracle. Experiments on NIST handwritten digits dataset demonstrated that the proposed learning method better utilizes the unlabeled data compared to existing approaches as measured by recognition accuracy. In addition, our active learning strategy is also more effective compared to baseline strategies.

Keywords: Handwritten character recognition
Semi-supervised learning · Active learning

1 Introduction

Optical character recognition (OCR) is a well-known automatic method for transforming digital images to machine encoded texts. Several OCR techniques have been developed during the past decade. Recognition of handwritten characters is a challenging task of the OCR process partly due to the limited number of labeled training data. In practice, characters sample used to train a recognition model is usually obtained from a small subset of writers. The problem is more pronounced in the case of ancient alphabet [1, 6]. As such, we cannot expect

the model to generalize well in real-world usage. Fortunately, we can still easily collect unlabeled sample from several sources such as digitized manuscripts, and this motivates exploring the possibilities of utilizing the unlabeled data to complement the model learning.

One way to tackle the problem is to adopt a semi-supervised learning (SSL) approach. SSL is halfway between supervised and unsupervised learning. It considers the situation that very few labeled examples are given while a large set of unlabeled examples is available. Often, the labeled sample size is so small that it cannot be used to construct a good classifier. SSL aims at extracting valuable information from the unlabeled instances, e.g., the distribution of the data [10] or relationships between labeled and unlabeled examples [3, 15–17], which can be used in conjunction with labeled data during training. Among several SSL techniques that have been proposed to date (for a comprehensive survey of SSL please refer to [4]), graph-based SSL is gaining more attention because of their high classification performance on many application domains [11]. Various graph-based SSL algorithms have been proposed in the literature such as label propagation [16], Gaussian random fields and harmonic function [17], local and global consistency [15], graph min-cut [3], and manifold regularization [2].

To improve label complexity even further, an Active Learning (AL) can also be employed. Unlike passive learning machine, AL-based learner may ask for the labels of some unlabeled examples from the oracle [8]. It has been shown that an active learning machine achieves comparable or even better classification accuracy with fewer training labels compared to a passive learner. Most of the work on active learning differ on how unlabeled instances are chosen for labeling. The criteria could be, for example, based on entropy, expected error reduction or variance reduction [14].

Recently, a combination of AL and SSL was proposed by Zhu et al. [18]. They combined AL with graph-based SSL. The AL operates on top of the SSL by greedily selecting queries from the unlabeled data to minimize the estimated expected classification error. The downside of the approach is that calculating the expected classification error is rather computationally expensive as it needs to consider all possible class assignments of the unlabeled examples. A semi-supervised active learning that selects query points based on the predicted class posterior probabilities was presented in [12]. The method tries to resolve the label of unlabeled instance whose prediction is the least confident. Active learning strategies based on entropy were also studied in [5, 9] in the context of natural language processing. A recent comparative study of different query point selection strategies seem to suggest that entropy-based selection method is among the top performers on average [14].

In this work, we propose a novel multi-class active semi-supervised learning framework for handwritten character recognition. Our approach is an extension of the graph-based SSL proposed by [17] to support multi-class classification. A new uncertainty measure based on Shannon entropy is also proposed for query point selection strategy of active learning. In summary, the key contributions of this work are the following.

- Proposal of a novel multi-class active semi-supervised learning framework for handwritten character recognition.
- Proposal of a new uncertainty measure based on Shannon entropy for using as the criterion of uncertainty sampling strategy of AL.
- Evaluation and analysis of the recognition performance of the proposed framework on modified NIST handwritten digits dataset reflecting the situation where labeled training data is scarce.

The rest of the paper is organized as follows. Section 2 formally introduces semi-supervised learning and graph-based semi-supervised learning for binary classification. Section 3 describes the proposed active semi-supervised HCR framework for multi-class HCR problems. Section 4 presents the dataset used in this study and provides the empirical results and discussion. Finally, Sect. 5 concludes the study.

2 Background and Problem Setting

2.1 Semi-supervised Learning

Semi-supervised classifier learning concerns the task of inferring a real-valued function $f : X \rightarrow Y$ using a set of labeled examples $D_L = \{(x_i, y_i)_{i=1}^L\}$ drawn i.i.d from some joint distribution $P(X, Y)$, and a set of unlabeled examples $D_U = \{(x_i)_{i=L+1}^N\}$ from the marginal distribution $P(X)$ so that we can use the classifier f to predict class label of an unseen example, x_q , drawn from the same data distribution with high accuracy. Here x_i denotes a data instance and y_i is its label assignment. The value of y_i is taken from a finite set C . In practice, we often find ourself in the situation that $|D_U| \gg |D_L|$. The main question is how to effectively leverage useful information contained in D_U to help the learning of f .

2.2 Graph-Based Semi-supervised Learning

One way to make use of unlabeled examples in classifier learning is through the SSL scheme. Recent SSL methods focuses on graph-based approach due to its superior empirical performance. The graph-based SSL expresses the geometry of examples as a weighted graph $G = (V, W)$ where the nodes $V = \{x_1, \dots, x_L, x_{L+1}, \dots, x_N\}$ are data points. The matrix $W = [w_{ij}]_{N \times N}$ is the weight matrix encoding similarities between the nodes. One example of the widely used similarity measure is the Gaussian kernel with bandwidth σ , where we have, $w_{ij} = \exp(-d(x_i, x_j)^2/2\sigma^2)$. The function $d(x_i, x_j)$ measures the distance between x_i and x_j .

Graph-based SSL assumes that connected nodes with high similarity value tend to have the same label. With this assumption, the labels of the unlabeled data could be inferred. A notable graph-based SSL method is probably the one proposed in [17] called GRF-SSL. Initially developed for binary classification problem, GRF-SSL makes use of Gaussian random fields and harmonic function

to infer the labels of the unlabeled nodes. Since our method is the extension of GRF-SSL to multi-class problem, it is worth describing the working of GRF-SSL.

Consider a binary classification problem, GRF-SSL seeks for a labeling function f that minimizes a weighted classification loss known as the energy function.

$$\Phi(f) = \frac{1}{2} \sum_{i,j} w_{ij} (f(x_i) - f(x_j))^2 \quad (1)$$

For $x_i \in D_L$, the output of $f(x_i)$ is simply its label assignment, y_i , and is fixed throughout the learning. For $x_i \in D_U$, the output from f is a ‘soft’ label, i.e., $y_i \in [0, 1]$. For clarity, f acting on unlabeled examples is named f_u . In effect, since labeling function for labeled examples does not change, the objective of GRF-SSL is to find the optimal f_u .

To do that, the Gaussian field $p_\beta(f) = \exp(-\beta\Phi(f))/Z_\beta$ is used to represent the probability distribution on f . Here β is an inverse temperature parameter and Z_β is the partition function. It has been shown that f which minimizes $\Phi(f)$ is harmonic; that is, its Laplacian is zero on unlabeled examples and is equal to $y_i : i = 1, \dots, L$ on labeled examples [17]. The Laplacian is defined as $\Delta = D - W$ where D is a diagonal matrix with $d_i = \sum_j w_{ij}$ and W is the weight matrix. By partitioning the Laplacian matrix Δ into 4 blocks of labeled and unlabeled examples,

$$\Delta = \begin{bmatrix} \Delta_{LL} & \Delta_{LU} \\ \Delta_{UL} & \Delta_{UU} \end{bmatrix}, \quad (2)$$

and letting $f = \begin{bmatrix} f_l \\ f_u \end{bmatrix}$, where f_l is a vector of the given labels, the optimal f_u can be found by

$$f_u = -\Delta_{UU}^{-1} \Delta_{UL} f_l. \quad (3)$$

3 The Proposed Framework

We shall start with an overview of the working of our proposed method which is shown in Fig. 1. In the first step, the labeled and unlabeled examples are feed into a graph-based SSL algorithm for constructing a recognition model. The model will then output pseudo ‘soft’ labels for the unlabeled examples, which essentially are their class posterior probabilities. The probabilities are utilized by an AL method in the calculation of query point selection criteria. The AL will then request a label of the query point from the oracle. The newly obtained labeled example is again feedbacked to the SSL algorithm for updating the recognition model. The aforementioned process is iteratively performed until some predefined condition is met i.e., query budget ran out. Our recognition model is an extension of the GRF-SSL [17] to multi-class problem, while our AL method query selection strategy is based on Shanon entropy. We are now in a position to present the details of the proposed method.

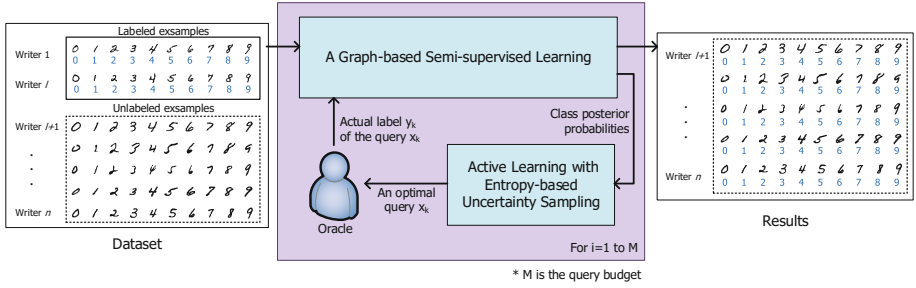


Fig. 1. The overview of the working of the proposed active semi-supervised handwritten character recognition model

3.1 Graph-Based Semi-supervised Learning for Multiclass Problem

In order to employ GRF-SSL in multi-class problem, we adopted the one-versus-all strategy. For this purpose, we train $|C|$ different labeling functions denoted by $f^j(\cdot)$, $j = 1, \dots, |C|$. Each $f^j(\cdot)$ is responsible for predicting if x_i is from class j . The label representation of examples in D_L is redefined to be a 0–1 vector of length $|C|$. It then follows that $y_i^j = 1$ (reads the j -the component of label i) if x_i is from class j and $y_i^j = 0$ otherwise. The modification for unlabeled examples is similar except that f_u^j outputs a ‘soft’ label. The value of $f^j(x_i)$ can be interpreted as a probability that a particle starting from node i on the graph G will reach a labeled node with label j . In other words, we can think of $f^j(x_i)$ as a posterior probability of class j , i.e., $p(y_i = j|x_i)$.

In our case, the character image will be represented by its Histogram of Oriented Gradients (HOG), and the Chi-Square distance will be the distance metric for comparing two histograms. We shall demonstrate in the experiment that the Chi-Square distance measure is indeed more suitable for the task than the standard Euclidean distance normally used in graph-based SSL. The Chi-Square distance between histogram P and Q in K -dimension can be computed by:

$$d(P, Q) = \frac{1}{2} \sum_{k=1}^K \frac{(P_k - Q_k)^2}{P_k + Q_k}. \quad (4)$$

3.2 Active Learning with Entropy-Based Uncertainty Sampling

To perform AL on top of the GRF-SSL, we adopted the uncertainty sampling strategy that directly utilizes the class posterior probability $p(y|x)$ provided by the recognition model. According to the meaning of the posterior probability, we can say that the class label of x is ‘not’ j with probability $1 - p(y = j|x)$. Let \hat{y} be the predicted class label, and $1 - p(y = j|x)$ for all $j \in C \setminus \{\hat{y}\}$ the *supporting probabilities*. For example, if $C = \{1, 2, 3\}$ and $\hat{y} = 1$, we will have $1 - p(y = 2|x)$ and $1 - p(y = 3|x)$ as the supporting probabilities. Accordingly, we defined an uncertainty measure called Entropy-based Uncertainty (EU) for an input x as follows:

$$EU(x) = -\left(p(y = \hat{y}|x) \log p(y = \hat{y}|x) + \sum_{j \in C \setminus \{\hat{y}\}} (1 - p(y = j|x)) \log(1 - p(y = j|x))\right), \quad (5)$$

The value of $EU(x)$ is equal or greater than 0. It indicates the uncertainty of predicting \hat{y} for x . The AL we performed in this work is the iterative process of selecting the next query point x_k based on the EU:

$$x_k = \arg \max_{x \in D_U} EU(x). \quad (6)$$

After x_k is assigned a label by the oracle and is added into the set of labeled data points D_L , the harmonic function f_u with x_k removed from D_U must be recomputed (using Eq. 3). Still, a recent technique for updating the harmonic function f_u after adding (x_q, y_q) to D_L was proposed in [18] where f_u can be efficiently updated by

$$f_u^{new} = f_u^{old} + (y_k - f_u^{old}(k)) \frac{\Delta_{UU}^{-1}(:, k)}{\Delta_{UU}^{-1}(k, k)}, \quad (7)$$

where $\Delta_{UU}^{-1}(:, k)$ is the k -th column of the invert Laplacian on unlabeled data, and $\Delta_{UU}^{-1}(k, k)$ is the k -th diagonal element of the same matrix.

4 Experiment and Results

4.1 Dataset

In our experiments, we used a handwritten digits dataset to evaluate recognition performance of the proposed framework. We randomly sampled digit characters data from NIST Special Database 19 [7], which consists of 810,000 handwritten digits from 3,600 writers. The resulting dataset contains 12,430 characters of handwritten digits from 116 writers with 100 examples per writer. The characters are evenly distributed into 10 classes. The input characters were centered in a 32×32 image by computing the center of mass. We represented each input character by its HOG descriptor with 4×4 blocks and 8 bins of gradient orientations for each block, the length of features vector is 128 ($4 \times 4 \times 8$).

4.2 Results and Discussion

We firstly evaluated the performance of GRF-SSL algorithm with two other graph-based SSL methods namely local and global consistency (LGC) [15] and greedy gradient Max-Cut (GGMC) [13]. For each of the graph-based SSL algorithm we also compared two node distance measures namely the Euclidean distance and the Chi-Square distance. To construct the graph used by all the graph-based SSL methods, the bandwidth parameter σ was empirically set to 1.25. The labeled examples are taken from the set of s random writers with s varies in $\{1, 2, 3, 4, 6, 8, 10, 12, 14, 16, 18\}$, while data from the remaining writers are used as unlabeled set. We performed 20 trials for each experimental setup.

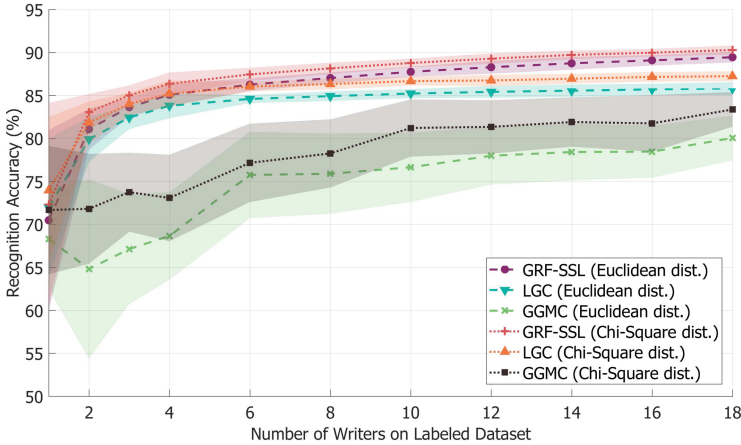


Fig. 2. Comparison of recognition performances of various graph-based semi-supervised learning methods employing the Euclidean distance and the Chi-Square distance as the distance metric.

The average recognition accuracies of the 20 trials provided by the GRF-SSL and comparing methods are shown in Fig. 2. As can be seen from the figure, the recognition accuracies provided by GRF-SSL using the Chi-Square distance were higher than that of GRF-SSL using the Euclidean distance. The effectiveness of the Chi-Square distance was expected since our character image is represented by HOG. This evidence indicates that the Chi-Square distance is more appropriate for this type of data representation. Comparing the accuracies among various graph-based SSL methods using the Chi-Square distance measure, we found that LGC achieved the highest accuracy when the labeled sample is from one writer, in which case the accuracy provided by the GRF-SSL is slightly lower. The Wilcoxon signed rank test was performed to verify the significance of the difference between accuracies provided by LGC and GRF-SSL. We found that the difference was not significant with p -value of 0.0617 at 0.05 significance level. By adding characters from more writers to the labeled set, all the recognition accuracies provided by all comparing methods improved. The GRF-SSL achieved the highest accuracy when it was trained using labeled sample from two and three writers. It still provided higher accuracy than others methods as labeled sample size increases. The above empirical results seem to suggest that the GRF-SSL with Chi-Square distance measure is suitable for HCR when character image is represented by HOG.

We continue to evaluate the combination of GRF-SSL using Chi-Square distance and AL using EU. Again we repeated the experiment for 20 trials. In each trial, the GRF-SSL was initialized with labeled data from one writer, and 50 label queries were allowed. The average recognition accuracies in the dynamic of number of label queries are shown in Fig. 3. Compared to entropy and random selection strategies, we see that the performance of EU is higher than that of

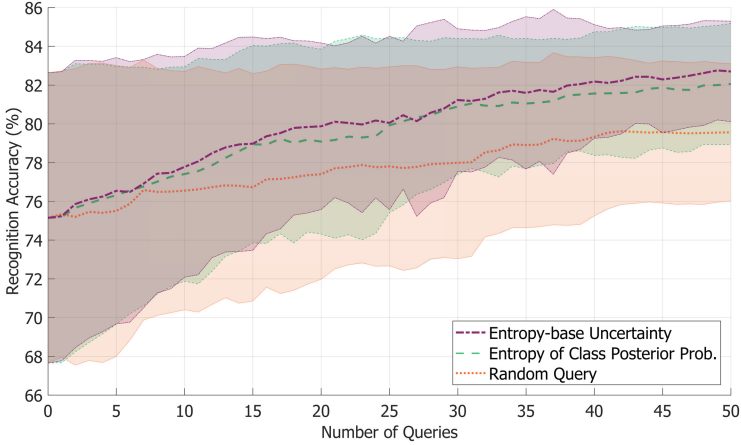


Fig. 3. Recognition performance of active learning using uncertainty sampling with entropy-based uncertainty, entropy of posterior probabilities and random query. The semi-supervised learning using Gaussian random fields and harmonic function is used as the recognition model.

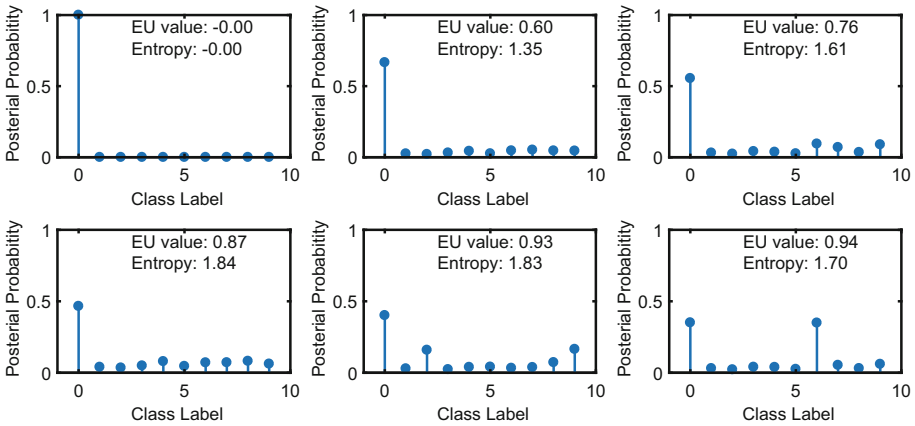


Fig. 4. Characteristics of Entropy-based Uncertainty (EU) and Shannon entropy measures under various class posterior probabilities settings.

random selection. Interestingly, the recognition accuracy provided by the EU is totally higher than that provided by entropy query method that directly calculating Shannon entropy from the posterior probabilities. To understand why EU works better than Shannon entropy, Fig. 4 illustrates the characteristics of the two uncertainty measures in various situations. We see that EU and Shannon entropy have the same trend when the posterior probability of first most probable class is larger than those of the remaining classes by a large margin (top row).

However, their trends are disagreeing when the posterior probabilities of all classes are too similar (bottom row). It seems that the EU measure we proposed well describes the uncertainty of prediction, compared to the Shannon entropy directly calculated from the posterior probabilities. The empirical results show that combining GRF-SSL and AL using EU as query point selection criteria performs well on HCR task.

5 Conclusion

We have proposed a recognition framework of handwritten character by combining SSL and AL. To address the limitation of labeled training examples, we applied a GRF-SSL as the recognition model. An AL using EU query point selection strategy was utilized to increase the number of labeled data points. In the experiment, a dataset of handwritten digits was used to evaluate the proposed framework. The experimental results showed that GRF-SSL using Chi-Square distance outperforms other methods for recognition of handwritten characters when the training data is limited. In addition, the combination of GRF-SSL and AL further improves label complexity. We believe that extending the approach to support rare ancient character recognition task is an interesting avenue for future study.

Acknowledgments. This study was funded by the Thailand Research Fund under the Royal Golden Jubilee Ph.D. Program (Grant No. PHD/0185/2556). International College of Digital Innovation, Chiang Mai University and the Department of Computer Science, Faculty of Science, Chiang Mai University provide computing facilities.

References

1. Antony, P.J., Savitha, C.K.: A framework for recognition of handwritten south dravidian tulu script. In: 2016 Conference on Advances in Signal Processing (CASP), pp. 7–12 (2016)
2. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.* **7**, 2399–2434 (2006)
3. Blum, A., Chawla, S.: Learning from labeled and unlabeled data using graph mincuts. In: Proceedings of the Eighteenth International Conference on Machine Learning, ICML 2001, pp. 19–26. Morgan Kaufmann Publishers Inc., San Francisco (2001)
4. Chapelle, O., Schölkopf, B., Zien, A.: *Semi-Supervised Learning*, 1st edn. The MIT Press, Cambridge (2010)
5. Hwa, R.: Sample selection for statistical parsing. *Comput. Linguist.* **30**(3), 253–276 (2004)
6. Inkeaw, P., Charoenkwan, P., Huang, H.L., Marukatat, S., Ho, S.Y., Chaijaruwanich, J.: Recognition of handwritten lanna dhamma characters using a set of optimally designed moment features. *Int. J. Doc. Anal. Recognit.* **20**(4), 259–274 (2017)

7. The National Institute of Standards and Technology: NIST special database 19 (2016). <http://www.nist.gov/srd/nistsd19.cfm>. Accessed 22 Apr 2018
8. Settles, B.: Active learning literature survey. Technical report, University of Wisconsin-Madison (2010)
9. Settles, B., Craven, M.: An analysis of active learning strategies for sequence labeling tasks. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP 2008, pp. 1070–1079. Association for Computational Linguistics, Stroudsburg (2008)
10. Soares, R.G.F., Chen, H., Yao, X.: Semisupervised classification with cluster regularization. *IEEE Trans. Neural Netw. Learn. Syst.* **23**(11), 1779–1792 (2012)
11. Sousa, C.A.R.: An overview on the Gaussian fields and harmonic functions method for semi-supervised learning. In: 2015 International Joint Conference on Neural Networks (IJCNN), pp. 1–8 (2015)
12. Tomanek, K., Hahn, U.: Semi-supervised active learning for sequence labeling. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2, vol. 2, pp. 1039–1047. ACL 2009, Association for Computational Linguistics, Stroudsburg (2009)
13. Wang, J., Jebara, T., Chang, S.F.: Semi-supervised learning using greedy max-cut. *J. Mach. Learn. Res.* **14**(1), 771–800 (2013)
14. Yang, Y., Loog, M.: A benchmark and comparison of active learning for logistic regression. arXiv preprint [arXiv:1611.08618](https://arxiv.org/abs/1611.08618) (2016)
15. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: Proceedings of the 16th International Conference on Neural Information Processing Systems, NIPS 2003, pp. 321–328. MIT Press, Cambridge (2003)
16. Zhu, X., Ghahramani, Z.: Learning from labeled and unlabeled data with label propagation. Technical report, School of Computer Science, Carnegie Mellon University (2002)
17. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using Gaussian fields and harmonic functions. In: Proceedings of the Twentieth International Conference on Machine Learning, pp. 912–919 (2003)
18. Zhu, X., Lafferty, J., Ghahramani, Z.: Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In: ICML 2003 workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining, pp. 58–65 (2003)



Differential Evolution for Association Rule Mining Using Categorical and Numerical Attributes

Iztok Fister Jr.^{1(✉)}, Andres Iglesias^{2,3}, Akemi Galvez^{2,3}, Javier Del Ser^{4,5,6}, Eneko Osaba⁴, and Iztok Fister^{1,3}

¹ Faculty of Electrical Engineering and Computer Science, University of Maribor, Koroška cesta 46, Maribor, Slovenia

iztok.fister1@um.si

² Toho University, 2-2-1 Miyama, 274-8510 Funabashi, Japan

³ University of Cantabria, Avenida de los Castros, s/n, 39005 Santander, Spain

⁴ TECNALIA, Derio, Spain

⁵ University of the Basque Country (UPV/EHU), Bilbao, Spain

⁶ Basque Center for Applied Mathematics (BCAM), Bilbao, Spain

Abstract. Association rule mining is a method for identification of dependence rules between features in a transaction database. In the past years, researchers applied the method using features consisting of categorical attributes. Rarely, numerical attributes were used in these studies. In this paper, we present a novel approach for mining association based on differential evolution, where features consist of numerical as well as categorical attributes. Thus, the problem is presented as a single objective optimization problem, where support and confidence of association rules are combined into a fitness function in order to determine the quality of the mined association rules. Initial experiments on sport data show that the proposed solution is promising for future development. Further challenges and problems are also exposed in this paper.

Keywords: Association rule mining · Classification
Differential evolution · Evolutionary computation

1 Introduction

Data mining methods are intended to infer new insights (knowledge) from a bunch of structured or unstructured data. With the rise of big data on almost all areas of human endeavor, data mining methods have received a high priority within business and industry. In the past, many methods for various data mining tasks were developed, which are primarily devoted for the classification, clustering, regression as well as association rule mining (ARM). ARM is basically the process of identifying the dependence rules between features inside transaction databases. Therefore, ARM is appropriate for market basket analysis, analysis of human habits, driver strategies and so on.

© Springer Nature Switzerland AG 2018

H. Yin et al. (Eds.): IDEAL 2018, LNCS 11314, pp. 79–88, 2018.

https://doi.org/10.1007/978-3-030-03493-1_9

Researchers introduced association rule mining in the 90s with seminal work of Agrawal [3]. From that time, ARM was deeply studied in theory and practice. After more than 20 years of research, these methods have become matured. On the other hand, rising of the big data has demanded development of the new scalable methods that can easily be parallelized. There exist many ARM algorithms, as for example Apriori [3], Eclat [19], FP-growth [12].

Interestingly, researchers also developed some ARM methods that are based on stochastic population-based nature-inspired algorithms [4, 6, 7, 16], such as Particle Swarm Optimization (PSO) [14], Ant Colony Optimization (ACO) [8], Differential Evolution (DE) [17]. Main features of these algorithms is that they can easily be parallelized, while they basically also ensure scalability.

However, ARM was initially applied for mining categorical (i.e., non-numerical) features, where all features must be discretized before usage. Normally, this task is trivial due to a simple mapping of the attribute to intervals of real values representing the corresponding attributes. However, similar methods that are able to handle also numeric attributes were proposed in [2, 5, 11, 15]. Introducing the numerical attributes demands dealing with explicit intervals of real values for each numerical attribute of the feature.

Therefore, the purpose of the paper is to extend our previous work on ARM for mining categorical features based on bat algorithm, i.e. BatMiner [10] with new feature that allows mining combination of numerical data as well as categorical data. Additionally, the bat algorithm [18] is here replaced by differential evolution. There are two issues for this decision as follows: (1) To show how the evolutionary algorithms behaves by solving this problem, and (2) To find out how the operator of crossover affects the results of the mining. As already known, the bat algorithm works only using the mutation operator [9]. Algorithm is evaluated on sport data that consists of 14 features, where attributes of three features are numerical, while the attributes of the 11 other features represent categorical values.

In a nutshell, main contributions of this paper are as follows:

- A new differential evolution algorithm for mining association rules is proposed.
- The algorithm is capable of dealing with numerical and categorical features.
- The algorithm is tested on dataset that encompass habits of athlete in training.

The structure of this paper is as follows: In Sect. 2, the background information is discussed that is necessary for understanding the subjects in the remainder of the paper. Section 3 describes the proposed algorithm for ARM using mixed numerical and categorical feature attributes. In Sect. 4, experiments and results are illustrated, while the paper is concluded with summarizing the performed work and outlining the possible directions for the future.

2 Background Information

2.1 Association Rule Mining

This section briefly presents formal definition of ARM. Let us suppose, a set of objects $O = \{o_1, \dots, o_n\}$ and transaction set D are given, where each transaction T is a subset of objects, in other words $T \subseteq O$. Then, an association rule can be defined as implication:

$$X \Rightarrow Y, \quad (1)$$

where $X \subset O, Y \subset O$, in $X \cap Y = \emptyset$. The following two measures are defined for evaluating the quality of association rule [3]:

$$\text{conf}(X \Rightarrow Y) = \frac{n(X \cup Y)}{n(X)}, \quad (2)$$

$$\text{supp}(X \Rightarrow Y) = \frac{n(X \cup Y)}{N}, \quad (3)$$

where $\text{conf}(X \Rightarrow Y) \geq C_{min}$ denotes confidence and $\text{supp}(X \Rightarrow Y) \geq S_{min}$ support of association rule $X \Rightarrow Y$. Thus, N in Eq. (3) represent number of transactions in transaction database D and n is number of repetitions of particular rule $X \Rightarrow Y$ within D . Here, C_{min} denotes minimum confidence and S_{min} minimum support. This means that only those association rules with confidence and support higher than C_{min} and S_{min} are taken into consideration, respectively.

2.2 Differential Evolution Basics

Differential Evolution is an Evolutionary Algorithm appropriate for continuous as well as combinatorial optimization. This algorithm was introduced by Storn and Price in 1995 [17]. It is a population-based and consists of Np real-coded vectors representing the candidate solutions, as follows:

$$\mathbf{x}_i^{(t)} = (x_{i,1}^{(t)}, \dots, x_{i,n}^{(t)}), \quad \text{for } i = 1, \dots, Np, \quad (4)$$

where each element of the solution is in the interval $x_{i,1}^{(t)} \in [x_i^{(L)}, x_i^{(U)}]$, and $x_i^{(L)}$ and $x_i^{(U)}$ denotes the lower and upper bounds of the i -th variable, respectively.

The variation operator in DE supports a differential mutation and a differential crossover. In particular, the differential mutation selects two solutions randomly and adds a scaled difference between these to the third solution. This mutation is expressed as follows:

$$\mathbf{u}_i^{(t)} = \mathbf{x}_{r1}^{(t)} + F \cdot (\mathbf{x}_{r2}^{(t)} - \mathbf{x}_{r3}^{(t)}), \quad \text{for } i = 1, \dots, Np, \quad (5)$$

where F denotes the scaling factor as a positive real number that scales the rate of modification while $r1, r2, r3$ are randomly selected values in the interval $1 \dots Np$. Note that, typically, the interval $F \in [0.1, 1.0]$ is used in the DE community.

As a differential crossover, uniform crossover is employed by the DE, where the trial vector is built from parameter values copied from two different solutions. Mathematically, this crossover is expressed as follows:

$$w_{i,j}^{(t+1)} = \begin{cases} u_{i,j}^{(t)} & \text{rand}_j(0, 1) \leq CR \vee j = j_{rand}, \\ x_{i,j}^{(t)} & \text{otherwise,} \end{cases} \quad (6)$$

where $CR \in [0.0, 1.0]$ controls the fraction of parameters that are copied to the trial solution. Note, the relation $j = j_{rand}$ ensures that the trial vector is different from the original solution $\mathbf{x}_i^{(t)}$.

A differential selection is, in fact, a generalized one-to-one selection that is expressed mathematically as follows:

$$\mathbf{x}_i^{(t+1)} = \begin{cases} \mathbf{w}_i^{(t)} & \text{if } f(\mathbf{w}_i^{(t)}) \leq f(\mathbf{x}_i^{(t)}), \\ \mathbf{x}_i^{(t)} & \text{otherwise.} \end{cases} \quad (7)$$

In a technical sense, crossover and mutation can be performed in several ways in Differential Evolution. Therefore, a specific notation is used to describe the varieties of these methods (also strategies) generally. For example, ‘DE/rand/1/bin’ denotes that the base vector is selected randomly, 1 vector difference is added to it, and the number of modified parameters in the mutant vector follows binomial distribution.

3 DE for ARM Using Numerical and Categorical Attributes

In this section, a new DE for ARM using mixed (i.e., numerical and categorical) attributes (ARM-DE) is proposed. Basically, development of this algorithm consists of the following stages:

- domain analysis,
- representation of solution,
- definition of a fitness function.

The aim of the first stage is the identification of features. In the second stage, the representation of solution is discussed, while a definition of the fitness function is illustrated in the last stage. In the remainder of the paper, the mentioned stages are presented in detail.

3.1 Domain Analysis

In this stage, we identify the features along with their attributes and corresponding domain values. Actually, there can be numerical as well as categorical attributes. If attributes are numerical values, minimum lower and maximum upper bounds are prescribed determining intervals of domain values, from which the minimum and maximum values for each particular numerical attribute can

be drawn. For categorical attributes, a set of feasible attributes needs to be enumerated, because of their discrete nature.

The minimum lower and maximum upper bounds, from which the numerical attributes can be drawn, are presented in Table 1. There are also eleven features with categorical attributes as illustrated in Table 2. The numerical attributes are presented in the tables as a particular feature with corresponding interval of domain values (i.e., DISTANCE \in [50, 100]), while the categorical attributes as the feature followed by an attribute preceded by point sign ‘.’ (i.e., CALORIES.SMALL).

Table 1. Numerical attributes and their domain values.

Feature	Attribute domain	
	Minimum lower bound	Maximum upper bound
DISTANCE	20	200
DURATION	20	330
HEART_RATE	80	185

Table 2. Categorical attributes and their domain values.

Feature	Attribute domain
CALORIES	{NULL, SMALL, MEDIUM, HIGH}
WEATHER	{NULL, SUNNY, CLOUDY, RAINY, SNOWY}
TYPE	{NULL, EASY, INTERVALS, POWER, ENDURANCE}
NUTRITION	{NULL, POOR, MODERATE, GOOD}
FOOD	{NULL, PROTEINS, CARBOHYDRATES, FAT, FRUITS}
BEVERAGES	{NULL, WATER, JUICE, ISO, COKE}
REST	{NULL, AFTER_TRAINING, NO}
NIGHT_REST	{NULL, BAD, MEDIUM, GOOD}
INJURIES	{NULL, NO, LOW, MEDIUM, HIGH}
CRAMPS	{NULL, NO, LOW, HIGH}
HEALTH_PROBLEMS	{NULL, NO, LITTLE, YES}

3.2 Representation of Solutions

Each individual is represented as a real-valued vector, where every numerical attribute is represented by a corresponding minimum and maximum boundaries determining domain values, from which the attribute can be drawn. In contrary, every categorical attribute is represented by the real-value drawn from interval [0, 1]. Additionally, the last element in the vector denotes the cut point. It means that this number says, which part of the vector belong to the antecedent and which to the consequence of the mined association rule.

In a nutshell, each solution is mathematically represented as the real-valued vector:

$$\mathbf{x}_i^{(t)} = \left\{ \underbrace{(x_{i,1}^{(t)}, x_{i,2}^{(t)})}_{Attr_1^{(num)}}, \dots, \underbrace{(x_{i,2n-1}^{(t)}, x_{i,2n}^{(t)})}_{Attr_n^{(num)}}, \underbrace{x_{i,2n+1}^{(t)}}_{Attr_{n+1}^{(cat)}}, \dots, \underbrace{x_{i,d}^{(t)}}_{Attr_{d-n}^{(cat)}}, \underbrace{x_{i,d+1}^{(t)}}_{\text{Cut point}} \right\}, \quad (8)$$

where $x_{i,j}^{(t)}$ for $i = 1, \dots, d$ codes attributes of features in association rule, $x_{i,d+1}^{(t)}$ denotes cut point and t is counter of iterations. Thus, each numerical attribute is represented as a pair $Attr_j^{(num)} = (x_{i,2j-1}^{(t)}, x_{i,2j}^{(t)})$ for $j = 1, \dots, n$, where $x_{i,2j-1}^{(t)} \in [Lb_{2j-1}]$ and $x_{i,2j}^{(t)} \in [Lb_{2j}]$, and the attribute is calculated according to the following equation:

$$Attr_j^{(num)} = \begin{cases} (NULL, NULL), & \text{if } |x_{i,2j-1}^{(t)} - x_{i,2j}^{(t)}| < \lfloor \frac{Ub_{2j-1} - Lb_{2j-1}}{50} \rfloor, \\ (x_{i,2j-1}^{(t)}, x_{i,2j}^{(t)}), & \text{if } x_{i,2j-1}^{(t)} > x_{i,2j}^{(t)}, \\ (x_{i,2j}^{(t)}, x_{i,2j-1}^{(t)}), & \text{otherwise.} \end{cases} \quad (9)$$

In Eq. 9, the Lb_{2j-1} and Ub_{2j-1} denote the lower and upper bound of the specific attribute value, respectively. On the other hand, the range of feasible values for categorical attributes in interval $x_{i,j}^{(t)} \in [0, 1]$ is divided into $m_j + 1$ equidistant intervals, where each interval $[k, k + 1)$ for $k = 0, \dots, m_j$ corresponds to one of the potential attributes $Attr_j^{(cat)} \in \{Attr_0, \dots, Attr_{m_j}\}$, and parameter m_j denotes the number of attributes belonging to the j -th feature. Actually, the categorical attribute is calculated according to the following equation:

$$Attr_j^{(cat)} = \lfloor \frac{x_{i,j}^{(t)}}{m_j + 1} \rfloor, \text{ for } j = n, \dots, d - n. \quad (10)$$

Attribute $Attr_0^{(cat)} = NULL$ has a special meaning, because it determines that the corresponding feature is not presented in the association rule.

Finally, the cut point is calculated according to the following equation:

$$cp_i^{(t)} = \lfloor x_{i,d+1}^{(t)} \cdot (d - n - 2) \rfloor + 1, \text{ for } i = 0, \dots, Np, \quad (11)$$

3.3 Definition of the Fitness Function

Fitness function is defined as follows [10, 13]:

$$f(\mathbf{x}_i^{(t)}) = \begin{cases} \alpha * conf(\mathbf{x}_i^{(t)}) + \gamma * supp(\mathbf{x}_i^{(t)}) / \alpha + \gamma, & \text{if } feasible(\mathbf{x}_i^{(t)}) = true, \\ -1, & \text{otherwise,} \end{cases} \quad (12)$$

where $conf()$ is confidence, $supp()$ support, α and γ are weights, function $feasible(\mathbf{x}_i)$, denotes if solution is feasible. The task of optimization is to find maximum value of fitness function.

4 Experiments and Results

The aim of experimental work was to test the performance of proposed ARM-DE in practice. In line with this, it is expected that the results of this algorithm make sense from the real sports trainer point of view. As mentioned before, ARM-DE is able to handle both categorical and numerical data stored in transaction database consisting of 14 features, where the first three features have numerical attributes (i.e., distance, duration and average heart rate), while the other eleven feature's attributes are represented by categorical data.

Let us mention that we taken the full set of attributes that can be obtained from the sport activity datasets. Obviously, each sport activity presents one transaction in transaction database. Unfortunately, the real data is hard to obtain. In the case, when the real activities can be gained from the athlete, the number of these activities is limited due to physical limitation of an athlete. The professional cyclist, for example, finishes typically one sports training per day. When we assume that one day is reserved for resting, this athlete can perform at most 300 sports activities per year. This number is rapidly decreased, if the amateur cyclist is taken into consideration that cannot be affordable to train each day. Therefore, the generator of sports training activities has been proposed by Fister et al. [1], with which these limitations can be circumvent. The generated sports activities were used also in our study.

Because of lack of real data, the transaction database consisting of 500 transactions were generated randomly, in this study. The following parameters were used in DE algorithm: $D = 19$, $NP = 100$, $FES = 70,000$, $F = 0.5$, $CR = 0.9$, where FES represents the number of fitness function evaluations. Thus, all feasible solutions obtained after 10 independent runs of the ARM-DE algorithm were accumulated, while the best results are presented in Table 3, and their corresponding quality measures in Table 4. Let us mention, that all numerical attributes in our experiments must be present in each association rule, otherwise this is considered as infeasible. Illustrated association rules referring to athlete's health condition are very similar between each other and say that cyclist overcoming moderate-distance courses in ultra-long duration and moderate intensity, in either endurance or interval type of training session, or drinking juice during

Table 3. Examples of the best solutions found by the ARM-DE algorithm.

Rule	Antecedent	Consequent
1	DISTANCE \in [25.80, 111.96] \wedge DURATION \in [31.08, 172.86] \wedge HEART_RATE \in [136.68, 180.52] \wedge WEATHER.SNOWY	REST.NO
2	DISTANCE \in [133.32, 220] \wedge DURATION \in [281.10, 315.19] HEART_RATE \in [104.62, 149.49] \wedge TYPE.ENDURANCE	HEALTH.PROBLEMS.NO
3	DISTANCE \in [160.14, 220] \wedge DURATION \in [181.22, 330.0] HEART_RATE \in [111.47, 139.56] \wedge TYPE.INTERVALS	HEALTH.PROBLEMS.NO
4	DISTANCE \in [121.37, 199.40] \wedge DURATION \in [187.66, 330] HEART_RATE \in [133.02, 168.80] \wedge BEVERAGES.JUICE \wedge REST.AFTER_TRAINING	HEALTH.PROBLEMS.NO

Table 4. Quality measures obtained by the best solutions found.

Rule	Fitness	Support	Confidence
1	0.519	0.038	1.0
2	0.516	0.032	1.0
3	0.512	0.024	1.0
4	0.511	0.022	1.0

and resting after the training probably should not have any health problems. Obviously, the rule is valid in real-world.

Similar as observed in our previous work [10], the best solutions usually have only one consequence. However, there are also more complex association rules in Table 5 with their corresponding quality measures in Table 6, where more attributes can be observed in association rules. The last association rule in the table is the most interesting and asserts that cyclist overcoming the medium-distance course in ultra-long duration with moderate intensity and high calorie consumption by performing the power training session should not have the injuries as well as cramps. This rule also hold in practice.

Table 5. Examples of more complex solutions that were found by ARM-DE algorithm.

Rule	Antecedent	Consequent
1	DISTANCE \in [20.0, 71.39] \wedge DURATION \in [178.55, 139.89] \wedge HEART_RATE \in [139.89, 180.18] \wedge CALORIES.HIGH \wedge TYPE.ENDURANCE	CRAMPS.NO \wedge HEALTH_PROBLEMS.NO
2	DISTANCE \in [85.03, 134.63] \wedge DURATION \in [81.81, 228.50] \wedge HEART_RATE \in [93.96, 146.05] \wedge NUTRITION.GOOD \wedge FOOD.PROTEINS \wedge BEVERAGES.JUICE \wedge REST.AFTER_TRAINING	CRAMPS.NO \wedge HEALTH_PROBLEMS.NO
3	DISTANCE \in [20.00, 86.60] \wedge DURATION \in [155.31, 161.87] \wedge HEART_RATE \in [161.87, 183.43] \wedge CALORIES.HIGH \wedge TYPE.POWER	INJURIES.NO \wedge CRAMPS.NO

Table 6. Quality measures obtained by the more complex solutions found.

Rule	Fitness	Support	Confidence
1	0.505	0.010	1.0
2	0.503	0.006	1.0
3	0.502	0.004	1.0

5 Conclusion and Future Challenges

Association rule mining with numerical attributes is a very challenging problem. In this paper, we try to solve the problem using differential evolution. Our proposed solution is capable to mine association rule, where features can consist of either numeric or categorical attributes. This assertion is evident by obtaining the highest confidence values of mined rules. Practical experiments on synthetically generated data showed that this approach is interesting, but there are still many problems that should be elaborated in future, i.e. how to shrink lower and upper borders on numerical attributes, how to better evaluate these borders, testing the algorithm on bigger transaction databases with more features as well as applying this approach to other population-based nature-inspired algorithms.

Acknowledgment. I. Fister Jr. and I. Fister acknowledge the financial support from the Slovenian Research Agency (Research Core Fundings No. P2-0041 and P2-0057). A. Iglesias and A. Galvez acknowledge the financial support from the projects #TIN2017-89275-R (AEI/FEDER, UE), and #JU12 (SODERCAN/FEDER UE). E. Osaba and J. Del Ser would like to thank the Basque Government for its funding support through the EMAITEK program.

References

1. SportyDataGen. <http://www.sport-slo.net/>. Accessed 30 May 2018
2. Agbehadji, I.E., Fong, S., Millham, R.: Wolf search algorithm for numeric association rule mining. In: 2016 IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), pp. 146–151. IEEE (2016)
3. Agrawal, R., Srikant, R., et al.: Fast algorithms for mining association rules. In: Proceedings of 28th International Conference on Very Large Data Bases, VLDB, vol. 1215, pp. 487–499 (1994)
4. Alatas, B., Akin, E.: An efficient genetic algorithm for automated mining of both positive and negative quantitative association rules. *Soft Comput.* **10**(3), 230–237 (2006)
5. Alatas, B., Akin, E.: Rough particle swarm optimization and its applications in data mining. *Soft Comput.* **12**(12), 1205–1218 (2008)
6. Alatas, B., Akin, E.: Multi-objective rule mining using a chaotic particle swarm optimization algorithm. *Knowl.-Based Syst.* **22**(6), 455–460 (2009)
7. Chiclana, F., Kumar, R., Mittal, M., Khari, M., Chatterjee, J.M., Baik, S.W., et al.: ARM-AMO: an efficient association rule mining algorithm based on animal migration optimization. *Knowl.-Based Syst.* **154**, 68–80 (2018)
8. Dorigo, M., Blum, C.: Ant colony optimization theory: a survey. *Theor. Comput. Sci.* **344**(2), 243–278 (2005)
9. Fister, I., Brest, J., Mlakar, U.: Towards the universal framework of stochastic nature-inspired population-based algorithms. In: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1–8, December 2016
10. Fister Jr., I., Fister, I.: Algoritem batminer za rudarjenje asociativnih pravil. *Presek* **44**(5), 26–29 (2017)

11. Fukuda, T., Morimoto, Y., Morishita, S., Tokuyama, T.: Mining optimized association rules for numeric attributes. In: Proceedings of the Fifteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 182–191. ACM (1996)
12. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: ACM Sigmod Record, vol. 29, pp. 1–12. ACM (2000)
13. Heraguemi, K.E., Kamel, N., Drias, H.: Association rule mining based on bat algorithm. In: Pan, L., Păun, G., Pérez-Jiménez, M.J., Song, T. (eds.) BIC-TA 2014. CCIS, vol. 472, pp. 182–186. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45049-9_29
14. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948. IEEE (1995)
15. Minaei-Bidgoli, B., Barmaki, R., Nasiri, M.: Mining numerical association rules via multi-objective genetic algorithms. *Inf. Sci.* **233**, 15–24 (2013)
16. Mlakar, U., Zorman, M., Fister Jr., I., Fister, I.: Modified binary cuckoo search for association rule mining. *J. Intell. Fuzzy Syst.* **32**(6), 4319–4330 (2017)
17. Storn, R., Price, K.: Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **11**(4), 341–359 (1997)
18. Yang, X.-S.: *Nature-Inspired Metaheuristic Algorithms*. Luniver press, Beckington (2010)
19. Zaki, M.J.: Scalable algorithms for association mining. *IEEE Trans. Knowl. Data Eng.* **12**(3), 372–390 (2000)



Predicting Wind Energy Generation with Recurrent Neural Networks

Jaume Manero^{1(✉)}, Javier Béjar¹, and Ulises Cortés^{1,2}

¹ Universitat Politècnica de Catalunya - BarcelonaTECH, Barcelona, Spain

jaume.manero@upc.edu, {bejar,ia}@cs.upc.edu

² Barcelona SuperComputing Center, Barcelona, Spain

Abstract. Decarbonizing the energy supply requires extensive use of renewable generation. Their intermittent nature requires to obtain accurate forecasts of future generation, at short, mid and long term. Wind Energy generation prediction is based on the ability to forecast wind intensity. This problem has been approached using two families of methods one based on weather forecasting input (Numerical Weather Model Prediction) and the other based on past observations (time series forecasting). This work deals with the application of Deep Learning to wind time series. Wind Time series are non-linear and non-stationary, making their forecasting very challenging. Deep neural networks have shown their success recently for problems involving sequences with non-linear behavior. In this work, we perform experiments comparing the capability of different neural network architectures for multi-step forecasting in a 12 h ahead prediction. For the Time Series input we used the US National Renewable Energy Laboratory's WIND Dataset [3], (the largest available wind and energy dataset with over 120,000 physical wind sites), this dataset is evenly spread across all the North America geography which has allowed us to obtain conclusions on the relationship between physical site complexity and forecast accuracy. In the preliminary results of this work it can be seen a relationship between the error (measured as R^2) and the complexity of the terrain, and a better accuracy score by some Recurrent Neural Network Architectures.

Keywords: Time series · Recurrent Neural Networks

Multi-step prediction · Seq2Seq · Wind forecast · NREL dataset

Wind energy prediction

This work is partially supported by the Joint Study Agreement no. W156463 under the IBM/BSC Deep Learning Center agreement, by the Spanish Government through Programa Severo Ochoa (SEV-2015-0493), by the Spanish Ministry of Science and Technology through TIN2015-65316-P project, and by the Generalitat de Catalunya (contracts 2014-SGR-1051).

© Springer Nature Switzerland AG 2018

H. Yin et al. (Eds.): IDEAL 2018, LNCS 11314, pp. 89–98, 2018.

https://doi.org/10.1007/978-3-030-03493-1_10

1 Introduction

Wind power generation is already a critical contributor to the electrical supply systems in many countries around the world. We can cite some nations with high wind penetration in their electricity generation mix (as a percentage of total production in 2016) like Denmark (36,8%), Ireland (27%), Portugal (24,7%), Spain (19%) or Germany (16%) [13]. This penetration, already relevant, will see a steep increase in the next few years due to the increased efforts being performed by most countries in order to accelerate the transition to a CO₂ free energy model [12] trying to reduce and stop the negative impact of the greenhouse gases in our atmosphere.

The integration of renewable energy into the grid is complex due to their inherent intermittent nature. The electricity grid has to assure continuous supply using all the different generation technologies available, and for this reason developing forecasting techniques (at demand and production sides of the system) is critical for its stability. Forecasting the energy generation output for the renewable generation assets becomes a core task in the energy management.

Wind energy forecasting has not only value for its contribution to the system stability but also has additional potential for savings in the overall wind-production life-cycle. In this direction, it has been established that a small increase of 10% in the quality of prediction would be able to generate savings of 140 million US\$ in the United States overall system [7].

Many methods have been designed for wind prediction, in two major families: methods based in meteorology input (NWP) and Time Series methods that use only the information from observations in the turbine sensors. This work deals with the second category, the wind time series data. Commercial systems use combinations of both families of methods, but this work will focus only in the learning that can be obtained from Wind Time series past observations. This work deals with the application of Recurrent Neural Networks to the wind time series.

In order to explore these series, large datasets are required. In this work the Wind dataset from the U.S. National Renewable Laboratory (NREL) has been used, A dataset with 120,000 wind points that is the largest available wind dataset in the world.

2 The Nature of Wind Time Series

Wind turbines are provided with hundreds of sensors that generate information in real time, creating a stream of data that is stored and analyzed. This information has many dimensions like energy generated, performance of the internal engines or meteorological data about the environment (wind speed, temperature, pressure, etc.). These data are used to monitor the functioning of the turbine and are a valuable input for prediction. For mid-term prediction the information of the sensors is converted into time series with readings every 5 to 10 min.

Typically, a wind time series will be a time-stamped sequence of several measures that can be related to wind. The dimensions are usually; wind power

(MW), wind direction (*degrees*), air pressure (Pa), wind speed (m/s), temperature (C or K), air density (kg/m^3), relative humidity (%). All these observations can be generated at different heights (floor, hub height, half height). As the wind at 100 m high (hub turbine height) is the one that moves the blades, it is probably the measure with the highest relevance, while wind direction is important to understand how the dominant winds might impact wind patterns and intensity. In Fig. 1 a summary of one-year data from the Sotavento wind park is shown in the wind rose, the dominance of E/NE and W/SW winds is clear on this site.

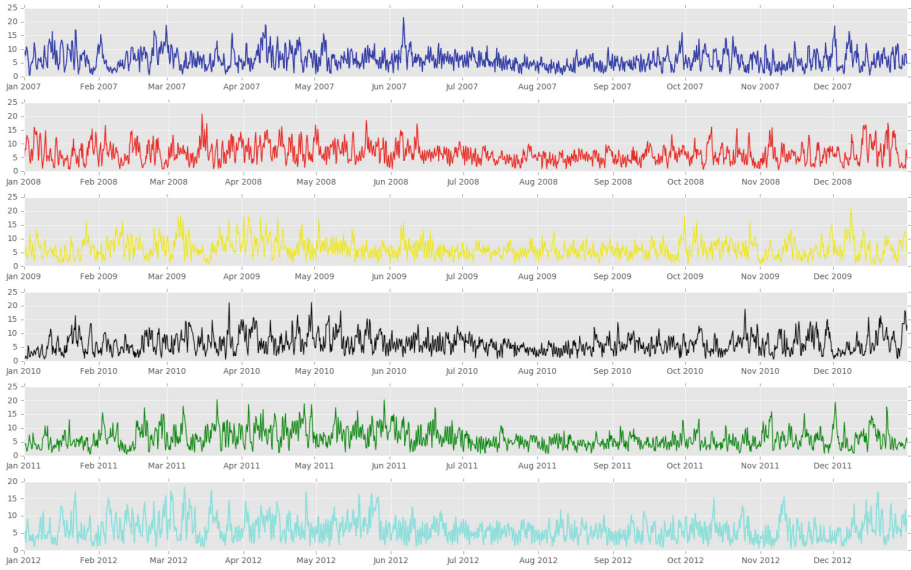


Fig. 1. Wind speed time series in site located in Techado New Mexico USA, vertical axis wind speed in m/s, horizontal axis time. Data from NREL Dataset [3]

Wind is a natural phenomenon that is created by various forces applied to the atmosphere at the same time, namely: the pressure gradient force, the frictional force, the Coriolis force and the gravitational force. For the energy forecast task in wind turbines, only winds close to the surface are studied, and those are impacted by the frictional force, which will depend on the specific orography of the site [6]. It is well-known that wind may vary in two locations not far away. It can be seen in a wind park the different speed of the blades in similar turbines or some turbines idle (no wind) while some others are turning, this shows empirically the wind variation in very closed locations.

But not only orography is relevant for the wind formation. The earth science has already stated that wind is the combination of periodical phenomena like day/night or summer/winter, a result of low/high-pressure variations and all of them combined with temperature, air density and pressure. The combination of all these factors is of high complexity and the result, over time, is the wind

as we know it. For this reason, it is quite usual that in a wind time series all these factors are overlapped (a storm in summer at night from the north), and extracting each factor is of high complexity (if possible at all) (Fig. 2).

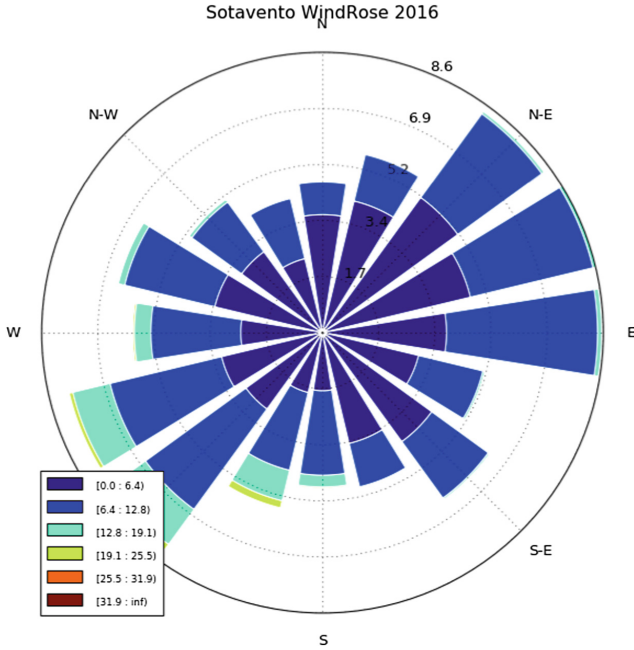


Fig. 2. Wind direction dimension in one year of time series measurements in the Sotavento Park located in Galicia, Spain 2016 [2]

2.1 Some Statistical Properties of the Wind Time Series

Stationarity in a time series is understood as the property where the statistical characteristics such as mean, variance, autocorrelation, etc. are all constant over time, or repeat over time in some sequences (seasonal, day/night,...).

There are several tests widely used to analyze the stationarity of a time series. The Dick-Füller (ADF) test (and its evolution the augmented ADF) are the most common. The ADF looks for a *unit root* in a time series sample. A *unit root* is a statistical feature that determines randomness in the series. Applying these tests to the dataset we obtain non-stationarity results.

Linearity is another relevant property to be found in the wind time series. Linearity will allow the use of linear forecasting methods and non-linearity needs of more complex methods (non-linear) have to be used to obtain accurate predictions. The validation of linearity in a time series is not an easy and straightforward task. The surrogate data method, described by Theiler in [11] is a powerful tool to validate linearity. This test applied to wind time series shows that linearity can be found in some wind datasets but not in all of them [4].

If the wind is nonlinear, how can linear models be used for forecasting? The answer lies in the fact that the wind series contains inner structures that might be linear. The best forecasting methods will extract this information (*learn*) the shape of these internal structures to produce more accurate results.

3 The Wind Energy Generation Forecasting Task

Energy in the turbines is generated from the kinetic energy of wind. The action of wind moves the blades and generates a rotational effect which produces electricity (by the Faraday law). In the field, wind turbines are usually grouped in wind parks that can range from a few turbines up to hundreds of them to leverage areas where the wind is steady and strong over the whole year. The power generated by a wind turbine (see Eq. 1) is directly dependent on the swept area of the Blade (A), or and in the Air density (ρ), but mainly on the airspeed (v^3).

$$Energy = \frac{1}{2}\rho A t v^3; \quad (Power = \frac{E}{t}) \tag{1}$$

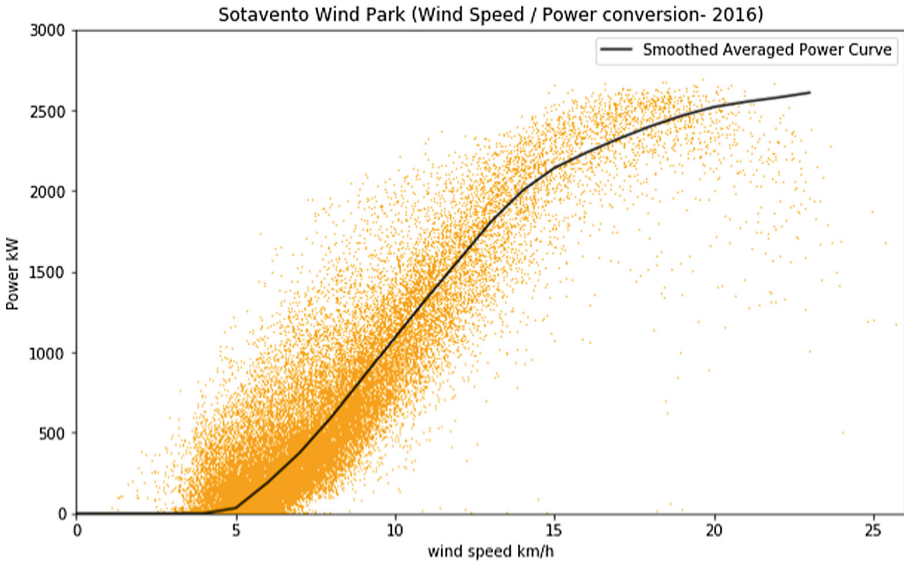


Fig. 3. Energy Generated in the Sotavento wind park (from observation data) [8] (Color figure online)

In Fig. 3 a graphical representation of the transformation of wind speed into power can be observed. The red-colored points are tuples of energy generated with wind speed (in 10 min periods). In this graph, the points concentrated along the original theoretical power curve (each turbine has its own). Additionally, it can be seen the existence of two relevant points: (a) the cut-in which defines the speed at which the turbine starts generating energy and, (b) the cut-off which

is the threshold where there is no additional power generated (it is quite usual that this point triggers safety mechanisms to avoid the blades to be damaged by strong winds). From all this points, it can be deducted that in order to forecast energy, it is mandatory to predict wind intensity, as it is the critical feature to convert wind to energy.

For the experiments the National Renewable Energy Laboratory (NREL) is used, which is the largest available dataset of wind points [3]. This large dataset offers production and meteorological data (wind speed, wind direction, temperature, humidity and energy) synthesized from Meteorological global models for over 120,000 sites evenly found in the US geography.

4 Methodology and Experiments

Two problems have been addressed in the experiments. First the multi-step prediction, and second the methodology to compute the prediction which can be defined as a regression problem.

According to the literature there are several approaches for a multi-step forecast, based on how the future time steps are obtained [10]. The first, and simplest, is the recursive approach, where a data window $[t_{i-k}, t_{i-1}]$ and prediction for time t_i are used for predicting t_{i+1} iteratively. This method is not very good for mid/long term prediction as there is a compounding error effect as the predicted data is reused. Another issue is the lack of the possibility to add exogenous variables in the prediction, which is a handicap for our approach.

The *direct* approach obtains predictions by computing a regression for each of the time points on the future horizon. This has the advantage of not reusing predicted data but is more computationally expensive given that multiple models have to be obtained. Another approach is to use multiple regression or sequence to sequence prediction methods [9]. This means that all the future time steps are obtained at the same time without reusing predictions and with a unique model. Both techniques are used in our experiments.

For solving the regression problem, we have chosen regression support vector machines (SVR) [1] and two neural network methods, multi-layer perceptron (MLP) and recurrent networks (RNN) [5]. The SVR will be used as a baseline and only for direct prediction. MLP and RNN will be used for direct prediction and multiple regression.

A systematic exploration of the parameters for all methods was performed. For SVR, different kernels will be used, namely Radial Basis Function (RBF), linear, quadratic and cubic polynomial with a wide range of values for the C parameter and the bandwidth for the RBF kernel.

For the MLP experiments, architectures from one to three layers of different sizes with a linear output were tried, with different activation functions (sigmoid and ReLU) and different values of dropout on each layer. The direct approach had one output neuron and the multiple regression as many outputs as the prediction horizon.

For the RNN architecture a structure from one to three layers of Long Short Term Memory units (LSTM) or Gated Recurrent (GRU) units with different

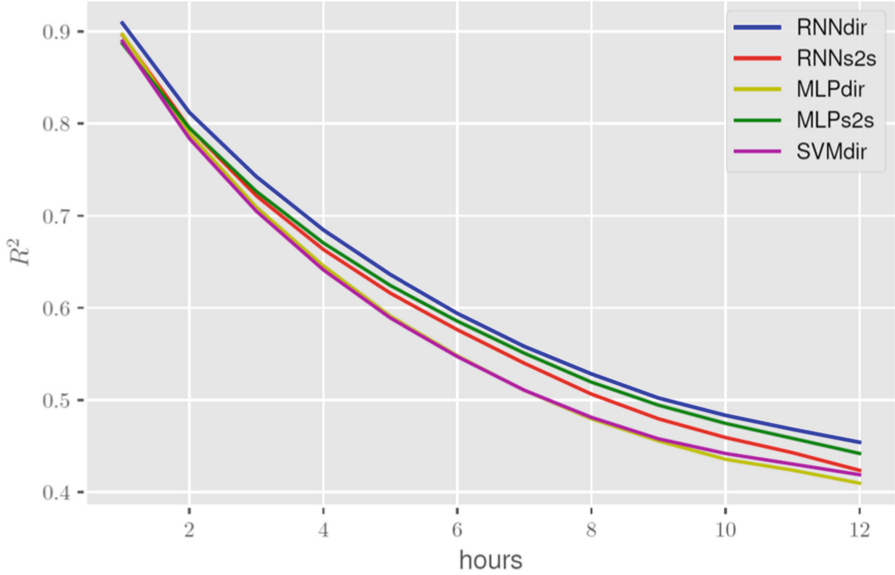


Fig. 4. Mean of R^2 for the 20 best experiments for each architecture, 12 h horizon.

sizes were used, with tanh, sigmoid and ReLU activation functions for the output of the recurrent units and different levels of recurrent dropout. For the direct approach, a MLP with linear output was used.

For the multiple regression, an encoder-decoder architecture was used [9], where the recurrent layers were used as a first stage, performing the encoder task. The state obtained from the encoder was used as input for another recurrent network, acting as a decoder, which generates a sequence with the length of the prediction horizon. Each time step of the decoder had direct access to the state from the encoder additionally to the state from the previous step.

5 Results

5.1 Main Results from Preliminary Experiments

The raw data in the dataset is characterized with five minutes sampling but in order to reduce computational cost and assuming a realistic forecasting scenario of hourly predictions, the data was reduced to hourly sampling by averaging the measures every hour. The data used for the prediction included the wind speed and direction at 100 m height plus barometric pressure and air density. The hour and month of the data were added as complementary variables in the time series.

The first four years of the time series was used for training, the fifth year was used as a test set for tuning the model parameters and the sixth year was used as the validation set. The training dataset has a size of 40912 values and the test and validation a size of 10228 values.

In order to analyze the algorithms behavior with different parameters and combinations, different windows lengths were used as input, ranging from three to 36 previous measures. The forecast was the wind speed from one to 12 h ahead.

To compare the accuracy of the different method results the determination coefficient measure (R^2) was chosen. The data was z-normalized, so the coefficient is equivalent to $1 - MSE$. Figure 4 shows the averaged R^2 for the best 20 results for each architecture for a specific site. The RNN model with direct prediction is consistently the best model followed by multiple regression with MLP. RNN with seq2seq performs similarly to the multiple regression MLP for the short term, but the mid-term predictions decay faster. The MLP and SVR with direct prediction have very similar results far from the rest.

Performing a two-sided Kolmogorov-Smirnov test for equality of distribution for the R^2 values for each time step of the prediction horizon for the best two architectures (RNN direct and MLP multi-regression) all distributions have less than $1e^{-4}$ as p-value, indicating that their distributions are different. The difference of the means for the different hourly prediction of the 20 best results is in the range 0.007 to 0.022.

The best RNN architectures have two layers of GRU units with ReLU activation functions, drop out of around 0.3 with a window input from 16 to 24 h. This result shows that the RNN architecture has a superior ability for the prediction task than the simpler MLP or SVR combinations. This shows that the RNN construct have some superior capabilities for this task.

5.2 Relationship with Site Complexity

An interesting geographical analysis of the errors obtained using different methods shows a measure of the site complexity, related to the errors obtained using different methods.

Representing graphically the error obtained by using persistence and multiple regression (seq2seq) MLP (see Figs. 5 and 6) it can be established a relationship

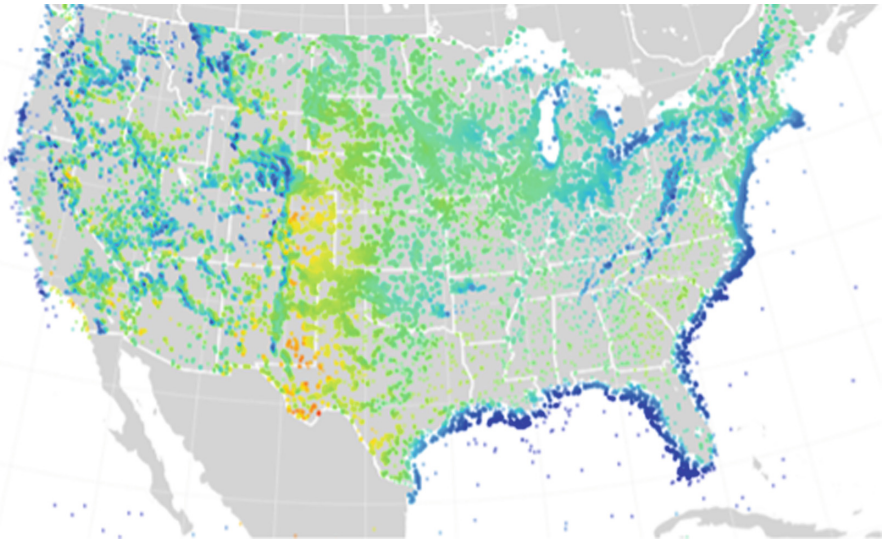


Fig. 5. Error R^2 in persistence over all NREL sites

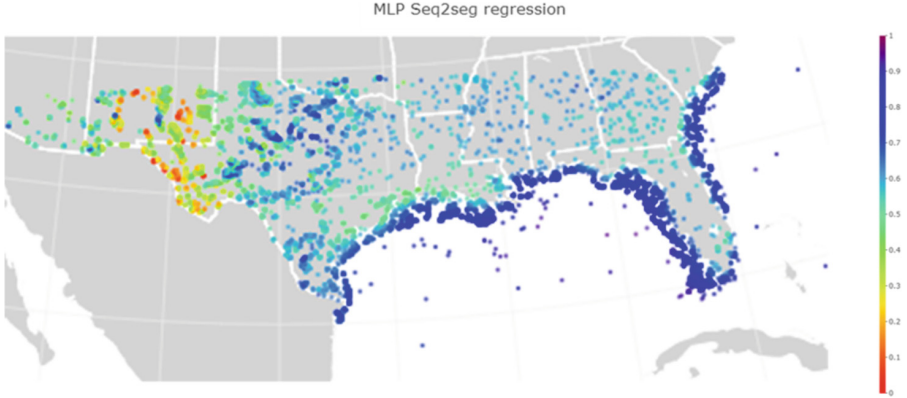


Fig. 6. Error R^2 in multiple regression MLP over NREL sites south Parallel 35°

between site complexity and prediction error. The most complex sites in the US are located in the western side of the central plains, where high regime winds have high variability and high difficulty for prediction.

6 Conclusions and Future Work

The preliminary results of our experiments show that RNN architectures with direct multi-step prediction can obtain reasonable mid-term predictions of wind speed with consistent accuracy among a limited number of sites. Other approaches show more significant decreases in accuracy the further the horizon of prediction.

The experiments also show that the direct approach for multi-step prediction have better results compared to a multiple regression/sequence to sequence approach.

Other multi-step prediction methods have to be explored, combined with different RNN architectures. For instance, given that it is more important the mid-term prediction, a multiple regression focused only on the more distant future seems more interesting. Also, further experiments using more advanced methods for the RNN sequence to sequence architectures have to be explored, like teacher forcing or attention mechanisms.

Finally as the available data has over 120,000 sites evenly distributed all across the US geography, a set of experiments to identify the best algorithm for each site topology is under way, with the objective to obtain the best algorithm structure that combines the characteristics of the time series with the site geographical characteristics of the site.

References

1. Bishop, C.M.: Pattern Recognition and ML. Springer, Berlin (2006)
2. Díaz-Dorado, E., Carrillo, C., Cidras, J., Albo, E.: Estimation of energy losses in a wind park. In: 2007 9th International Conference on Electrical Power Quality and Utilisation, pp. 1–6, October 2007. <https://doi.org/10.1109/EPQU.2007.4424125>
3. Draxl, C., Clifton, A., Hodge, B.M., McCaa, J.: The wind integration national dataset (WIND) Toolkit. Appl. Energy **151**, 355–366 (2015)
4. Gan, M., Li, H.X., Chen, C.L.P., Chen, L.: A potential method for determining nonlinearity in wind data. IEEE Power Energy Technol. Syst. J. **2**(2), 74–81 (2015). <https://doi.org/10.1109/JPETS.2015.2424700>
5. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
6. Landberg, L.: Meteorology for Wind Energy. Wiley, Hoboken (2016)
7. Lew, D., Milligan, M., Jordan, G., Piwko, R.: The value of wind power forecasting. NREL (2011). <http://www.nrel.gov/docs/fy11osti/50814.pdf>
8. López, J., Dorado, A.D., Alvarez, J., Carrillo, C., Cidras, J.: The sotavento experimental wind park. In: Global Windpower Conference Paris (2002)
9. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. CoRR (2014). <http://arxiv.org/abs/1409.3215>
10. Taieb, S.B., Atiya, A.F.: A bias and variance analysis for multistep-ahead time series forecasting. IEEE Trans. Neural Netw. Learn. Syst. **27**(1), 62–76 (2016)
11. Theiler, J., Eubank, S., Longtin, A., Galdrikian, B., Farmer, J.D.: Testing for nonlinearity in time series: the method of surrogate data. Phys. D: Nonlinear Phenom. **58**(1), 77–94 (1992). [https://doi.org/10.1016/0167-2789\(92\)90102-S](https://doi.org/10.1016/0167-2789(92)90102-S)
12. UNFCCC: Secretariat: report of the conference of the parties. Paris agreement 2015. United Nations Office at Geneva (2015)
13. Wind Europe organization: Wind in power 2016 European Statistics, Windeurope Organization, Belgium (2017)



Improved Architectural Redesign of MTree Clusterer in the Context of Image Segmentation

Marius Andrei Ciurez^(✉) and Marian Cristian Mihaescu

Faculty of Automation, Computers and Electronics, University of Craiova,
Str. Decebal nr. 107, Craiova, Romania
mariusandrei.ciurez@gmail.com, mihaescu@software.ucv.ro

Abstract. Image segmentation by clustering represents a classical use-case of unsupervised learning. A key aspect of this problem is that instances that are being clusters may have various types and thus requesting specific algorithms that implement particular distance functions and quality metrics. This paper presents an improved version of MTree clusterer that has been tested in the context of image segmentation in the same setup as a new recently k-MS algorithm. The redesigned MTree algorithms allows many levers for setup so that many configurations are available depending on the particularities of the tackled problem. The experimental results are promising especially as compared with the ones from previous MTree version and also as compared with classical clustering algorithms or newly developed k-MS algorithm. Further improvements in terms of available algorithms for configuration and algorithmic efficiency of integrations may lead the way to a general purpose clusterer that may be used for processing various data types.

Keywords: Clustering · MTree · Image segmentation

1 Introduction

Clustering algorithms have found many application domains where unsupervised learning provides efficient solutions to tackled problems. Among the most well known application domains there are medical image processing (i.e., pattern recognition and image segmentation) [1, 2], general and natural language processing knowledge discovery [3, 4, 11] navigation of robots [5, 6] and in many other contexts.

In the area of unsupervised learning there are several general classes of clustering algorithms (i.e., flat, hierarchical and density based) that all share two common problems: finding the optimal number of clusters and quickly and efficiently finding the correct clusters taking into consideration specific distance measures appropriate for the objects (i.e., pixels, points, persons, books, etc.) that are being grouped.

The objective of this work is to present an improved version of the MTree clustering algorithm [7] that is currently implemented as a Weka package [8,9]. The improved version has been tested in a comparative benchmark with k-MS morphological reconstruction clustering algorithm [10] as well as classical algorithms such as simple k-means, Cobweb, Farther First and Canopy.

The proposed approach tackles the practical problem of recognizing shapes as described in [10] by improving MTree clustering algorithm in terms of dataset preprocessing for finding optimal number of clusters and adjusting the business logic of the clusterer in terms of division policy and distance metric between instances. As compared with the initial results obtained by MTree clusterer reported in [10] we conclude that current version provides significantly better results than initial version and in several aspects challenges the clustering algorithms used in benchmarking process. The progress of MTree clusterer from the initial version consists in several improvements from algorithmic and implementation perspectives. The experimental results are validated by classical clustering quality metrics as in [10].

The paper is organized as follows. In Sect. 2, we perform a literature review with regards to finding optimal K (i.e., the number of clusters), clustering algorithms in Weka, division policies in clustering and validation by clustering quality metrics. Section 3 describes the proposed approach with a detailed presentation of each module from the clustering data analysis process with focus on algorithmic challenges. We also perform a complexity analysis of the newly obtained algorithm compared with the older one and with the other clustering algorithms used in the comparative analysis. In Sect. 4, we present experimental results that compare the quality and time performance of MTree implementation with other clustering algorithms. Finally, Sect. 5 contains the conclusions of this work, summarizes the key approaches of the improved version of the MTree algorithm and discusses potential improvements and applications.

2 Related Works

Data clustering is represented by classical area of unsupervised machine learning that come in many flavours and have found their way in image clustering or segmentation [12]. From this perspective, a wide range of variations we proposed in the literature.

Dhanachandra et al. in [12] use subtractive clustering along classical K-means algorithm in order to preprocess the data for optimal centroid initialization. The experimental results were obtained on medical images representing infected blood cells with malaria and on classical images used for segmentation obtaining better results than k-means taking into account RMSE and PSNR metrics.

A more elaborate approach for image clustering was proposed by Chang et al. in [13]. They propose a Deep Adaptive Clustering (DAC) approach that reduces to a classification problem in which similarity is determined by cosine distance and learned labeled features tend to be one-hot vectors obtaining good results on popular datasets like MNIST, CIFAR-10 and STL-10.

Retrieval of similar images from an image database (CBIR – Content Based Image Retrieval) represents a challenging task that has been addressed in [14]. The first approach uses as features color and texture and employs K-means and hierarchical clustering for finding the most similar images. The second approach uses color, texture and shape as features and K-means as business logic for building four different groups of images: dinosaurs, flowers, buses and elephants. The obtained experimental results are promising in terms of good precision and recall values.

A more complex context occurs when the image source is unknown or when the ground truth for the training dataset is also unknown [15,16]. In this situation, optimal K represents a critical issue as well as using an efficient distance function such that usage of a particular loss function provides good experimental results. These approaches propose as solution a workaround hierarchical clustering and clustering ensembles based graph partitioning methods, such as Cluster-based Similarity Partitioning Algorithm (CSPA), Hyper Graph Partitioning Algorithm (HGPA), and Meta Clustering Algorithm (MCLA).

Another critical aspect of unsupervised learning is represented by the optimal number of clusters that reside in the dataset. Unfortunately, scenarios in which the value of K is known occur in only a subset of practical scenarios. In general, image processing applications do not have a value of K that is known beforehand. This may occur when dealing with data streams [17] or with very high-dimensional datasets [18]. In general, the most suitable approach reduces to automatic determination of K that may be based on dynamic clustering [19] or joint tracking segmentation [20]. Finally, the whole clustering process needs validation, and this may be accomplished by many quality metrics for a wide range of algorithms [24]. Depending on the structure of the dataset various clustering quality frameworks [22,23] have been proposed. The key issue that always arises regards choosing the proper similarity and quality metrics [23].

3 Proposed Approach

The proposed approach follows a classical data analysis pipeline that is appropriate for unsupervised learning and is presented in Fig. 1. The input is represented by one image that is preprocessed in order to load the pixels and build an .arff file suitable for processing by Weka clustering algorithm implementations.

The clustering analysis benchmark uses several classical clustering implementations such as k-means, Cobweb, Farthest First and Canopy along our own improved version of MTree, the k-MS algorithm and other utility algorithms such as xMeans, voteK, Hierarchical Clustering, EM or Cascade k-Means. Running the clustering algorithms within the benchmark is managed by several key properties. One regards all the clustering algorithms and is represented by the number of clusters which are searched in the input image. The other settings are the division policy, seed selection mechanism and number of seeds and these apply only to MTree clusterer. All other clusterers use the same k as MTree along with other default settings. This approach makes possible further comparative analysis of various configurations of our MTree clusterer with clusterers

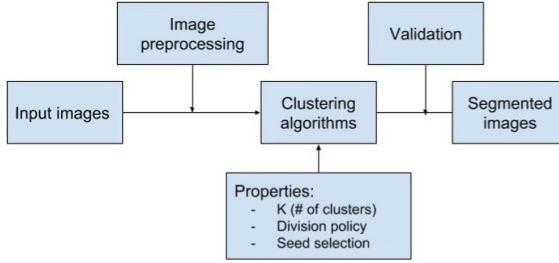


Fig. 1. Block diagram of the clustering analysis benchmark

that are already implemented in Weka and with k-MS algorithm reported in [10]. Another key aspects regard the number of seeds that are taken into consideration and the order in which data points are provided as input. These settings are provided as levers for MTree configurations and may influence the quality of the clustering results. As general rules, a minimum number of seeds needs to be taken into consideration as in many other clustering algorithms such that global optima is not missed due to a local optima. As for the order in which instances are provided to the MTree clusterer a random choosing approach represents a baseline scenario.

Finally, the clustering analysis benchmark returns a set of segmented images along with their corresponding validation metrics. For current approach we use two validation techniques: SSE and visual analytics. The key component of the clustering analysis benchmark is represented by the clustering algorithms module and especially by the settings that accompany the MTree cluster and represent the core improvements in terms its capabilities and efficiency.

Figure 2 presents the algorithmic infrastructure of the clustering analysis benchmark with emphasis on the list of clustering algorithms and main options in terms of possible settings for used algorithms in general and for MTree clusterer in particular.

Finally, we describe the key improvements of MTree implementation as compared with previous one presented in [7]. The first improvement regards the logic of split method that is performed for a full node. In this regard, there are two issues that were addressed: one regards the number of clusters and one regards the division policy. In the improved MTree the number of clusters may be set before running, but we may also leave this parameter to be determined at runtime by specifying a particular algorithm for determining the optimal k in the input dataset. According with the value of k (i.e., specified or not specified) the splitting procedure uses an appropriate division policy. Thus, if the value of k is known, than the division policy is performed by an algorithm which require a value for k as input (i.e., k-Means, Farthest First). On the contrary, if k is not known, the division polity is performed by an algorithm which does not need a value for k, such that x-Means, Cascade k-means, EM or Cobweb. A final improvement in MTree regards the seed selection, as a general issue in clustering

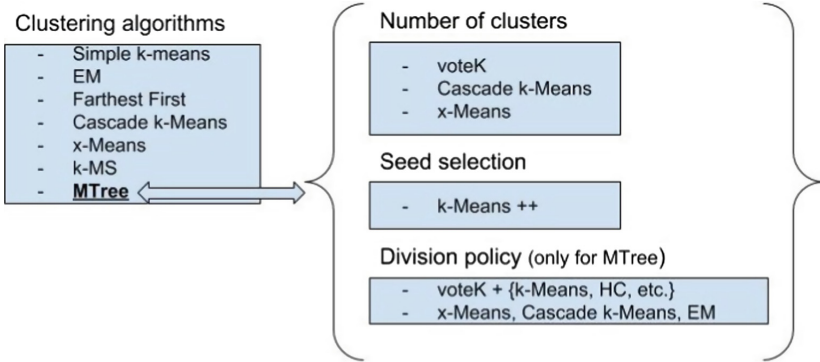


Fig. 2. Infrastructure of the clustering analysis benchmark

data. The current approach uses random seed selection and selection based on k-means++ algorithm. All algorithmic choices were made such that they are available in Weka and can be integrated in the business logic of the MTree and in the infrastructure of the clustering analysis benchmark.

4 Experimental Results

All the processing is performed on the image from [10] which reports good results for the proposed k-MS algorithm and poor results for MTree which justify current improvements.

The input image is preprocessed such that an .arff file with two features is obtained. As in [10] the features are represented by the numeric values representing the Cartesian coordinates of 9163 points. These input points are given as input to all configurations of MTree parametrized by various methods within split procedure. Figure 3 presents a comparative result of the five MTree configurations versus five classical algorithms. Performed experiments use MTree configurations that integrate the voteK algorithm for getting the optimal number of clusters along with Cobweb (MT_vK_CW), Farther First (MT_vK_FF), Canopy (MT_vK_C), Hierarchical Clustering (MT_vK_HC) and Cascade simple K-Means (MT_vK_cSKM) algorithms within the split procedure. All MTree configurations provide computed SSE values as well as classical simple k-Means. Therefore, all the obtained results from Fig. 3 have as optimality criteria the minimum value for SSE and for providing a sound comparative analysis the number of clusters was set to eight. The minimum SSE criteria and K equals to eight were chosen in order to have similar context with experimental results from [10]. The other algorithms do not provide values for SSE because of two reasons: either this functionality is not implemented in Weka (i.e., Farthest First, Canopy) or the algorithm itself - by its inner logic - is not suited for computing SSE values due to lack of notion of centroid (i.e., k-MS, Cobweb). This is the

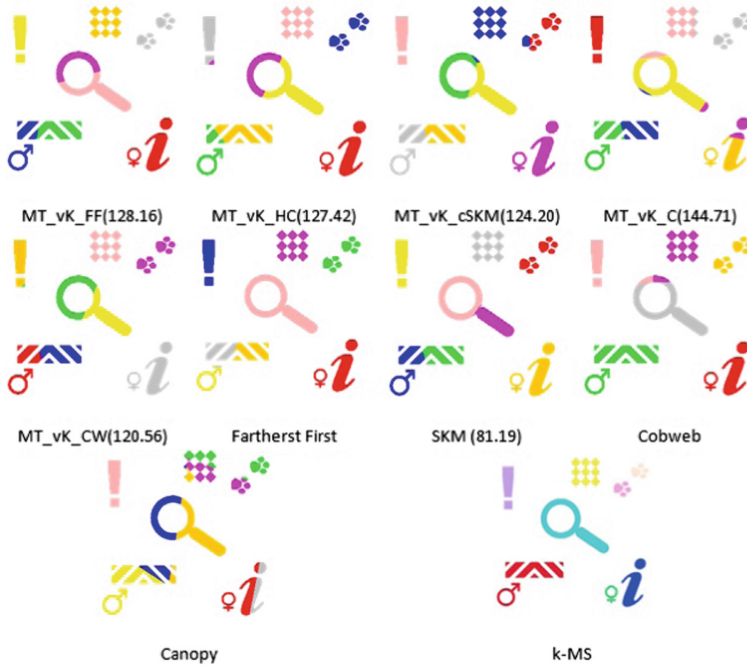


Fig. 3. Comparison between MTree results and other classical clustering algorithms

reason why visual analytics is implied as a second evaluation technique. Therefore, visual (i.e., manual) analytics was used to evaluate the suboptimal clustering distributions, that is distributions that have larger SSE value although provide a better clustering. Figure 4 presents the visual analytics results for the five MTree configurations that were used in the clustering analysis benchmark. Initial results show slight arguable improvements in two configurations (i.e., MT_vK_HC, MT_vK_C and MT_vK_cSKM) and better improvements in three configurations (i.e., MT_vK_FF, MT_vK_CW). Two key settings for the experiments regard the number of seeds and the order in which data points are provided as input. Presented results were obtained after runs on 100 seeds as this is the usual default value in such situations. The data points were streamed to the MTree clusterer in random order. As we are dealing with images, experiments show a degradation of accuracy that is poor clustering results in terms of SSE values and correctly segmented images when data points are given into a particular (i.e., row-wise or column-wise) order. The distributions obtained by all MTree configurations are much better than the result reported in [10]. Still, the MT_vK_FF and MT_vK_CW configurations that use the SSE metric are arguable better than classical algorithms. The other three configurations, MT_vK_HC, MT_vK_cSKM and MT_vK_C are much better than their corresponding classical algorithms but do not outperform k-MS. The advantage of

MTree algorithm resides in the speed by which it clusters new images once a clusterer has been trained.

5 Conclusion

Current study tackles the problem of image clustering. It provides an improved version of the MTree algorithm that is used for image segmentation in the same context as previously discussed in [10]. Improvements of MTree take into account the algorithmic approach that is based on the split method in which the number of clusters, the seed selection and the division policy are key ingredients which have been parametrized such that various configurations may be obtained. We performed experiments in various configurations and presented the ones that use the same setup as in [10] for a reproducible and comparative analysis. The improved MTree package along with voteK method for choosing optimal K are open-source and available in MTree Clusterer package [9].

Current results of all MTree configurations that were taken into consideration are highly improved as compared with initial one used in [10] and challenge classical clustering algorithms and k-MS. An advantage of the MTree clusterer is the feasibility for customization such that it may process other data types (i.e., educational data) compared with k-MS that may work only for images. Further improvements should take into account other clustering quality metrics and distances that may be better suited for this particular problem or for similar problems. Having access to SSE values for other clustering algorithms implemented in Weka and which have centroids and distances may provide a better objective comparative analysis. Observing that visual analytics may obtain slightly better distributions opens the way the need to take into consideration other relevant aspects that may automatically provide optimal solutions.

References

1. Silva, L.F., Santos, A.A.S., Bravo, R.S., Silva, A.C., Muchaluat-Saade, D.C., Conci, A.: Hybrid analysis for indicating patients with breast cancer using temperature time series. *Comput. Methods Programs Biomed.* **130**, 142–153 (2016)
2. Goswami, S., Bhaiya, L.K.P.: Brain tumour detection using unsupervised learning based neural network. In: 2013 International Conference on Communication Systems and Network Technologies (CSNT), pp. 573–577. IEEE, April 2013
3. Gan, G., Ma, C., Wu, J.: *Data Clustering: Theory, Algorithms, and Applications*, vol. 20. SIAM, Philadelphia (2007)
4. Miarro-Gimnez, J.A., Kreuzthaler, M., Schulz, S.: Knowledge extraction from MEDLINE by combining clustering with natural language processing. In: *AMIA Annual Symposium Proceedings*, vol. 2015, p. 915. American Medical Informatics Association (2015)
5. Di Caro, G.A., Ducatelle, F., Gambardella, L.: A fully distributed communication-based approach for spatial clustering in robotic swarms. In: *Proceedings of the 2nd Autonomous Robots and Multirobot Systems Workshop (ARMS)*, Affiliated with the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 5 June, Valencia, Spain, pp. 153–171, June 2012

6. Gauci, M., Chen, J., Li, W., Dodd, T.J., Gross, R.: Clustering objects with robots that do not compute. In: Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems, pp. 421–428. International Foundation for Autonomous Agents and Multiagent Systems, May 2014
7. Mihaescu, M.C., Burdescu, D.D.: Using M tree data structure as unsupervised classification method. *Informatica* **36**(2) (2012)
8. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. *ACM SIGKDD Explor. Newsl.* **11**(1), 10–18 (2009)
9. MTree Clusterer. <http://weka.sourceforge.net/packageMetaData/MTreeClusterer/index.html>
10. Rodrigues, O., Torok, L., Liatsis, P., Viterbo, J., Conci, A.: k-MS: a novel clustering algorithm based on morphological reconstruction. *Pattern Recognit.* **66**, 392–403 (2017)
11. Rebedea, T., Chiru, C.-G., Trăușan-Matu, Ș.: News Web Portal based on Natural Language Processing. <http://rochi.utcluj.ro/rrioc/en/rochi2008.html>
12. Dhanachandra, N., Manglem, K., Chanu, Y.J.: Image segmentation using K-means clustering algorithm and subtractive clustering algorithm. *Procedia Comput. Sci.* **54**(2015), 764–771 (2015)
13. Chang, J., Wang, L., Meng, G., Xiang, S., Pan, C.: Deep adaptive image clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5879–5887, October 2017
14. Maheshwari, M., Silakari, S., Motwani, M.: Image clustering using color and texture. In: 2009 First International Conference on Computational Intelligence, Communication Systems and Networks, CICSYN 2009, pp. 403–408. IEEE, July 2009
15. Caldelli, R., Amerini, I., Picchioni, F., Innocenti, M.: Fast image clustering of unknown source images. In: 2010 IEEE International Workshop on Information Forensics and Security (WIFS), pp. 1–5. IEEE, December 2010
16. Dash, A., Chatterjee, S., Prasad, T., Bhattacharyya, M.: Image clustering without ground truth. arXiv preprint [arXiv:1610.07758](https://arxiv.org/abs/1610.07758) (2016)
17. Guha, S., Mishra, N.: Clustering data streams. *Data Stream Management. DSA*, pp. 169–187. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-540-28608-0_8
18. Esmín, A.A., Coelho, R.A., Matwin, S.: A review on particle swarm optimization algorithm and its variants to clustering high-dimensional data. *Artif. Intell. Rev.* **44**(1), 23–45 (2015)
19. Ozturk, C., Hancer, E., Karaboga, D.: Dynamic clustering with improved binary artificial bee colony algorithm. *Appl. Soft Comput.* **28**, 69–80 (2015)
20. Milan, A., Leal-Taix, L., Schindler, K., Reid, I.: Joint tracking and segmentation of multiple targets. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5397–5406. IEEE, June 2015
21. Sleit, A., Qataweh, M., Al-Sharief, M., Al-Jabaly, R., Karajeh, O.: Image clustering using color, texture and shape features. *KSII Trans. Internet Inf. Syst.* **5**(1) (2011)
22. Castellanos, A., Cigarrn, J., Garca-Serrano, A.: Formal concept analysis for topic detection: a clustering quality experimental analysis. *Inf. Syst.* **66**, 24–42 (2017)
23. dos Santos, T.R., Zrate, L.E.: Categorical data clustering: what similarity measure to recommend? *Expert Syst. Appl.* **42**(3), 1247–1260 (2015)
24. Fahad, A.: A survey of clustering algorithms for big data: taxonomy and empirical analysis. *IEEE Trans. Emerg. Top. Comput.* **2**(3), 267–279 (2014)



Exploring Online Novelty Detection Using First Story Detection Models

Fei Wang^{1,2}(✉), Robert J. Ross^{1,2}, and John D. Kelleher^{1,2}

¹ School of Computing, Dublin Institute of Technology, Dublin, Ireland
d13122837@mydit.ie, {robert.ross,john.d.kelleher}@dit.ie

² ADAPT Research Centre, Dublin, Ireland

Abstract. Online novelty detection is an important technology in understanding and exploiting streaming data. One application of online novelty detection is First Story Detection (FSD) which attempts to find the very first story about a new topic, e.g. the first news report discussing the “Beast from the East” hitting Ireland. Although hundreds of FSD models have been developed, the vast majority of these only aim at improving the performance of the detection for some specific dataset, and very few focus on the insight of novelty itself. We believe that online novelty detection, framed as an unsupervised learning problem, always requires a clear definition of novelty. Indeed, we argue the definition of novelty is the key issue in designing a good detection model. Within the context of FSD, we first categorise online novelty detection models into three main categories, based on different definitions of novelty scores, and then compare the performances of these model categories in different features spaces. Our experimental results show that the challenge of FSD varies across novelty scores (and corresponding model categories); and, furthermore, that the detection of novelty in the very popular Word2Vec feature space is more difficult than in a normal frequency-based feature space because of a loss of word specificity.

Keywords: Online novelty detection · First Story Detection (FSD)
Unsupervised learning · Novelty score · Feature space · Word2Vec

1 Introduction

Novelty detection, also known as anomaly detection or outlier detection, is the task of identifying data that are different in some respect from other predominant chunks of data in a dataset [10]. Novelty is the property of abnormal data that usually indicates a defect (industry), a fraud (business), or a new topic in texts (media). In many cases, no definition is given for novelty before detection, and the abnormal data embedded in a large amount of normal data are not sufficient to build a class for novelty. Novelty detection is thus best conceptualized as an unsupervised machine learning application, i.e., no labels are available and detections can only be based on the intrinsic properties of the data.

In the context of streaming data, there are two additional constraints on the design of a novelty detection model that do not generally apply in the classical case [4]: (1) the model can only use the data that has arrived before detection is performed, and (2) the detection must be made within a bounded (short) time. If these two requirements are satisfied, this type of novelty detection is called online novelty detection. When online novelty detection is applied in the context of First Story Detection (FSD), the detection is targeted at finding the very first story about each new topic, e.g. the first news report discussing the “Beast from the East” hitting Ireland. Hundreds of online novelty detection models have been applied to FSD, however, the vast majority of research aims only at improving the performance of the detection for some specific dataset, and very few of them focus on the insight of novelty itself. A general categorisation of novelty detection models was previously proposed by Pimentel [10] which includes: probabilistic-based, distance-based, reconstruction-based, domain-based, and information-theoretic models. However, this categorisation is established solely on the techniques used in the detection, and therefore does not naturally provide comparisons across the categories, nor insights into how the different categories of models define the concept of novelty.

We believe that online novelty detection, as a typical unsupervised learning application, always requires a transparent definition of novelty. Moreover, the clear exposition of this definition is the key issue in designing a good detection model and enhancing its performance. Within the context of FSD, in this paper we identify three main categories of online novelty detection models as motivated by previous research. We will see that each of these three categories is based on different definitions of novelty scores. Across these categories, we present an analysis of how each category treats the concept of novelty, and furthermore, we compare the performances of these model categories in different feature spaces. We proceed by providing further detail on the problem of First Story Detection.

2 Background

The challenge FSD was initially defined as a task within the Topic Detection and Tracking (TDT) competition series. In TDT, the definition of FSD is given specifically as “the task of identifying new events in a stream of stories. Each story is processed in sequence, and a decision is made whether or not a new event is discussed in the story, after processing the story but before processing any subsequent stories” [1]. In this context, a “topic” is “a seminal event or activity, along with all directly related events and activities” [3].

In TDT, a number of online novelty detection models were developed for FSD. Of these, the UMass model [2] and CMU model [15] achieved the best performance. The UMass model was based on a nearest neighbour approach and the CMU model employed two types of clustering models. Petrovic et al. [8] proposed an extension to the UMass nearest neighbor model that integrates Locality Sensitive Hashing (LSH) when seeking the nearest neighbour. The primary contribution of this extension was to improve the computational efficiency

of the model. However, Petrovic et al.’s work is also noteworthy because it was the first FSD work applied to Twitter data, and since then Twitter has become a popular application domain for FSD research. In 2012 Petrovic [9] published the first Twitter dataset for FSD, the Edinburgh Twitter corpus. In recent years, more and more FSD models have been proposed, such as k-term hashing [14] in which the new data is compared to a look-up table to generate the novelty score, and the Nuggest-based method [11] in which only the top-k events need to be preserved as the comparison targets.

3 Three Categories of FSD Models

As claimed in our previous research [13], we believe that unsupervised learning always requires an appropriate definition of the learning target. Similarly, novelty detection always requires a definition of novelty. However, a reviewing of the literature on FSD research makes it apparent that every FSD model has its own way to calculate a novelty score, which can be considered as the definition of novelty used by that model. In other words, every FSD model aims to find the novelty represented by a specific novelty score.

In order to frame definitions of novelty in FSD, we propose (based on our analysis of FSD literature) three categories of novelty scores, and, three corresponding categories of FSD models, these are: Point-to-Point (P2P) models, Point-to-Cluster (P2C) models, and Point-to-All (P2A) models. This categorisation is based on different distances used to define novelty scores in different models. The concept of distance we are using here is a general expression that refers to the difference or dissimilarity between two objects, which can be two points, a point and a cluster, or a point and all other data. The three categories of models are detailed as follows:

Point-to-Point (P2P) models, in which the novelty score is defined as the distance from the new data to an existing data point:

$$Novelty_Score_{P2P} \stackrel{\text{def}}{=} distance(data_{new}, data_{existing}) \quad (1)$$

The nearest neighbour based model is a typical P2P FSD model, in which the novelty score is defined as the distance from the new data to the closest existing data point to it. In order to improve efficiency, P2P models usually accept an approximate nearest neighbour to each incoming data point, instead of the true nearest neighbour. For example, the UMass model [2] only seeks the nearest neighbour from the data points that have at least a single word in common with the new data.

Point-to-Cluster (P2C) models, in which the novelty score is defined as the distance from the new data to a cluster of existing data:

$$Novelty_Score_{P2C} \stackrel{\text{def}}{=} distance(data_{new}, cluster_{existing}) \quad (2)$$

The distance in defining the novelty score in P2C models is between a data point and a sub-space (or the union of sub-spaces) formed by a cluster of existing data points in the feature space. In the calculation, it could be the distance to a representative of the space, the distance to the range of the sub-space, or even the distance to a model trained by the cluster of data points in the sub-space. In the context of FSD, a cluster can be intuitively understood as a topic behind the texts. To make it simple, the cluster is usually represented by some data point within its range, e.g. the centroid of the cluster, the furthest point or the closest point to the new data point. The CMU model [15] that is based on single-pass clustering is a typical P2C model based on the distance from the new data to the centroid of the closest cluster.

Point-to-All (P2A) models, in which the novelty score is defined as the distance from the new data to all the existing data:

$$Novelty_Score_{P2A} \stackrel{\text{def}}{=} distance(data_{new}, all_data_{existing}) \quad (3)$$

Given all existing data, the detection of novelty can be considered as a One-Class Classification (OCC) problem, in which the quantity of existing normal data is large enough to build the “normality”, but the quantity of abnormal data is insufficient to build the novelty class for classification. One-Class SVM [12] is a popular model for OCC, the basic idea of which is to generate a hyper-sphere based on all existing data, and all the data points outside the hyper-sphere are considered as novel data.

It is worth highlighting that any novelty detection model that is based on a model trained on all the existing data can be viewed as a P2A model. For example, the k-term hashing model [14] compares all the terms of the new data with a look-up table created using all the existing data, and takes the proportion of new terms as the novelty score, and so it is a P2A model. The same goes for the probabilistic-based and reconstruction-based models. On the other hand, if the novelty score is defined by calculating the distance to subsets of data or using models built by subsets of data, then the model is a P2C model. For example, we can use a different distance from the incoming data point to each existing cluster in a P2C model by calculating the distance to the hyper-sphere around all the data of each cluster. Even though this model uses the same technique as a P2A model, this is still a P2C model.

Based on the descriptions given above, we can see that the differences between these three categories of models are actually dependent on the target object from which the distance of new data is defined, rather than what domain theories and/or model architectures are used. Using these three categories of models, we can analyze not only the performance of a single detection model, but also the common characteristics of models within a category, and furthermore, do cross category comparisons based on these general characteristics. As a practical example of this we can compare the performances of novelty detection in different feature spaces, and obtain deeper insights into both novelty and the appropriateness of feature spaces.

4 Experimental Design

We use the above categorization of FSD models to design our experiments. Our first experiment, compares the performance of three representative instances of the model categories. The second experiment is designed to examine the performance of different categories of models across different feature spaces.

4.1 Dataset

In our experiment we use a standard FSD benchmark dataset known as TDT5. The dataset is composed of news reports from a number of news agencies during the period from April to September of the year 2003. Compared with the Twitter FSD dataset, the contents of TDT5 dataset are all in plain text and standard English, and contain very few special expressions. Our research is to explore novelty detection across different types of models and feature spaces, rather than improve the performance of a specific model on a particular domain, so we believe the TDT5 is the better option for this work, as it contains more standard English, and hence model performance is not as affected by a model’s ability to use features particular to a specific domain. For our experiments, we used the first 20,000 stories in the TDT5 dataset. Within this TDT5 subset there are 18 labeled topics (and hence 18 first stories), with a further 256 stories labeled as belonging to one of these 18 topics. The remaining stories are unlabeled.

4.2 FSD Performance Across Different Categories of Models

In the first experiment, we select a typical model for each category of model classes outlined in Sect. 3, and apply them to FSD on the TDT5 subset. For this experiment, all the text data was prepared by stemming each word into its root and removing stopwords and the words with frequency less than 3. After that, all the data was mapped into the tf-idf feature space, so the detections and analysis for this experiment are carried out within that feature space.

For P2P, we use a nearest neighbor model as the representative model. The incoming story is compared to all the existing stories to find the nearest neighbour. If the distance to the nearest neighbour exceeds a threshold, the story is declared novel. In order to improve efficiency, our implementation adopts some ideas from previous successful models - the UMass [2] model and the LSH [8] model that are different from a vanilla nearest neighbour model. Firstly, cosine distance is used to calculate the distance between two data points. Secondly, the model only seeks the nearest neighbour from the data points that have at least a single word in common with the new data. This is done by initially establishing an inverted index, and then continually updating the index as the detection process is ongoing. Thirdly, in order to improve computational efficiency only the 2,000 most recent data points obtained from the inverted index are considered when searching for the nearest neighbor.

The P2C category is represented by a single-pass (or called follow-the-leader) clustering model. In this model, we group the incoming stories into clusters.

As discussed in the introduction of P2C, the clusters represent topics behind the texts, and each cluster can be represented by its centroid, which is calculated as the mean of the vector representations of the stories in that cluster. Similarly, the incoming story is compared to the centroids of all the clusters to find the nearest cluster. If the distance to the nearest cluster does not exceed a threshold, the story is declared non-novel and assigned to the cluster after which the cluster centroid is updated; otherwise, we declare the story novel, and create a new cluster with this new story as the only data point within the cluster. Cosine distance is adopted in calculating the distance from a new data to a centroid.

For P2A, a One-Class SVM is adopted as the representative model because using this model it is easy to interpret the distance from a data point to all the existing data points. Given a parameter V between 0 and 1, all data are mapped into a hyper-space using a kernel function to generate a sphere that contains $1-V$ of the data inside it as normal data and V of the data outside it as novel data. In our model, we do not take the label by One-Class SVM as the label of novelty for a new data, but take the distance of the data to the sphere as the novelty score, with a positive value if the data is outside the sphere and negative value if the data inside it. Based on the results of validation tests, we select 0.1 as the value of V and use only the most frequent 300 features as the representations of stories in this part of experiment. Finally, to reduce the computational cost associated with repeatedly rebuilding a One-Class SVM, for each new query we also only use the 2000 most recent data points to build the model.

4.3 FSD Performance in Different Feature Spaces

The first experiment was conducted using tf-idf based representations of documents. For this second experiment, for all model categories, we also implemented FSD in the average Word2Vec feature space. Word2Vec [6] is a neural network model that generates vector representations of words (known as, word embeddings). It is trained with a large corpus of texts, and produces a dense vector space, typically of several hundred dimensions, in which each word is represented by a vector. Within this vector space, words with similar contexts are located in close proximity to one another, which is the most important property of Word2Vec. However, todate Word2Vec representations have not been directly applied to FSD; although, Word2Vec has been used indirectly to generate paraphrases in order to alleviate the problem caused by lexical variation [7]. In this experiment, we generate a vector representation for a story by averaging vector representations of the words in the story. We then compare the performance of the different FSD models using this word embedding based vector representations with the tf-idf representations used in the first experiment.

For this second experiment, we use the same data preparation approach that was used for the first experiment. We are aware that there are pre-trained Word2Vec embeddings available; however, for this experiment we train our own Word2Vec embeddings using all the TDT5 data so that the word embedding vectors reflect the most relevant word meanings for the TDT5 context. The number

of dimensions of the Word2Vec vectors is set as 300 and the slicing window used in the training is set as 10.

4.4 Evaluation

In both experiments we evaluate FSD performance using the Detection Error Tradeoff (DET) curve [5], which is also the standard evaluation method for FSD since the TDT series. For each FSD model, we run the evaluation with a large range of thresholds. Each threshold generates a pair of False Alarms and Misses error scores, all of which are mapped to the DET curve to show the tradeoff between these two types of errors as the model threshold is varied.

5 Experimental Results

5.1 FSD Performances Across Different Categories of Models

Figure 1 presents the DET curves generated during in experiment 1, by the representatives of the different categories of models. Inspecting the DET curves there is a general trend that performance becomes worse from P2P to P2C and P2A.

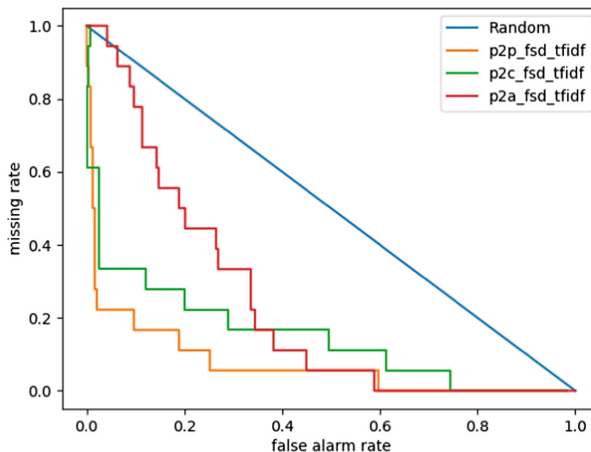
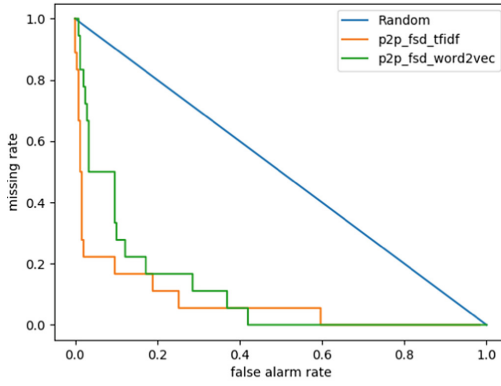


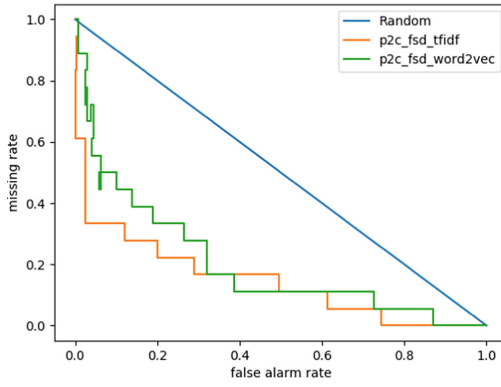
Fig. 1. FSD performances across different categories of models

5.2 FSD Performances in Different Feature Spaces

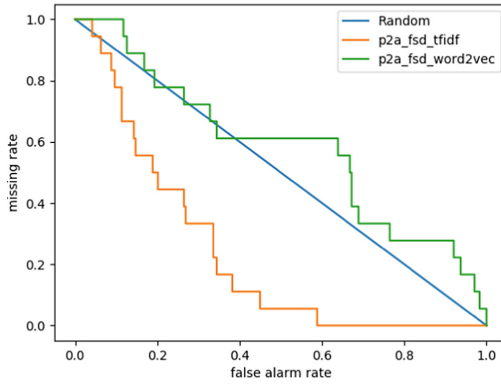
Figure 2 presents three DET graphs, one for each category of model. Within each graph two DET curves are plotted one for each feature space: tf-idf and Word2Vec. The most important finding is that, for all three categories of models, using Word2Vec representations result in worse performance on FSD, compared with tf-idf. Furthermore, the performance of Word2Vec representations decreases as we move from P2P to P2C and onto P2A. Indeed, the performance of the P2A model with Word2Vec features is worse than random selection.



(a) P2P



(b) P2C



(c) P2A

Fig. 2. Performances of P2P, P2C, and P2A models in different features spaces.

5.3 Result Analysis

Based on the experimental results, general trends in performance across different categories of models emerge, and it is possible to see clear differences in the performance of feature spaces across different categories of models. FSD performance drops as we move from P2P to P2C and P2A, and this trend is evident in the results from both experiments. Furthermore, using Word2Vec representations results in worse performance on FSD, and, again, performance drops as we move from P2P to P2C and onto P2A.

In terms of the explaining why the performances become worse from P2P to P2C and P2A, or from tf-idf to Word2Vec, one potential reason could be that the word specificity is diluted in a large number of documents or in the Word2Vec feature space, which is an important loss of information for novelty detection. The experimental results provide support for our hypothesis. For example, for the topic “Sweden rejected the euro”, the P2P model in the tf-idf feature space finds the first story easily, but the P2P model in the Word2Vec feature space usually fails because the first story is considered to be very similar to a previous document that discusses another topic “Portugal and the euro”. “Sweden” and “Portugal” are two different words in the frequency-based feature spaces like tf-idf, so the events in Sweden and Portugal can be clearly distinguished from each other within that representation. However, in the Word2Vec feature space, the words with common contexts are located in close proximity to one another, that is, the two words make little difference, so it is difficult to find the novelty caused by the word specificity in the Word2Vec feature space. Similarly, the loss of word specificity happens in P2C and P2A.

6 Conclusions

In this paper, we explored novelty detection by firstly identifying three categories of models, Point-to-Point (P2P), Point-to-Cluster (P2C) and Point-to-All (P2A), based on different definitions of novelty scores. We believe this categorisation leads to a good understanding of novelty and other topics across general categories of detection models. We further put this idea into practice and explored FSD based on different categories of models and different feature spaces. Our experimental results show that the challenge of FSD varies across different novelty scores and corresponding model categories; and, furthermore, that the detection of novelty in the very popular Word2Vec feature space is more difficult than in a normal frequency-based feature space because of a loss of word specificity.

The next phase of our research will be developed across two strands: the application of our categorisation to other contexts to enhance the understanding of the novelty concept; and the examination of the utility of other document embeddings like Doc2Vec to build qualitative evaluation of different feature spaces.





Acknowledgement. The authors wish to acknowledge the support of the ADAPT Research Centre. The ADAPT Centre for Digital Content Technology is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Funds.

References

1. Allan, J., et al.: Topic detection and tracking pilot study final report (1998)
2. Allan, J., et al.: Detections, bounds, and timelines: UMass and TDT-3. In: Proceedings of Topic Detection and Tracking Workshop. sn (2000)
3. Fiscus, J., et al.: NISTs 1998 topic detection and tracking evaluation (TDT2). In: Proceedings of the 1999 DARPA Broadcast News Workshop (1999)
4. Ma, J., Perkins, S.: Online novelty detection on temporal sequences. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM (2003)
5. Martin, A., et al.: The DET curve in assessment of detection task performance. National Institute of Standards and Technology, Gaithersburg, MD (1997)
6. Mikolov, T., et al.: Efficient estimation of word representations in vector space. arXiv preprint [arXiv:1301.3781](https://arxiv.org/abs/1301.3781) (2013)
7. Moran, S., et al.: Enhancing first story detection using word embeddings. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2016)
8. Petrovic, S., Osborne, M., Lavrenko, V.: Streaming first story detection with application to Twitter. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics (2010)
9. Petrovic, S., Osborne, M., Lavrenko, V.: Using paraphrases for improving first story detection in news and Twitter. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics (2012)
10. Pimentel, M.A.F.: A review of novelty detection. *Signal Process.* **99**, 215–249 (2014)
11. Qiu, Y., et al.: Time-aware first story detection in Twitter stream. In: IEEE International Conference on Data Science in Cyberspace (DSC). IEEE (2016)
12. Schlkopf, B.: Estimating the support of a high-dimensional distribution. *Neural Comput.* **13**(7), 1443–1471 (2001)
13. Wang, F., Franco-Penya, H.-H., Kelleher, J.D., Pugh, J., Ross, R.: An analysis of the application of simplified silhouette to the evaluation of k -means clustering validity. In: Perner, P. (ed.) *MLDM 2017*. LNCS (LNAI), vol. 10358, pp. 291–305. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-62416-7_21
14. Wurzer, D., Lavrenko, V., Osborne, M.: Twitter-scale new event detection via K-term hashing. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (2015)
15. Yang, Y., Pierce, T., Carbonell, J.: A study of retrospective and on-line event detection. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (1998)



A Fast Metropolis-Hastings Method for Generating Random Correlation Matrices

Irene Córdoba¹, Gherardo Varando^{1,2}, Concha Bielza¹,
and Pedro Larrañaga¹

¹ Department of Artificial Intelligence, Universidad Politécnica de Madrid, Madrid, Spain

irene.cordoba@upm.es

² Department of Mathematical Sciences, University of Copenhagen, Copenhagen, Denmark

Abstract. We propose a novel Metropolis-Hastings algorithm to sample uniformly from the space of correlation matrices. Existing methods in the literature are based on elaborated representations of a correlation matrix, or on complex parametrizations of it. By contrast, our method is intuitive and simple, based the classical Cholesky factorization of a positive definite matrix and Markov chain Monte Carlo theory. We perform a detailed convergence analysis of the resulting Markov chain, and show how it benefits from fast convergence, both theoretically and empirically. Furthermore, in numerical experiments our algorithm is shown to be significantly faster than the current alternative approaches, thanks to its simple yet principled approach.

Keywords: Correlation matrices · Random sampling
Metropolis-Hastings

1 Introduction

Correlation matrices are a fundamental tool in statistics and for the analysis of multivariate data. In many application domains, such as signal processing or regression analysis, there is a natural need for tools that generate synthetic, benchmark correlation matrices [1–3]. Existing algorithms for this task usually randomly sample either the eigenvalues of the matrix or the elements of its Cholesky decomposition, instead of directly sampling the matrix from its uniform distribution.

Uniform sampling of correlation matrices has received little attention until recently [4, 5]. By contrast with the classical methods [1, 2], uniform sampling does not assume any a priori information and allows to obtain an unbiased random correlation matrix. In this paper, we propose a new Metropolis-Hastings algorithm for such task, which is significantly faster than the existing algorithms

in the literature [4,5]. We perform a detailed analysis of the convergence properties of the Markov chain that we construct.

Our approach is similar to that of Pourahmadi and Wang [4] in the sense that we rely on the Cholesky factorization of a positive definite matrix; however they reparametrize the triangular factor with spherical coordinates, resulting in an additional layer of complexity. Lewandowsky et al. [5] present two methods based on alternative representations of the correlation matrix, vines and elliptical distributions, which are arguably less direct than our classical Cholesky factorization.

The rest of the paper is organized as follows. In Sect. 2 we briefly overview the Cholesky factorization of correlation matrices, and other technical results needed for our method. Section 3 contains the details of our Metropolis-Hastings algorithm, whose convergence properties are analyzed in Sect. 4, both from a theoretical and experimental point of view. In Sect. 5 we empirically compare the computational performance of our method with the alternatives in the literature. Finally, we conclude the paper in Sect. 6.

2 Upper Cholesky Factorization of a Correlation Matrix

Let \mathbf{R} be $p \times p$ a correlation matrix, that is, a symmetric positive definite (SPD) matrix with ones on the diagonal. Since \mathbf{R} is SPD, it has a unique upper Cholesky factorization $\mathbf{R} = \mathbf{U}\mathbf{U}^t$, with \mathbf{U} an upper triangular matrix with positive diagonal entries. Let \mathcal{U} denote the set of upper triangular $p \times p$ matrices with positive diagonal entries. We will define the set of SPD correlation matrices as

$$\mathcal{R} = \{\mathbf{R} = \mathbf{U}\mathbf{U}^t \text{ s.t. } \text{diag}(\mathbf{R}) = \mathbf{1}, \mathbf{U} \in \mathcal{U}\}. \quad (1)$$

The set \mathcal{R} of SPD correlation matrices is known to form a convex body called *elliptope* [6], whose volume has been explicitly computed by Lewandowski et al. [5]. Observe that the constraint $\text{diag}(\mathbf{R}) = \mathbf{1}$ in Eq. (1) simply translates to the rows of \mathbf{U} being normalized vectors. Denoting the subset of \mathcal{U} with such normalized rows as \mathcal{U}_1 , \mathcal{R} can be written more compactly as

$$\mathcal{R} = \{\mathbf{R} = \mathbf{U}\mathbf{U}^t \text{ s.t. } \mathbf{U} \in \mathcal{U}_1\}.$$

Consider now $\Phi(\mathbf{U}) = \mathbf{U}\mathbf{U}^t$ as a parametrization of \mathcal{U}_1 into SPD matrices. In order to sample uniformly from \mathcal{R} , which is the image of Φ , we need to compute the Jacobian matrix $J\Phi(\mathbf{U})$ [7]. Then, when sampling in \mathcal{U}_1 from a density proportional to the Jacobian $\det(J\Phi(\mathbf{U}))$, the induced distribution on \mathcal{R} by Φ is the uniform measure. In our case, the Jacobian is [8]

$$\det(J\Phi(\mathbf{U})) = 2^p \prod_{i=1}^{p-1} u_{ii}^i, \quad (2)$$

where u_{ii} is the i -th diagonal element of $\mathbf{U} \in \mathcal{U}_1$ and we have omitted u_{pp} because it is equal to 1.

3 Metropolis-Hastings Uniform Sampling

We will use a Metropolis-Hastings method for sampling from a density proportional to the Jacobian in Eq. (2). Observe that the i -th row in \mathbf{U} , denoted as \mathbf{u}_i in the remainder, can be sampled independently from all the other rows $\{\mathbf{u}_j\}_{i \neq j}$, from a density $f(\mathbf{u}_i) \propto u_{ii}^i$. Furthermore, \mathbf{u}_i is a unitary vector and has its first $i-1$ entries equal to zero, therefore it lives in the $(p-i)$ -dimensional hemisphere,

$$\mathcal{S}_+^{p-i} = \{\mathbf{v} \in \mathbb{R}^{p-i+1} \text{ s.t. } \mathbf{v}\mathbf{v}^t = 1 \text{ and } v_1 > 0\},$$

where the positivity constraint is to ensure that $u_{ii} > 0$. This independent row-wise sampling procedure is described in Algorithm 1.

Algorithm 1. Uniform sampling in \mathcal{R}

Input: Sample size N

Output: Uniform sample from \mathcal{R} of size N

```

1: for  $n = 1, \dots, N$  do
2:    $\mathbf{U}^n \leftarrow \mathbf{0}_p$ 
3:   for  $i = 1, \dots, p$  do
4:      $\mathbf{u}_i^n \leftarrow$  sample from  $f(\mathbf{u}_i) \propto u_{ii}^i$  on  $\mathcal{S}_+^{p-i}$ 
5:   end for
6: end for
7: return  $\{\Phi(\mathbf{U}^1), \dots, \Phi(\mathbf{U}^N)\}$ 

```

Since each row of \mathbf{U} can be sampled independently, in the remainder of this section we will concentrate on how to perform step 4 in Algorithm 1. In order to lighten the notation, we will restate our problem as sampling vectors \mathbf{v} from the hemisphere \mathcal{S}_+^{p-i} with respect to the density $f(\mathbf{v}) \propto v_1^i$, where p is fixed and $1 \leq i < p$.

In the Metropolis-Hastings algorithm, we need to generate a proposed vector $\tilde{\mathbf{v}}$ from the current vector \mathbf{v} already sampled from \mathcal{S}_+^{p-i} . For this, we will propose the new state as a normalized perturbation of the current vector, specifically,

$$\tilde{\mathbf{v}} = \frac{\mathbf{v} + \boldsymbol{\epsilon}}{\|\mathbf{v} + \boldsymbol{\epsilon}\|}, \quad (3)$$

where $\boldsymbol{\epsilon}$ is a Gaussian random vector of dimension $p-i+1$ with zero mean and component-independent variance σ_ϵ^2 .

With the transformation of Eq. (3) the induced proposal distribution $q(\tilde{\mathbf{v}}|\mathbf{v})$ is a projected Gaussian over \mathcal{S}_+^{p-i} [9], with parameters \mathbf{v} and $\sigma_\epsilon^2 \mathbf{I}_{p-i+1}$. The expression for the density of this angular distribution is given in the general case by [10]. In our setting, we obtain a simplified expression,

$$q(\tilde{\mathbf{v}}|\mathbf{v}) = \frac{\exp\left(\frac{((\mathbf{v}^t \tilde{\mathbf{v}})^2 - 1)/2\sigma_\epsilon^2}{(2\pi)^{(p-i+1)/2}}\right)}{\int_0^\infty s^{p-i} \exp\left(-\frac{1}{2}\left(s - \frac{\mathbf{v}^t \tilde{\mathbf{v}}}{\sigma_\epsilon}\right)^2\right) ds}. \quad (4)$$

The density for the proposal $q(\tilde{\mathbf{v}}|\mathbf{v})$ in Eq. (4) is a function of the scalar product $\mathbf{v}^t\tilde{\mathbf{v}}$, therefore it is symmetric because the roles of \mathbf{v} and $\tilde{\mathbf{v}}$ can be exchanged, and we can omit the Hastings correction from the sampling scheme. Thus the acceptance probability at each step of the algorithm becomes

$$\min\left(1, \frac{f(\tilde{\mathbf{v}})}{f(\mathbf{v})}\right) = \min\left(1, \mathbb{I}_{\geq 0}(\tilde{v}_1) \left(\frac{\tilde{v}_1}{v_1}\right)^i\right),$$

where \tilde{v}_1 is the first component of the proposed vector $\tilde{\mathbf{v}}$ and $\mathbb{I}_{\geq 0}$ denotes the indicator function of the positive real numbers. The described Metropolis sampling is illustrated in Algorithm 2.

Algorithm 2. Metropolis sampling of vectors \mathbf{v} in \mathcal{S}_+^{p-i} from $f(\mathbf{v}) \propto v_1^i$

Input: Sample size N , variance σ_ϵ^2 and burn-in time t_b

Output: Sample of size N from \mathcal{S}_+^{p-i}

```

1:  $\mathbf{v}_0 \leftarrow$  random standard multivariate Gaussian observation of dimension  $p - i + 1$ 
2:  $v_{01} \leftarrow |v_{01}|$ 
3:  $\mathbf{v}_0 \leftarrow$  normalize  $\mathbf{v}_0$ 
4: for  $t = 0, \dots, t_b + N$  do
5:   for  $j = 1, \dots, p - i + 1$  do
6:      $\epsilon_j \leftarrow$  random Gaussian observation with zero mean and variance  $\sigma_\epsilon^2$ 
7:   end for
8:    $\tilde{\mathbf{v}} \leftarrow \mathbf{v}_t + \epsilon$ 
9:    $\tilde{\mathbf{v}} \leftarrow$  normalize  $\tilde{\mathbf{v}}$ 
10:   $\delta \leftarrow$  random uniform observation on  $[0, 1]$ 
11:  if  $\tilde{v}_1 \geq 0$  and  $\delta \leq (\tilde{v}_1/v_{t1})^i$  then
12:     $\mathbf{v}_t \leftarrow \tilde{\mathbf{v}}$ 
13:  end if
14: end for
15: return  $\{\mathbf{v}_{t_b+1}, \mathbf{v}_{t_b+2}, \dots, \mathbf{v}_{t_b+N}\}$ 

```

4 Convergence Assessment

In this section we will analyze, both theoretically and empirically, the convergence properties of the proposed Algorithm 2, following [11].

4.1 Theoretical Convergence Properties

A minimal requirement for a Metropolis chain with proposal q to have the target density f as its stationary distribution is the following relationship between the supports

$$\text{supp}(f) \subseteq \bigcup_{\mathbf{v} \in \text{supp}(f)} \text{supp}(q(\cdot | \mathbf{v})).$$

Since in our case the support of $q(\cdot | \mathbf{v})$ is the $(p - i)$ -dimensional unit sphere, for all $\mathbf{v} \in \mathcal{S}_+^{p-i}$, this condition is automatically satisfied.

Given the above minimal requirement, if the chain additionally is f -irreducible and aperiodic, then it converges to its stationary distribution (Theorem 7.4 in [11]). The first condition holds in our case because the proposal is strictly positive for all $\mathbf{v}, \tilde{\mathbf{v}} \in \mathcal{S}_+^{p-1}$. A sufficient condition for aperiodicity is that the probability of remaining in the same state for the next step is strictly positive, that is, $P(f(\mathbf{v}) \geq f(\tilde{\mathbf{v}})) > 0$. In our case, we have

$$P(f(\mathbf{v}) \geq f(\tilde{\mathbf{v}})) = P(v_1^i \geq \mathbb{I}_{\geq 0}(\tilde{v}_1)\tilde{v}_1^i) = P(v_1 \geq \tilde{v}_1, \tilde{v}_1 \geq 0) + P(\tilde{v}_1 \leq 0).$$

Expanding the second summand and using the fact that $v_1 \geq 0$, we obtain

$$P(\tilde{v}_1 \leq 0) = P(\epsilon_1 \leq 0) - P(-v_1 \leq \epsilon_1 \leq 0) = \frac{1}{2} - \int_0^{v_1} \frac{1}{\sqrt{2\pi}\sigma_\epsilon} e^{-s^2/(2\sigma_\epsilon^2)} ds, \quad (5)$$

Therefore $P(f(\mathbf{v}) \geq f(\tilde{\mathbf{v}}))$ is strictly positive, the chain is aperiodic, and our proposed algorithm converges to f .

Some additional insights can be gained on the convergence of the algorithm when the variance σ_ϵ^2 increases. From Eq. (4), we observe that in such scenario, the proposal density approaches to a constant, yielding an independent Metropolis algorithm [11]. The expression for such limiting proposal, which coincides with the inverse sphere volume, is

$$\lim_{\sigma_\epsilon \rightarrow \infty} q(\tilde{\mathbf{v}} | \mathbf{v}) = \frac{\Gamma((p-i+1)/2)}{2\pi^{(p-i+1)/2}}. \quad (6)$$

In this scenario, denoting as C_f and C_q the integration constants for f and q respectively, taking $M \geq (C_q C_f)^{-1}$ we have for all $\mathbf{v} \in \mathcal{S}_+^{p-i}$ that $f(\mathbf{v}) \leq Mq(\mathbf{v})$. Therefore the chain is uniformly ergodic, and $2(1 - M^{-1})^n$ is an upper bound for the total variation norm between the transition kernel after n iterations and the target distribution f (Theorem 7.8 in [11]). Furthermore, M^{-1} is also a lower bound for the expected acceptance probability.

4.2 Empirical Monitoring

The above theoretical analysis assures the convergence of the proposed Algorithm 2. Such convergence can also be empirically monitored in order to get insight on how to tune its hyper-parameters: the burn-in time t_b and the perturbation variance σ_ϵ^2 . This will be the focus of this subsection. For this task, the most challenging matrices are arguably high dimensional therefore we will focus on the case where $p = 1000$.

There is no standard assessment scheme to follow that will guarantee an expected behaviour for the Metropolis chain [11]. However, we can study the behaviour of some characteristic quantities. We have chosen to focus on the acceptance ratio, that is, the percentage of times that we have accepted the proposed value, so it can be thought of as an approximation for $P(f(\mathbf{v}) \leq f(\tilde{\mathbf{v}}))$.

Whether a high acceptance ratio is desirable or not depends on the particular chain designed. For our case, in Fig. 1 this quantity is depicted as a function of the row number and, complementarily, of the perturbation variance σ_ϵ^2 .

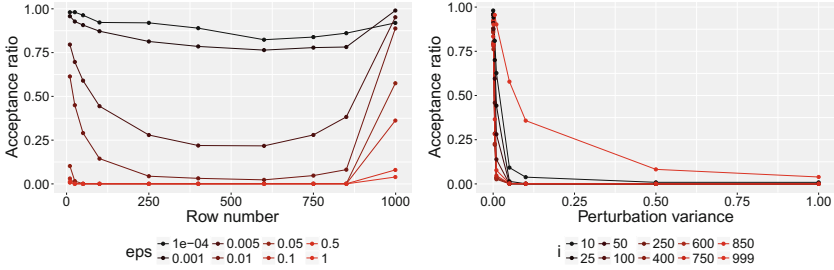


Fig. 1. Acceptance ratio as a function of the row number i (left) and the perturbation variance σ_ϵ^2 (right). eps: σ_ϵ^2 .

We observe how, as σ_ϵ^2 increases, the proposed value is rejected more often. This could be already expected by looking at Eq. (5) above, where we see that the second term goes to zero as σ_ϵ^2 increases, yielding $\lim_{\sigma_\epsilon^2 \rightarrow \infty} P(f(\mathbf{v}) \leq f(\tilde{\mathbf{v}})) \leq 1/2$. Furthermore, as σ_ϵ increases the proposal distribution is more similar to the uniform density on the $(p-i)$ -dimensional sphere (Eq. (6)), which also hints the higher rejection rate.

The row number i also has a significant influence on the acceptance ratio. Recall that $1 \leq i \leq p-1$, $\mathbf{v} \in \mathcal{S}_+^{p-i}$ and $f(\mathbf{v}) \propto v_1^i$, therefore as the row number i increases the target distribution f approaches a delta function, and the dimensionality of \mathbf{v} decreases. Therefore it is reasonable to assume that the larger i is, the smaller σ_ϵ^2 should be for achieving a high acceptance ratio, since it means that we are proposing new states that are, with high probability, very close to the current state.

The above conclusions are further illustrated in Fig. 2, where we have plotted the contour lines of the acceptance ratio surface as a function of σ_ϵ^2 and the row number i . Observe that small values for σ_ϵ always lead to high acceptance ratios, however this might not always be desirable since it can be a sign of slow convergence. By contrast, a low acceptance ratio can be expected when approaching to a delta in moderately high dimensions, as is the case for row numbers approximately between 250 and 750.

5 Performance Analysis

In this section we will compare our method, in terms of computational performance, with the existing approaches in the literature for the same task: uniform sampling of correlation matrices. We will generate 5000 correlation matrices of dimension $p = 10, 20, \dots, 100$ using our algorithm, the vine and onion methods of Lewandowski et al. [5], and the polar parametrization of Pourahmadi [4].

Our algorithm has been implemented in R [12]. The vine and onion methods are available in the function `genPositiveDefMat` from the R package `clusterGeneration`¹, provided by the authors. Since we have not found an

¹ <https://CRAN.R-project.org/package=clusterGeneration>.

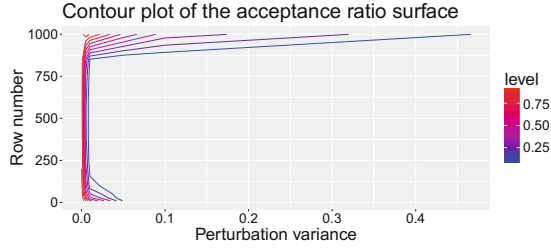


Fig. 2. Contour lines of the acceptance ratio surface. Level: magnitude of the acceptance ratio.

implementation of the polar parametrization method of Pourahmadi [4], we have developed our own function, also in R, mimicking the method therein described. For our method, based on the analysis of the previous section we have fixed $\sigma_\epsilon = 0.01$ and $t_b = 1000$, which have empirically provided good convergence results. The experiment has been executed on a machine equipped with Intel Core i7-5820k, 3.30 GHz \times 12 and 16 GB of RAM.

The results of the experiment are shown in Fig. 3. We observe that our method is faster than all of the existing approaches in the literature. The polar parametrization method has the worst performance, several orders of magnitude slower than the other algorithms. This can be explained by the use of inverse transformation sampling for simulating the angles. By contrast, our method achieves highly competitive results by taking advantage of the direct representation provided by the Cholesky factorization, as well as the simple form of the target distribution and the proposed values on each iteration. The scripts used for generating the data and figures described throughout the paper are publicly available, as well as the implementation of the algorithms described², so all the above experiments can be replicated.

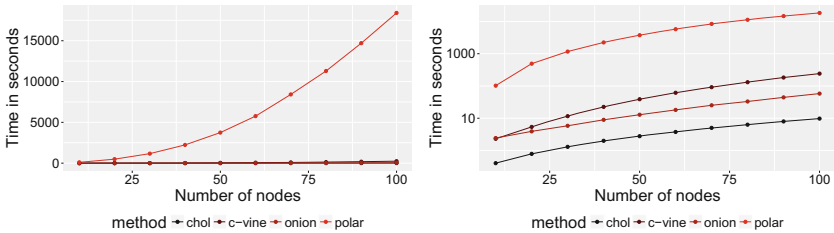


Fig. 3. Execution time of available methods for uniform sampling of correlation matrices, both in linear (left) and logarithmic (right) scale. *chol*: our proposal; *c-vine*, *onion*: methods by [5]; *polar*: method by [4].

² <https://github.com/ireneccrsn/rcor>.

6 Conclusions

In this paper we have proposed a Metropolis-Hastings method for uniform sampling of correlation matrices. We have studied its properties, both theoretically and empirically, and shown fast convergence to the target uniform distribution. We have also executed a comparative performance study, where our approach has yielded faster results than all of the related approaches in the literature.

In the future, we would like to further explore variants of our Markov chain algorithm, such as the independent Metropolis or adaptive schemes. We would also like to expand on the theoretical convergence analysis of such variants, as well extend the empirical convergence monitoring to other relevant quantities apart from the acceptance ratio.

Acknowledgements. This work has been partially supported by the Spanish Ministry of Economy, Industry and Competitiveness through the Cajal Blue Brain (C080020-09; the Spanish partner of the EPFL Blue Brain initiative) and TIN2016-79684-P projects; by the Regional Government of Madrid through the S2013/ICE-2845-CASI-CAM-CM project; and by Fundación BBVA grants to Scientific Research Teams in Big Data 2016. I. Córdoba has been supported by the predoctoral grant FPU15/03797 from the Spanish Ministry of Education, Culture and Sports. G. Varando has been partially supported by research grant 13358 from VILLUM FONDEN.

References

1. Marsaglia, G., Olkin, I.: Generating correlation matrices. *SIAM J. Sci. Stat. Comput.* **5**(2), 470–475 (1984)
2. Holmes, R.: On random correlation matrices. *SIAM J. Matrix Anal. Appl.* **12**(2), 239–272 (1991)
3. Fallat, S., Lauritzen, S., Sadeghi, K., Uhler, C., Wermuth, N., Zwiernik, P.: Total positivity in Markov structures. *Ann. Stat.* **45**(3), 1152–1184 (2017)
4. Pourahmadi, M., Wang, X.: Distribution of random correlation matrices: hyper-spherical parameterization of the Cholesky factor. *Stat. Prob. Lett.* **106**, 5–12 (2015)
5. Lewandowski, D., Kurowicka, D., Joe, H.: Generating random correlation matrices based on vines and extended onion method. *J. Multivar. Anal.* **100**(9), 1989–2001 (2009)
6. Laurent, M., Poljak, S.: On the facial structure of the set of correlation matrices. *SIAM J. Matrix Anal. Appl.* **17**(3), 530–547 (1996)
7. Diaconis, P., Holmes, S., Shahshahani, M.: Sampling from a Manifold, *Collections*, vol. 10, pp. 102–125. Institute of Mathematical Statistics (2013)
8. Eaton, M.L.: *Multivariate Statistics: A Vector Space Approach*. Wiley, Hoboken (1983)
9. Mardia, K., Jupp, P.: *Directional Statistics*. Wiley, Hoboken (1999)
10. Pukhila, T.M., Rao, C.R.: Pattern recognition based on scale invariant discriminant functions. *Inf. Sci.* **45**(3), 379–389 (1988)
11. Robert, C., Casella, G.: *Monte Carlo Statistical Methods*. Springer, New York (2004). <https://doi.org/10.1007/978-1-4757-4145-2>
12. R Core Team: *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna (2018)



Novel and Classic Metaheuristics for Tuning a Recommender System for Predicting Student Performance in Online Campus

Juan A. Gómez-Pulido¹✉, Enrique Cortés-Toro²,
Arturo Durán-Domínguez¹, Broderick Crawford³, and Ricardo Soto³

¹ University of Extremadura, 10003 Cáceres, Spain
{jgomez, arduran}@unex.es

² Universidad de Playa Ancha, 2340000 Valparaíso, Chile
enrique.cortes@upla.cl

³ Pontificia Universidad Católica de Valparaíso, 2374631 Valparaíso, Chile
{broderick.crawford, ricardo.soto}@pucv.cl

Abstract. The prediction of the students' performance allows to improve the learning process using the online campus tools. In this context, recommender systems are useful for prediction purposes. This collaborative filtering tool, predicts the unknown performances analyzing the database that contains the performance of the students for particular tasks, considering matrix factorization and stochastic gradient descent. If we consider a fixed number of latent factors, the prediction error is mainly influenced by two parameters: learning rate and regularization factor. The best settings for these parameters is an optimization problem that can be tackled by soft computing techniques. In this work, we analyze three solving methods to select the optimal values of both parameters: a simple direct search, a classic evolutionary algorithm, and a novel metaheuristic. The results show the advantages of using metaheuristics instead of direct search in accuracy and computing effort terms.

Keywords: Metaheuristics · Genetic algorithms
Vapour-liquid equilibrium · Recommender systems · Prediction
Matrix factorization · Stochastic gradient descent · Learning rate
Regularization factor

1 Introduction

Collaborative filtering methods as Recommender Systems (RS) [1] can predict the future behaviour of a user according to the past information of his activity in several tasks, and the activity of the other users in the same task. A RS can be applied to online environments where predicting the users' behaviour is useful. This is the case of the Predicting Student Performance (PSP) problem, where the students' performance is predicted for academic tasks [2].

Our research framework deals with datasets extracted from the online campus databases of the University of Extremadura, Spain. Each dataset is represented by a matrix P of S rows by I columns, where S is the number of students, I is the number of academic tasks, and $p_{s,i}$ represents the performance or score of the student s for the task i . Some cells in P could be unknown because of tasks not completed or students not attending (D^{unk}), so they can be predicted from a mathematical model built by Matrix Factorization (MF) [3]. In this model, P can be approximated by $P \approx W1 \times W2^T$ [4], where $W1$ and $W2$ are smaller matrices of sizes $S \times K$ and $I \times K$ respectively.

The prediction model based on MF considers the number of latent factors K implicit in the relationship mentioned before. The model is implicitly able to encode latent factors of students and tasks. The intuition behind using MF is that there should be some latent features that determine how a student performs a task. However, it is difficult to establish the proper number of such latent factors. In our work, we have considered different values of K depending on the dataset, according to experiments done for this particular purpose.

From $W1$ and $W2$, the unknown value $\hat{p}_{s,i}$ can be predicted by (1).

$$\hat{p}_{s,i} = \sum_{k=1}^K (w1_{s,k} \times w2_{i,k}) = (W1 \times W2^T)_{s,i} \quad (1)$$

$W1$ and $W2$ are calculated by Gradient Descent (GD) [5], which is very efficient dealing with large data sets [6]. This algorithm updates $W1$ and $W2$ iteratively using training data D^{train} chosen from the known values in P . The iterations try to minimize the error done in the prediction, considering the Root Mean Squared Error criterion (RMSE) (2). The error is calculated for test data D^{test} , a smaller subset of known values in P .

$$RMSE = \sqrt{\frac{\sum_{s,i \in D^{test}} (p_{s,i} - \hat{p}_{s,i})^2}{|D^{test}|}} \quad (2)$$

2 Optimization Problem

There are two important parameters involved in GD: the learning rate β and the regularization factor λ . The first one is needed to determine the gradient, whereas the second one prevent the over-fitting. They are constant real numbers, although their values are set without any strict rule, simply according to other works or after few tests. In any case, both parameters have a certain influence on the prediction accuracy. Figure 1 helps us understand this fact. This figure was generated from a Direct Search (DS) method, where thousands predictions were calculated for different values of the pair (β, λ) . The plot on the left shows the values of RMSE, whereas the plot on the right shows the GRMSE metric. GRMSE is the same RMSE after applying a proportional function that highlights the maximum and minimum peaks, in order to make easier visualizing the

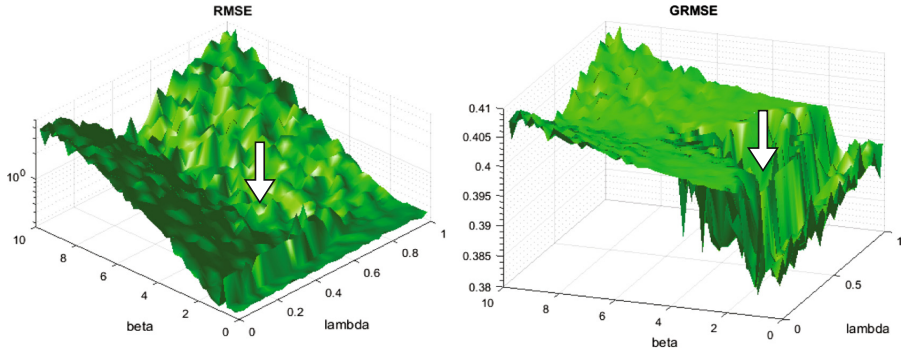


Fig. 1. Direct search of the minimum RSME corresponding with the best pair (β, λ) , which is pointed out by the arrow. The plot on the right corresponds to the GRMSE metric to display better that location.

minimums. We can check how a global minimum exists, together a high number of local minimums. This fact was verified considering other datasets.

The selection of the pair (β, λ) affects the prediction accuracy. The question is: How can we set the optimal values of (β, λ) without performing a thorough direct search of high computational effort? In this work we propose using metaheuristics to find good values for both parameters with low computational effort.

3 Metaheuristics Description

When the space of possible solutions is very big, metaheuristics [7] allow to intensify the search in the nearness of a promising solution. They are approximate, non-deterministic algorithms based on heuristic search, according to two different focus: trajectory or population. In this work we want to solve the optimization problem confronting a novel trajectory-based metaheuristic with a classic population-based one.

The objective function $f(X)$ considered by the metaheuristics is the function to be optimized, where X is a set of $NVAR$ decision variables x_i . In this work we consider $NVAR = 2$ decision variables, $x_1 = \beta$ and $x_2 = \lambda$, and $f(X)$ is $RMSE$ (2). Therefore, each evaluation of the objective function implies to perform a prediction from a matrix factorization model calculated using gradient descent.

3.1 Vapour-Liquid Equilibrium Metaheuristic

Vapour-Liquid Equilibrium (VLE) is a trajectory algorithm with local search, multiple reboot and autonomous adjustment of the search space, inspired by the liquid-vapor equilibrium, present in many chemical engineering processes [8]. Unlike the rest of the metaheuristics, this algorithm has embedded a simple mathematical model of liquid-vapor equilibrium simulation and the bisection

numerical method [9] for determining the roots of the model's equations. VLE is able to solve complex optimization problems in real domains.

When the chemical compounds form ideal mixtures, the distribution of these between both phases can be modeled applying, simultaneously, the Law of Raoult and the Law of Dalton [10]. The equations of the model allow the calculation of the bubble point of a mixture in liquid phase and the calculation of the dew point of a mixture in vapor phase at a certain pressure. The bubble point is the temperature (T_{BP}) at which a liquid mixture begins to boil, whereas the dew point is the temperature (T_{DP}) at which a mixture of vapors begins to condense. Both processes represent liquid-vapor equilibrium states that are based on the principle of mass conservation and the phase equilibrium.

The determination of roots of the equations that model these conditions establishes the values of the movement operators (bubble and dew operators). During the search for an optimal, the operators are applied on domains in parallel to the real domain of each decision variable. These domains represent molar fractions of the most volatile chemical species of binary mixtures. There are as many binary mixtures as decision variables of the optimization problem. The operators are used in the exploration and exploitation phases, creating and updating neighborhood structures around the best solution found in the previous iteration by changing just one decision variable each time. The values of the operators depend on the chemical nature of the most volatile component of each system, the saturation temperature, and the total system pressure. The application of the operators is given by (3) and (4), where l and v are the molar fractions of the most volatile chemical component of each mixture (subscript 1). The mole fraction of the most volatile component at iteration t , is calculated by a linear transformation of the decision variable $x_d(t)$, from the real domain $[min, max] \in R$, to the value of the parallel domain $[newmin, newmax] = [0, 1] \in R$.

$$l_1(t+1) = v_1(t) = BubOp l_1(t) = K_1 l_1(t) \quad (3)$$

$$l_1(t+1) = DewOp v_1(t) = \frac{1}{K_1} v_1(t) \quad (4)$$

The input information is limited to the criteria for stopping the algorithm, its execution parameters and the specification of the objective function and number of decision variables. The termination criteria are the number of movements to be performed (M) and the number of restarts to try (R), with $R \leq M$. The parameters with highest incidence in the search process are α (it autonomously adjusts the size of the search subset of the decision variables) and β (the probability of acceptance of worse solutions than the best solution found so far). Each time the algorithm restarts, it creates a new starting solution and new search neighborhoods guided by the values of the molar fractions that suggest where there could be at least one local optimum. The possible values of α are odd numbers greater than or equal to 3. The algorithm calculates the probability of acceptance of a possible solution in a random way and compares it with β . If the solution is not accepted, the algorithm restarts in another region of the

search space, either conserving or changing the chemical species of the mixtures according to the user specifications. The characterization of chemicals is carried out by means of their vapor pressure, according to Antoine’s equation [10]. On the other hand, for a given number of experiments, the output information includes the optimal value found, the location of the corresponding solution, the convergence graphs of all the experiments carried out, and the box plot.

3.2 Genetic Algorithm

Genetic Algorithms (GA) [11] are well known population-based Evolutionary Algorithms (EAs) [12]. A population is the set of individuals X that evolves along generations, where an individual is a pair of decision variables (β, λ) . We consider the fitness of an individual (the error done in the prediction) as its objective value with regard to the entire population.

The GA starts generating an initial population. Next, all the individuals in the population are evaluated. From here, six phases are performed sequentially. The assignment phase occurs at the beginning of the first iteration (generation) of the GA, where a fitness value is assigned to each individual. Next, in the selection phase the best individuals (parents) are chosen for crossover according to their fitness values. The recombination phase crosses the parents to generate new individuals (offspring), updating the fitness values. In the mutation phase, particular mutations are applied to some individuals of the offspring, updating the fitness values accordingly. Next, the offspring is evaluated in the evaluation phase. Finally, the reinsertion phase includes the offspring individuals in the population. Here, the algorithm goes back to the assignment phase again. These phases are performed iteratively along many generations, up to a stop criterion is reached. This criterion can be a predefined number of generations.

4 Experimental Framework

The dataset considered for the experiments was extracted from a real virtual classroom corresponding to a medium-size representative case in a university online campus. The original database was composed of 128 students and 16 tasks. Nevertheless, many of these students and tasks were removed after a detailed analysis because of their limited academic activity; otherwise, these data could introduce noise in the prediction. Thus, the performance matrix P , built after applying the corresponding filters, represents an academic environment where there is not any student with less than 60% of activity in all the evaluation tasks, and the task with the minimum students’ activity has 90% of participation.

Figure 2 represents the performance matrix P , composed of $S = 95$ students, $I = 5$ tasks and 18 unknown performances. We used all the known performances to train the prediction model, and chose 92 performances to test the model.

Table 1 shows the values of the main parameters involved in the experiments. The dataset and MF model is characterized by the number of students and tasks,

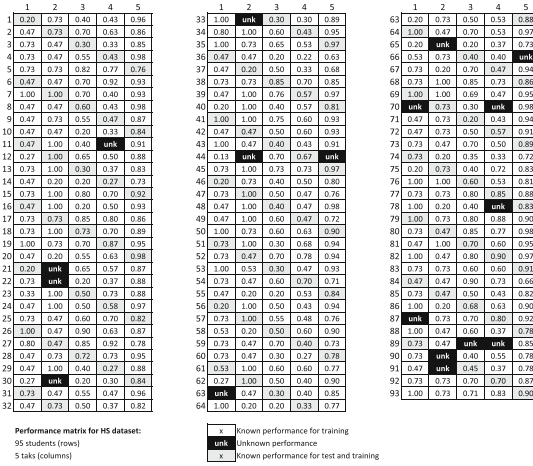


Fig. 2. Dataset: performance matrix extracted from a real subject.

training and test data sizes, and number of latent factors. Gradient descent initializes the values of $W1$ and $W2$ randomly according to a fixed standard deviation ($stddev$); this stochastic nature needs several runs for each case (num_runs). In addition, we choose a fixed number of iterations (num_iter) as stop criterion ($itermethod$) and ignore the possible bias of the students ($biased$). Finally, we consider fixed searching areas, limited by the maximum and minimum values for β and λ (β_{max} , β_{min} , λ_{max} , and λ_{min} respectively); these limits are applied to all the solving methods: VLE, GA and DS.

Table 1. Main parameters of the experimental framework: dataset, matrix factorization, gradient descent and search area.

Dataset&MF	GD	Limits
S	95 num_runs 5	β_{min} 0.0001
I	5 itermethod 0	β_{max} 10
Dtest	92 num_iter 300	λ_{min} 0.0001
Dunkn	18 biased 0	λ_{max} 1
K	64 stddev 0.1	

5 Experimental Framework

Table 2 shows the main settings for the solving methods. The number of evaluations of the fitness function ($funccount$) or predictions for each case (β, λ)

should be the same for VLE, GA and DS in order to do a realistic comparison. Therefore, the parameters of each solving method were tuned to guarantee a similar computational effort with regard to the number of predictions done.

Table 2. Main parameters of the three solving methods: VLD, GA and DS.

VLE		GA		DS	
M	200	popsize	106	N_β	50
R	200	maxgen	16	N_λ	36
funccount	1,800	funccount	1,802	funccount	1,800
VLE_α	5				
VLE_β	0.99(A), 0.01(B)				

Each experiment (two for VLE and one for GA) consisted of 41 runs in order to obtain enough stats, except for DS, since it is a deterministic method. The results are shown in Table 3, from which we can conclude that the minimum RMSE found ($fval$) was obtained by GA, followed by VLE and DS. In other words, both metaheuristics improved the results of a simple direct search. Although the computing effort is lower by DS, we should note that the main goal is accuracy. The optimal found by GA could be found too by DS considering more values for (β, λ) , in other words, a higher number of predictions expressed by $funccount$, with the corresponding increased computing time. Both VLE and GA show fast convergence when searching the optimal solution, obtaining good solutions long before DS with lesser computing effort.

Table 3. Comparison VLE-GA-DS.

Tests (41 runs):		VLE-A	VLE-B	GA	DS (1 run)
Optimum found:	$fval$	0.267	0.268	0.245	0.271
	β	3.299	2.329	2.599	2.400
	λ	0.604	0.699	0.589	0.560
Stats:	Mean	0.285	0.314	0.255	
	Maximum	0.330	0.646	0.261	
	Std. dev.	0.012	0.069	0.001	
Time (1 run):		8.5 min	17.5 min	17.7 min	4.2 min

6 Conclusions

We have applied two metaheuristics (novel VLE and classic GA) to solve an optimization problem with regard to the stochastic gradient descent when building

a mathematical model based on matrix factorization for predicting the students performance, basing on recommender systems. The optimization problem tries to find the best values of learning rate and regularization factor in order to obtain predictions with minimum error. The metaheuristics have been compared to a simple direct search of these parameters. The obtained results encourage us to consider metaheuristics rather than exhaustive searches.

Since many data mining and machine learning approaches have been applied to predict the students' behaviour [13], our proposal could be interesting to enhance the prediction accuracy provided by these other methods.

Finally, we would like to consider more and larger datasets of different characteristics in the future, in order to check if the advantages of metaheuristics with regard to simple direct search are the same in all the cases.

Acknowledgments. This work was funded by the Government of Extremadura and the State Research Agency (Spain) under the projects IB16002 and TIN2016-76259-P respectively. PhD. B. Crawford and PhD. R. Soto are supported by grants CONICYT/FONDECYT/REGULAR/1171243 and 1160455 respectively. MSc. E. Cortés-Toro is supported by grant INFPUCV 2015.

References

1. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: *Recommender Systems. An Introduction*. Cambridge University Press, Cambridge (2011)
2. Thai-Nghe, N., Drumond, L., Horvath, T., Krohn-Grimberghe, A., Nanopoulos, A., Schmidt-Thieme, L.: Factorization techniques for predicting student performance. In: *Educational Recommender Systems and Technologies: Practices and Challenges*, pp. 129–153. IGI-Global (2012)
3. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. *Computer* **42**, 30–37 (2009)
4. Rendle, S., Schmidt-Thieme, L.: Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In: *Proceedings of the 2008 ACM Conference on Recommender Systems*, pp. 251–258 (2008)
5. Koren, Y.: Factor in the neighbors: scalable and accurate collaborative filtering. *ACM Trans. Knowl. Discov. Data* **4**, 1–24 (2010)
6. Bottou, L.: Large-scale machine learning with stochastic gradient descent. In: Lechevallier, Y., Saporta, G. (eds.) *COMPSTAT 2010*, pp. 177–186. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-7908-2604-3_16
7. Gendreau, M., Potvin, J. (eds.): *Handbook of Metaheuristics*. Springer, Heidelberg (2010). <https://doi.org/10.1007/978-1-4419-1665-5>
8. Crawford, B., Soto, R., Cortés, E., Astorga, G.: A new thermodynamic equilibrium-based metaheuristic. In: Silhavy, R., Silhavy, P., Prokopova, Z. (eds.) *CoMeSySo 2017. AISC*, vol. 661, pp. 336–346. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-67618-0_31
9. Chapra, S., Canale, R.: *Numerical Methods for Engineers*. McGraw-Hill, New York (2015)
10. Smith, J.M., Van Ness, H., Abbott, M.: *Introduction to Chemical Engineering Thermodynamics*. McGraw-Hill, New York (2005)

11. Reeves, C., Rowe, J.: Genetic Algorithms. Principles and Perspectives. A guide to GA Theory. Kluwer Academic Publisher, Norwell (2003). EE.UU
12. Fogel, G., Corne, D.: Evolutionary Computation in Bioinformatics. Morgan Kaufmann Publishers, Burlington (2003)
13. Roy, S., Garg, A.: Analyzing performance of students by using data mining techniques: a literature survey. In: Proceedings of 2017 IEEE International Conference on Electrical, Computer and Electronics, UPCON 2017, pp. 130–133 (2018)



General Structure Preserving Network Embedding

Sinan Zhu(✉) and Caiyan Jia(✉)

School of Computer and Information Technology and Beijing Key Lab of Traffic Data Analysis and Mining, Beijing Jiaotong University, Beijing 100044, People's Republic of China
zhusinan0601@163.com, cyjia@bjtu.edu.cn

Abstract. Network embedding has attracted increasing attention in recent years since it represents large scale networks in low-dimensional space and provides an easier way to analysis networks. Existing embedding methods either focus on preserving the microscopic topology structure, or incorporate the mesoscopic community structure of a network. However, in the real world, a network may not only contain community structure, but have bipartite-structure, star-structure or other general structures, where nodes in each cluster have similar patterns of connections to other nodes. Empirically, general structure is important for describing the features of networks. In this paper, based on nonnegative matrix factorization framework, we propose GS-NMF which is capable of integrating topology structure and general structure into embedding process. The experimental results show that GS-NMF overcomes the limitation of previous methods and achieves obvious improvement on node clustering, node classification, and visualization.

Keywords: Network embedding · General structure
Nonnegative matrix factorization

1 Introduction

Network is ubiquitous in our life. Many systems can be abstracted into networks to characterize all kinds of relationships, such as social relationships between people, predatory relationships between creatures or link relationships between web pages and so on. The traditional way of representing networks is to convert these relationships into forms of adjacency matrices, where each element represents the relationship between pairs of nodes. However, great challenges on storage and computational complexity arise when process large-scale networks, because in face of the various networks, dimension of matrix ranges from

Supported by the National Nature Science Foundation of China No. 61473030 and No. 61632004, the Fundamental Research Funds for the Central University No. 2017JBM023.

hundreds to millions even more. Moreover, most of the adjacency matrices are sparse, it may lead to poor results for network analysis tasks.

Network embedding is effective for representing networks by learning a low-dimensional vector representation for each vertex in a network, the new representations always show better performance in network analysis tasks including node clustering and classification [6, 26, 29], visualization [6, 28], link prediction [26, 28, 31], social relation extraction [27], etc. Many existing methods focus on preserving the topology structure, and they are mainly based on the assumption that nodes with high topological similarity should be represented closely. Some typical methods, such as DeepWalk [22], LINE [25], Node2vec [7], SDNE [28], capture topology structure first by using low-order proximities or contextual information and then learn vector representation of each node. Other methods like GraRep [3], DNGR [4], NEU [31] further take advantage of higher order proximities for enriching the structural information. However, as claimed in M-NMF [29], those methods ignore the mesoscopic community structure contained in networks. The representations of nodes in the same community should be more similar than those belonging to different communities. Therefore, M-NMF aims to preserve microscopic structure and mesoscopic community structure of a network simultaneously. However, in the real world, a network may not only contain community structure, but also have bipartite-structure [17], star-structure [23], core-periphery structure [1] or other general structures [24], in which nodes in each class have similar patterns of connections to other nodes. In this study, we propose a network embedding method called GS-NMF (General Structure preserved network embedding based on NMF framework) which is able to preserve both microscopic topology structure and mesoscopic general structure contained in a network.

Our main contributions are summarized as follows.

- We use T -order signal propagation [9] to store high-order proximities between pairs of nodes. It overcomes the shortcoming of the first-order proximity which is not applicable to bipartite or multipartite networks.
- Nonnegative matrix tri-factorization is used to capture general structures. We have observed that the middle matrix in the tri-factorization has the similar significance as the block matrix in stochastic block models which characterizes the probability of connecting between classes [24].
- A joint NMF framework is employed to preserve both topology structure and general structure. The empirical studies on node clustering, classification and visualization show the effectiveness of GS-NMF.

2 Related Work

In recent years, network embedding has attracted many researchers attention, its pioneer work can be traced back to some traditional dimensionality reduction methods, such as Isomap, LLE and LE. However, these traditional methods cannot preserve the topology structure. Therefore, some outstanding practical works have been put forward to deal with this shortage. DeepWalk [22] first employs the

idea of natural language processing to establish an embedding model, in which the sequence of nodes that obtained from random walk is treated as contextual information in SkipGram model. LINE [25] defines the conditional probability of first-order proximity and second-order proximity, then obtains the new vector representation of each node by minimizing the distance between conditional probability and empirical probability. Based on the idea of LINE, GraRep [3] is proposed to build a matrix to store higher order proximity and then perform SVD to get specific representation. DNGR [4] exploits higher order structural information as the input of deep neural network to learn latent representations. Node2vec [7] also adopts SkipGram model, but combines two strategies, DFS and BFS, to choose the context of nodes. In addition, a semi-supervised embedding model SDNE [28] employs deep encoder to preserve local structure. A fast network embedding method NEU [31] is developed by implicitly approximating higher order proximities.

Besides topology structure, nodes in networks are often attached with attribute or content information. Therefore, CANE [26], TADW [32] and LANE [10] are proposed to represent networks by integrating content information of nodes. In addition, HOPE [20] preserves high order proximity and asymmetrical transmission information of large-scale directed networks. What is more, some embedding methods such as matapath2vec [6], aims to represent heterogeneous networks. M-NMF [29] learns microscopic structure and mesoscopic community structure of networks simultaneously by using a joint nonnegative matrix factorization framework. It exerts a good effect on networks with community structures. However, when the target network contains bipartite structure or other general structures, M-NMF might perform poorly in analysis tasks due to the lack of mesoscopic structure.

3 GS-NMF: General Structure Preserving Network Embedding

In this section, we present the details of GS-NMF. First we provide some notations that will be used in our paper. Suppose that there is a set of vertices V and a set of edges E constructing a network $G(V, E)$, which contains n nodes and e edges. $\mathbf{A} = [A_{ij}] \in \mathbf{R}^{n \times n}$ represents the binary adjacency matrix of a network. The goal of GS-NMF is to embed a network $\mathbf{A} \in \mathbf{R}^{n \times n}$ into low-dimensional space $\mathbf{U} \in \mathbf{R}^{m \times n}$, in which $m (m \leq n)$ is the dimension of new representation.

3.1 M-NMF Model

First, we briefly introduce the preliminary work M-NMF [29]. It adopts the matrix form to represent modularity function [18], which measures the quality of community partitioning. Community structure and microscopic structure are

integrated into a joint nonnegative matrix factorization framework to perform network embedding. The objective function is:

$$\begin{aligned} \min_{\mathbf{M}, \mathbf{U}, \mathbf{F}, \mathbf{C}} \quad & \|\mathbf{S} - \mathbf{M}\mathbf{U}^T\|_F^2 + \alpha \|\mathbf{H} - \mathbf{U}\mathbf{C}^T\|_F^2 - \beta \text{tr}(\mathbf{H}^T \mathbf{B} \mathbf{H}), \\ \text{s.t.} \quad & \mathbf{M} \geq 0, \mathbf{U} \geq 0, \mathbf{H} \geq 0, \mathbf{C} \geq 0, \text{tr}(\mathbf{H}^T \mathbf{H}) = n. \end{aligned} \quad (1)$$

Where $\mathbf{S} \in \mathbf{R}^{n \times n}$ represents the similarity matrix which capture the first-order and second-order proximity, $\mathbf{M} \in \mathbf{R}^{n \times m}$ is a nonnegative auxiliary matrix, $\mathbf{C} \in \mathbf{R}^{k \times m}$ is regarded as the new representation of k classes. \mathbf{H} indicates the community membership of each node. The element of modularity matrix \mathbf{B} is calculated by $B_{ij} = A_{ij} - \frac{d_i d_j}{2e}$, where d_i represents the degree of the node i . Hence, the modularity of a network partition can be represented as

$$Q = \frac{1}{4e} \sum_{ij} (A_{ij} - \frac{d_i d_j}{2e}) h_i h_j = \frac{1}{4e} \mathbf{H}^T \mathbf{B} \mathbf{H} = \text{tr}(\mathbf{H}^T \mathbf{B} \mathbf{H}). \quad (2)$$

Where h_i is the community indicator of node i . Thus, maximizing \mathbf{Q} is equivalent to minimize the third term in Eq. (1). According to the second term of Eq. (1), $\mathbf{U}\mathbf{C}^T$ is expected to approach \mathbf{H} as closely as possible to learn the community indicator information contained in matrix \mathbf{H} . Therefore, M-NMF achieves the purpose of preserving community structures and microscopic structures by the learning of \mathbf{U} , which is simultaneously reflected by \mathbf{S} and \mathbf{H} . However, when networks contain other general structures, M-NMF might behave poorly on account of the limitation of modularity function.

3.2 GS-NMF Model

In order to overcome the limitation of M-NMF that only detects community structures, we propose GS-NMF which is capable of integrating the topology structure and the general structure of networks. Inspired by the previous studies [21], we find that nonnegative matrix tri-factorization can exert a great effect on discovering the general structure. In the process of factorization, it can learn class indicator matrix and inter-class relation matrix simultaneously, where the latter represents the probability of interaction between two classes. We find the inter-class relation matrix has the similar significance as the block matrix in stochastic block models [24]. Both methods detect general structure of networks by learning patterns of interactions between classes. Therefore, based on tri-factorization, we propose an objective function to capture general structure, such that

$$\min \|\mathbf{A} - \mathbf{F}\mathbf{V}\mathbf{F}^T\|_F^2 \quad \text{s.t.} \quad \mathbf{F} \geq 0, \mathbf{V} \geq 0. \quad (3)$$

Where $\mathbf{F} \in \mathbf{R}^{n \times k}$ represents the class membership matrix, the value of each element indicates the probability of a node belonging to a given class, and k is the number of classes. $\mathbf{V} \in \mathbf{R}^{k \times k}$ represents the inter-class relation matrix which reveals the relationship between two classes. Each element \mathbf{V}_{ij} indicates

the strength of association between class i and class j . For networks with community structures, the nodes within the same class are closely connected and nodes are sparsely connected in different classes. As a result, the elements of main diagonal in matrix \mathbf{V} are usually larger than the other elements. When networks contain bipartite structures, the patterns of connection are opposite of community structures. Therefore, the elements of vice diagonal in matrix \mathbf{V} tend to be larger than others. As for networks which contain other general structures, the relational patterns between classes can also be captured by \mathbf{V} . Therefore, the objective function of Eq. (1) has been changed as follows:

$$\begin{aligned} \min_{\mathbf{M}, \mathbf{U}, \mathbf{F}, \mathbf{C}, \mathbf{V}} \quad & \|\mathbf{S} - \mathbf{M}\mathbf{U}^T\|_F^2 + \alpha\|\mathbf{F} - \mathbf{U}\mathbf{C}^T\|_F^2 + \beta\|\mathbf{A} - \mathbf{F}\mathbf{V}\mathbf{F}^T\|_F^2 \\ \text{s.t.} \quad & \mathbf{M} \geq 0, \mathbf{U} \geq 0, \mathbf{F} \geq 0, \mathbf{C} \geq 0, \mathbf{V} \geq 0. \end{aligned} \quad (4)$$

Where \mathbf{S} is defined by the signal propagation method [9], in which each node of networks is regarded as a function of receiving and transmitting signals. The process of transmitting can be expressed as $S = (I + A)^T$, where I is the identity matrix, T indicates the number of transmitting. Therefore, each element \mathbf{S}_{ij} stores the number of shortest path between nodes i and j . Thus, T -order proximity is preserved in matrix \mathbf{S} which effectively capture high-order structural information.

4 Alternating Optimization

The above objective function is a nonconvex problem. Following the previous work of NMF [2], we use ADMM (Alternating Direction Method of Multipliers) framework to optimize the objective function. By fixing other parameters, we separate the optimization process of Eq. (4) into five subproblems respectively. Then we optimize the solution of subproblems by gradient descent to approximate to global optimal and obtain the updating rules of \mathbf{M} , \mathbf{U} , \mathbf{F} , \mathbf{C} and \mathbf{V} . First, we solve the problem of \mathbf{M} with \mathbf{U} fixed in $\|\mathbf{S} - \mathbf{M}\mathbf{U}^T\|_F^2$. Then, according to the method proposed in [12], the updating rule of \mathbf{M} is

$$\mathbf{M} \leftarrow \mathbf{M} \otimes \frac{\mathbf{S}\mathbf{U}}{\mathbf{M}\mathbf{U}^T\mathbf{U}}. \quad (5)$$

With \mathbf{M} , \mathbf{F} , \mathbf{C} fixed in joint NMF problem $\|\mathbf{S} - \mathbf{M}\mathbf{U}^T\|_F^2 + \alpha\|\mathbf{F} - \mathbf{U}\mathbf{C}^T\|_F^2$, the updating rule of \mathbf{U} is,

$$\mathbf{U} \leftarrow \mathbf{U} \otimes \frac{\mathbf{S}^T\mathbf{M} + \alpha\mathbf{F}\mathbf{C}}{\mathbf{U}(\mathbf{M}^T\mathbf{M} + \alpha\mathbf{C}^T\mathbf{C})}. \quad (6)$$

We fix \mathbf{U} and \mathbf{F} in $\|\mathbf{F} - \mathbf{U}\mathbf{C}^T\|_F^2$ then acquire the updating rule of \mathbf{C} ,

$$\mathbf{C} \leftarrow \mathbf{C} \otimes \frac{\mathbf{F}^T\mathbf{U}}{\mathbf{C}\mathbf{U}^T\mathbf{U}}. \quad (7)$$

We adopt Lagrange multiplier method and the Karush-Kuhn-Tucker conditions to solve the problem of \mathbf{V} in $\|\mathbf{A} - \mathbf{F}\mathbf{V}\mathbf{F}^T\|_F^2$, so we have

$$\mathbf{V} \leftarrow \mathbf{V} \otimes \frac{\mathbf{F}^T \mathbf{A} \mathbf{F}}{\mathbf{F}^T \mathbf{F} \mathbf{V} \mathbf{F}^T \mathbf{F}}. \quad (8)$$

Similarly, as for the problem of \mathbf{F} , we also adopt Lagrange multiplier method and KKT conditions on $\alpha\|\mathbf{F} - \mathbf{U}\mathbf{C}^T\|_F^2 + \beta\|\mathbf{A} - \mathbf{F}\mathbf{V}\mathbf{F}^T\|_F^2$, then we receive

$$\mathbf{F} \leftarrow \mathbf{F} \otimes \frac{\alpha\mathbf{U}\mathbf{C}^T + \beta\mathbf{A}\mathbf{F}\mathbf{V}^T + \beta\mathbf{A}^T\mathbf{F}\mathbf{V}}{\alpha\mathbf{F} + \beta\mathbf{F}\mathbf{V}\mathbf{F}^T\mathbf{F}\mathbf{V}^T + \beta\mathbf{F}\mathbf{V}^T\mathbf{F}^T\mathbf{F}\mathbf{V}}. \quad (9)$$

5 Initialization

It is widely accepted that the result of nonnegative matrix factorization is heavily affected by initial values of matrices. If unknown matrices are initialized with random values, the result often reaches a local optimal instead of global optimal. Maximum entropy initialization, adopted in EM algorithm [30] to avoid the local optimal is employed to initialize \mathbf{M} , \mathbf{U} , \mathbf{C} and \mathbf{V} . As we know, entropy increases with the uncertainty of initial value growing. Hence, for each matrix, we set their initial values with only a slight difference. This method effectively avoid the instability due to random initialization.

To obtain an appropriate initial value for \mathbf{F} , we adopt the idea of K-Rank-D [13] to extract k columns from \mathbf{S} . In detail, we select k columns with the first k largest comprehensive values, which characterizes the PageRank importance of nodes in a network as well as the distance between nodes with larger centralities. The comprehensive value is defined as $CV(i) = p_i \bar{\delta} / (\max_{i=1}^n(p_i) \max_{i=1}^n(\bar{\delta}))$, p_i indicates the PageRank centrality vector of node i which is calculated by the power method, and $\bar{\delta}$ is the dispersion of i to other nodes with higher PageRank centrality by $\bar{\delta}_i = \min_{j:p_j > p_i}(d_{ij})$, d_{ij} is the distance between node i and j .

6 Experiments and Results

6.1 Datasets

In order to compare the effectiveness among different methods, we apply four groups of real networks with different kinds of general structures, including clear community structure, bipartite structure, mixture structure and unclear community structure. The first group consists of Zachary Karate club network (Karate) [34], American college Football network (Football) [19], American political Books network¹(Book) and co-appearance network of characters in the novel Les Miserables (Lesmis) [11]. The second group contains adjective-noun network (Adjnoun) [17] and southern women’s activity network (Women) [5]. As for the third group, we use WebKB that contains webpages collected from computer

¹ <http://www.orgnet.com>.

Table 1. Dataset overview.

Network	Node	Edge	#class
Karate	34	78	2
Football	115	613	2
Lesmis	77	254	11
Book	105	441	3
Adjnoun	112	425	2
Women	32	89	2
Cornell	195	283	5
Texas	187	289	5
Washington	230	366	5
Wisconsin	265	469	5
Cora	2708	5278	5
Citeseer	3312	4560	6

science departments of four universities². They are Cornell, Texas, Washington and Wisconsin. The last group of networks includes citation networks Cora [14] and Citeseer [15]. The property of datasets are showed in Table 1.

6.2 Baselines and Experimental Settings

We take several baselines into consideration to demonstrate the effectiveness of GS-NMF algorithm. And we adopt two commonly used accuracy metric, NMI and PWF [33], to evaluate the performance of methods.

DeepWalk [22]. DeepWalk performs random walks over networks and employs Skip-Gram model to learn new representations. The parameters are set as window size $w=10$, walk length $t=40$, walks per node $r=40$.

LINE [25]. We learn LINE1 and LINE2 separately which preserve first-order and second-order proximity. Meanwhile, we set the number of negative samples as 5 and the starting value of learning rate as 0.025.

Node2vec [7]. It extends DeepWalk mainly in the choice of context nodes. We set window size w , walk length t and walks per node r the same as DeepWalk. Then we employ grid search on p and q and report the best results.

GraRep [3]. GraRep captures high-order proximity of a network and learn specific representation by SVD. We set the maximum transition step size be 5.

DNDR [4]. In DNDR, we also set the maximum transition step size as 5. Additionally, the neural networks for all datasets consist of 3 layers with the exception of Cora and Citeseer which consist of 4 layers.

² <http://linqs.cs.umd.edu/projects/projects/lbc/>.

M-NMF [29]. It learns new representation based on a joint nonnegative matrix factorization. Also we employ the grid search over parameters $\alpha, \beta \in \{0.1, 0.5, 1, 5, 10\}$ and select the best one.

In GS-NMF, α and β are set in the same way as M-NMF to adjust the weight of general structure. We set the order of signal propagation $T=3$ as [9].

6.3 Node Clustering

We use newly generated representations of the compared methods to perform node clustering. The dimension of new representation is set as 20 for the first 10 networks with small scales in Table 1. As for Cora and Citeseer, we set $m = 100$. We apply Kmeans to conduct node clustering. Due to the sensitivity of Kmeans to initial values, we repeat algorithm 10 times and report mean and standard deviation of NMI and PWF in Tables 2 and 3, respectively.

Table 2. NMI of clustering in real networks

	Deepwalk	LINE1	LINE2	Node2vec	GraRep	DNGR	M-NMF	GS-NMF
Karate	0.8364 (0.0000)	0.3159 (0.2331)	0.9311 (0.2526)	0.9677 (0.0975)	1.0000 (0.0000)	1.0000 (0.0000)	0.7009 (0.3406)	1.0000 (0.0000)
Football	0.8922 (0.0239)	0.8869 (0.0275)	0.8907 (0.0236)	0.8980 (0.0225)	0.9104 (0.0152)	0.8945 (0.0337)	0.9185 (0.0050)	0.9306 (0.0089)
Lesmis	0.8107 (0.0275)	0.7530 (0.0338)	0.7453 (0.0266)	0.8108 (0.0208)	0.7714 (0.0197)	0.7792 (0.0097)	0.7828 (0.0164)	0.8258 (0.0250)
Book	0.5583 (0.0196)	0.3366 (0.0815)	0.5455 (0.0367)	0.5774 (0.0222)	0.5312 (0.0000)	0.5296 (0.0445)	0.4192 (0.1019)	0.5921 (0.0096)
Adjnoun	0.0047 (0.0064)	0.0037 (0.0041)	0.2164 (0.0916)	0.0247 (0.0000)	0.4703 (0.0000)	0.0039 (0.0038)	0.0342 (0.0391)	0.5108 (0.0196)
Women	0.0109 (0.0021)	0.0014 (0.0020)	0.9119 (0.2000)	0.0067 (0.0088)	1.0000 (0.0000)	0.0193 (0.0165)	0.5589 (0.3069)	1.0000 (0.0000)
Cornell	0.0596 (0.0167)	0.0596 (0.0160)	0.1118 (0.0141)	0.0771 (0.0105)	0.0822 (0.0185)	0.0693 (0.0062)	0.1098 (0.0143)	0.1448 (0.0290)
Texas	0.0611 (0.0198)	0.0606 (0.0221)	0.1873 (0.0249)	0.1113 (0.0274)	0.1203 (0.0111)	0.0614 (0.0030)	0.1965 (0.0168)	0.2097 (0.0548)
Washington	0.0648 (0.0152)	0.0432 (0.0153)	0.2088 (0.0122)	0.1328 (0.0139)	0.0621 (0.0174)	0.0406 (0.0034)	0.1658 (0.0219)	0.1818 (0.0215)
Wisconsin	0.0780 (0.0185)	0.0641 (0.0206)	0.0841 (0.0142)	0.0832 (0.0166)	0.0723 (0.0082)	0.0938 (0.0066)	0.0944 (0.0132)	0.1220 (0.0242)
Cora	0.3401 (0.0587)	0.1200 (0.0333)	0.1854 (0.0144)	0.4198 (0.0199)	0.4241 (0.0105)	0.4150 (0.0321)	0.1020 (0.0357)	0.2439 (0.0691)
Citeseer	0.1541 (0.0161)	0.0396 (0.0098)	0.0671 (0.0128)	0.2204 (0.0211)	0.2366 (0.0086)	0.2249 (0.0187)	0.0838 (0.0132)	0.1420 (0.0412)

Table 3. PWF of clustering in real networks

	Deepwalk	LINE1	LINE2	Node2vec	GraRep	DNGR	M-NMF	GS-NMF
Karate	0.9399 (0.0000)	0.6800 (0.1126)	0.9651 (0.1256)	0.9883 (0.0353)	1.0000 (0.0000)	1.0000 (0.0000)	0.8513 (0.1670)	1.0000 (0.0000)
Football	0.8174 (0.0612)	0.7992 (0.0718)	0.8209 (0.0558)	0.8343 (0.0592)	0.8715 (0.0373)	0.8313 (0.0773)	0.8807 (0.0370)	0.8960 (0.0230)
Lesmis	0.6922 (0.0482)	0.6024 (0.0566)	0.6187 (0.0471)	0.7079 (0.0380)	0.6494 (0.0302)	0.6645 (0.0247)	0.6792 (0.0294)	0.7206 (0.0324)
Book	0.7644 (0.0374)	0.5797 (0.0648)	0.7595 (0.0330)	0.7974 (0.0216)	0.7594 (0.0000)	0.7308 (0.0362)	0.6518 (0.0758)	0.7602 (0.0207)
Adjnoun	0.5362 (0.0359)	0.5104 (0.0259)	0.6336 (0.0568)	0.5433 (0.0000)	0.7780 (0.0000)	0.5003 (0.0071)	0.6056 (0.0363)	0.7877 (0.0123)
Women	0.4924 (0.0392)	0.5308 (0.0537)	0.9570 (0.0990)	0.4769 (0.0071)	1.0000 (0.0000)	0.4866 (0.0130)	0.7519 (0.1740)	1.0000 (0.0000)
Cornell	0.3416 (0.0363)	0.3323 (0.0274)	0.3586 (0.0079)	0.3498 (0.0291)	0.3246 (0.0116)	0.3116 (0.0116)	0.3058 (0.0185)	0.3964 (0.0340)
Texas	0.4680 (0.0260)	0.4480 (0.0303)	0.5259 (0.0129)	0.5067 (0.0229)	0.4566 (0.0428)	0.4316 (0.0105)	0.4453 (0.0377)	0.5751 (0.0389)
Washington	0.4094 (0.0318)	0.4070 (0.0336)	0.5198 (0.0257)	0.4724 (0.0146)	0.3893 (0.0179)	0.3579 (0.0375)	0.4048 (0.0522)	0.5045 (0.0280)
Wisconsin	0.3922 (0.0305)	0.3801 (0.0336)	0.3606 (0.0163)	0.3599 (0.0319)	0.3293 (0.0165)	0.3336 (0.0241)	0.3701 (0.0227)	0.3941 (0.0494)
Cora	0.4129 (0.0466)	0.2861 (0.0175)	0.2555 (0.0102)	0.4565 (0.0251)	0.4619 (0.0163)	0.4220 (0.0412)	0.3024 (0.0091)	0.3400 (0.0542)
Citeseer	0.3050 (0.0111)	0.2679 (0.0156)	0.2199 (0.0168)	0.3378 (0.0119)	0.3292 (0.0059)	0.3157 (0.0132)	0.3002 (0.0014)	0.3065 (0.0169)

From Tables 2 and 3, we conclude that GS-NMF achieves competitive results on networks with community structure, even gains an advantage on most of the results. As for the bipartite network, GS-NMF achieves high accuracy on Women and reaches around 8% improvement on NMI of Adjnoun. For the dataset of WebKB, our method still shows the best performance on most of them. The results demonstrate the effectiveness of GS-NMF on networks with community structure, mix structure, especially bipartite structure.

6.4 Node Classification

We also evaluate the performance of the network embedding methods on node classification task. Since node classification is more meaningful on the networks with unclear structure, we select the third and the last group of networks to perform experiments. We set $m = 100$ for WebKB networks and $m = 500$ for Cora and Citeseer. We apply LIBLINEAR package to train classifier and perform ten-fold cross-validation on new representation. We report the mean and

standard deviation of PWF in Table 4. It can be noticed from Table 4 that our approach shows competitive results except Texas and Cora. The experiments further illustrate that GS-NMF has a certain ability to capture mixture structures and weak community structures in networks while effectively representing these networks.

Table 4. PWF of classification in real networks

	Deepwalk	LINE1	LINE2	Node2vec	GraRep	DNGR	M-NMF	GS-NMF
Cornell	0.3431 (0.0947)	0.2605 (0.0741)	0.3821 (0.0697)	0.4394 (0.0935)	0.4243 (0.1053)	0.3553 (0.0973)	0.4261 (0.0953)	0.4410 (0.1129)
Texas	0.4850 (0.1200)	0.3710 (0.0891)	0.5895 (0.1535)	0.5570 (0.1275)	0.6034 (0.1142)	0.5610 (0.2005)	0.5909 (0.1016)	(0.5948) (0.1219)
Washington	0.4291 (0.0991)	0.2703 (0.0645)	0.4384 (0.1038)	0.4716 (0.1073)	0.4667 (0.1061)	0.3916 (0.0591)	0.5059 (0.1079)	0.5585 (0.1095)
Wisconsin	0.3896 (0.0837)	0.2759 (0.0497)	0.3968 (0.0847)	0.4410 (0.1146)	0.4529 (0.0920)	0.4186 (0.0701)	0.4732 (0.0988)	0.4778 (0.0909)
Cora	0.6793 (0.0331)	0.6962 (0.0336)	0.5847 (0.0348)	0.7069 (0.0425)	0.6866 (0.0327)	0.7217 (0.0385)	0.6619 (0.0329)	0.7373 (0.0349)
Citeseer	0.4113 (0.0264)	0.4434 (0.0283)	0.3401 (0.0313)	0.5106 (0.0325)	0.3863 (0.0183)	0.4093 (0.0344)	0.4486 (0.0300)	0.4905 (0.0135)

6.5 Visualization

Another important application of network embedding is visualization, which visualize a network on two-dimensional space and help to evaluate the quality of embedding. The new representations of nodes are served as features to feed into t-SNE tool [16]. We mark the same color on the points belonging to the same class. Therefore, a good visualization result is more discriminative and easier to recognize different classes. Here we use Adjnoun network as example and show its results in Fig. 1. We only select several methods to display for the constraints of space. It is clear that the visualization from first two figures show unclear results and the points from different class mixed with each other. Obviously, the visualization of GS-NMF performs better in these methods.



Fig. 1. t-SNE 2D representations on Adjnoun.

6.6 Parameter Analysis

GS-NMF has two parameters: α and $\beta \in \{0.1, 0.5, 1, 5, 10\}$. We employ grid search and select a group of α and β with the best performance. Here we only adopt Lesmis as example because the effects of parameters in other networks show similar trends. We perform node clustering on new representation. The results are showed in Fig. 2. As we can see, the results with different parameters setting are relatively stable, which just vary within a certain range, even the worst results still reflect a good performance.

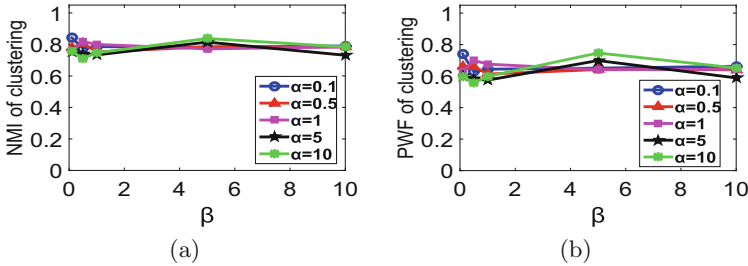


Fig. 2. The effect of α and β on Lesmis

7 Conclusion and Future Work

GS-NMF is a network embedding method which preserve network microscopic topology structure and diverse mesoscopic-general structures such as community structures, bipartite structures, etc. Based on nonnegative matrix factorization model, we define a joint objective function, which performs general structure discovery and network embedding simultaneously. Experimental results on node clustering, node classification and visualization verify the effectiveness of our model in representing networks with different structures. Usually, NMF requires high space and time complexities, This limits its the applicability on large-scale networks. Some existing works propose efficient updating rules to speed up the convergence [35]. Alternatively, as claimed in previous work [8], adding samples step by step instead of keeping entire dataset remained in memory during whole optimization is expected to reduce space occupancy. Giving s good strategy to alleviate the space and the time complexity will be our future concerns.

References

1. Borgatti, S.P., Everett, M.G.: Models of core/periphery structures. Soc. Netw. **21**(4), 375–395 (2000)
2. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. **3**(1), 1–122 (2010)

3. Cao, S., Lu, W., Xu, Q.: Grarep: learning graph representations with global structural information. In: Proceedings of the 24th ACM International Conference on Information and Knowledge Management, Melbourne, Australia, pp. 891–900 (2015)
4. Cao, S., Lu, W., Xu, Q.: Deep neural networks for learning graph representations. In: Proceedings of the 30th AAAI Conference on Artificial Intelligence, Arizona, USA, pp. 1145–1152 (2016)
5. Davis, A., Gardner, B.B., Gardner, M.R.: Deep south: a social anthropological study of caste and class. *Am. J. Sociol.* **48**(3), 432–433 (1941)
6. Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: scalable representation learning for heterogeneous networks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Nova Scotia, Canada, pp. 135–144 (2017)
7. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, California, USA, pp. 855–864 (2016)
8. Guan, N.Y., Tao, D.C., Luo, Z.G., Yuan, B.: Online nonnegative matrix factorization with robust stochastic approximation. *IEEE Trans. Neural Netw.* **23**(7), 1087–1099 (2012)
9. Hu, Y., Li, M., Zhang, P., Fan, Y., Di, Z.: Community detection by signaling on complex networks. *Phys. Rev. E* **78**(1), 016115 (2008)
10. Huang, X., Li, J., Hu, X.: Label informed attributed network embedding. In: Proceedings of the 10th ACM International Conference on Web Search and Data Mining, Cambridge, UK, pp. 731–739 (2017)
11. Knuth, D.E.: *The Stanford GraphBase: A Platform for Combinatorial Computing*. ACM (1993)
12. Lee, D.D.: Algorithms for nonnegative matrix factorization. *Adv. Neural Inf. Process. Syst.* **13**(6), 556–562 (2000)
13. Li, Y., Jia, C., Yu, J.: A parameter-free community detection method based on centrality and dispersion of nodes in complex networks. *Phys. A-Stat. Mech. Appl.* **438**, 321–334 (2015)
14. Lim, K.W., Buntine, W.L.: Bibliographic analysis with the citation network topic model. In: Asian Conference on Machine Learning, pp. 142–158 (2016)
15. McCallum, A.K., Nigam, K., Rennie, J., Seymore, K.: Automating the construction of internet portals with machine learning. *Inf. Retrieval* **3**(2), 127–163 (2000)
16. Maaten, L.V.D., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**, 2579–2605 (2008)
17. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E* **74**(3), 036104 (2006)
18. Newman, M.E.J.: Modularity and community structure in networks. *Proc. Natl. Acad. Sci.* **103**(23), 8577–8582 (2006)
19. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. *Phys. Rev. E* **69**(2), 026113 (2004)
20. Ou, M., Cui, P., Pei, J., Zhang, Z., Zhu, W.: Asymmetric transitivity preserving graph embedding. In: Proceedings of the 22nd SIGKDD International Conference on Knowledge Discovery and Data Mining, California, USA, pp. 1105–1114 (2016)
21. Pei, Y., Chakraborty, N., Sycara, K.: Nonnegative matrix tri-factorization with graph regularization for community detection in social networks. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, pp. 2083–2089 (2015)

22. Perozzi, B., Alrfou, R., Skiena, S.: Deepwalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, USA, pp. 701–710 (2014)
23. Roberts, L.G., Wessler, B.D.: Computer network development to achieve resource sharing. In: Proceedings of the Spring Joint Computer Conference, 5–7 May 1970, New Jersey, USA, pp. 543–549 (1970)
24. Shen, H., Cheng, X., Guo, J.: Exploring the structural regularities in networks. *Phys. Rev. E Stat. Nonlinear Soft Matter Phys.* **84**(5), 056111 (2011)
25. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: large-scale information network embedding. In: Proceedings of the 24th International Conference on World Wide Web, Florence, Italy, pp. 1067–1077 (2015)
26. Tu, C., Liu, H., Liu, Z., Sun, M.: Cane: context-aware network embedding for relation modeling. In: The 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada, vol. 1, pp. 1722–1731 (2017)
27. Tu, C., Zhang, Z., Liu, Z., Sun, M.: Translation-based network representation learning for social relation extraction. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence, Melbourne, Australia, pp. 2864–2870 (2017)
28. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, California, USA, pp. 1225–1234 (2016)
29. Wang, X., Cui, P., Wang, J., Pei, J., Zhu, W., Yang, S.: Community preserving network embedding. In: The AAAI Conference on Artificial Intelligence, California, USA, pp. 203–209 (2017)
30. Xuan, G., Shi, Y.Q., Chai, P., Sutthiwan, P.: An enhanced EM algorithm using maximum entropy distribution as initial condition. In: International Conference on Pattern Recognition, Tsukuba Science City, Japan, pp. 849–852 (2012)
31. Yang, C., Sun, M., Liu, Z., Tu, C.: Fast network embedding enhancement via high order proximity approximation. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI 2017, 19–25 August 2017, Melbourne, Australia, pp. 19–25 (2017)
32. Yang, C., Zhao, D., Chang, E.Y.: Network representation learning with rich text information. In: International Conference on Artificial Intelligence, Buenos Aires, Argentina, pp. 2111–2117 (2015)
33. Yang, T., Jin, R., Chi, Y., Zhu, S.: Combining link and content for community detection: a discriminative approach. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, pp. 927–936 (2009)
34. Zachary, W.W.: An information flow model for conflict and fission in small groups. *J. Anthropol. Res.* **33**(4), 452–473 (1977)
35. Zhang, D.K., Yin, J., Zhu, X.Q., Zhang, C.Q.: Collective classification via discriminative matrix factorization on sparsely labeled networks. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, Indianapolis, USA, pp. 1563–1572 (2016)



Intelligent Rub-Impact Fault Diagnosis Based on Genetic Algorithm-Based IMF Selection in Ensemble Empirical Mode Decomposition and Diverse Features Models

Manjurul Islam , Alexander Prosvirin, and Jong-Myon Kim 

School of Electrical and Computer Engineering, University of Ulsan, Ulsan, South Korea

m.m.manjurul@gmail.com, jongmyon.kim@gmail.com,
a.prosvirin@hotmail.com

Abstract. Rub-impact faults condition monitoring is a challenging problem due to the complexity in vibration signal of rub-impact faults. These complexities make it hard to use traditional time- and frequency-domain analysis. Recently, various time-frequency analysis approaches namely empirical mode decomposition (EMD) and ensemble EMD (EEMD) have been used for rubbing fault diagnosis. However, traditional EMD suffers from “mode-mixing” problems that cause difficulty to find physically meaningful intrinsic mode functions (IMF) for feature extraction. We propose an intelligent rub-impact fault diagnosis scheme using a genetic algorithm (GA)-based meaningful IMF selection technique for EEMD and diverse features extraction models. First, the acquired signal is adaptively decomposed into a series of IMFs by EEMD that correspond to different frequency bands of the original signal. Then, a GA search using a new fitness function, which combines the mean-peak ratio (MPR) of rub impact and mutual information (MI)-based similarity measure, is applied to select the meaningful IMF components. The designed fitness function ensures the selection of discriminative IMFs which carry the explicit information about rubbing faults. Those selected IMFs are utilized for extracting fault features, which are further employed with k -nearest neighbor (k -NN) classifier for fault diagnosis. The obtained results show that the proposed methodology efficiently selects discriminant signal-dominant IMFs, and the presented diverse feature models achieve high classification accuracy for rub-impact faults diagnosis.

Keywords: Empirical mode decomposition · Feature extraction
Genetic algorithm · Intelligent rub-impact fault diagnosis
Data-driven diagnostic

1 Introduction

Rubbing between the rotor and stationary components in rotating machines, such as wind turbines, pumps, and induction motors is a common phenomenon [1, 2]. The rubbing occurs whenever the rotor interacts with the stationary components due to faults, for instance, misalignment, self-excited vibrations, and shaft imbalance [2]. If

these faults are not detected in the initial stage, rubbing can cause severe malfunctions that could lead to a catastrophic downtime and costly maintenance. Thus, detecting onset rubbing faults and determining the intensity of the rub-impact fault are the critical issue in rotating machinery fault diagnosis.

Rubbing faults are highly non-stationary and nonlinear faults [1, 3], which cause a lot of impacts in the signal. Traditional signal processing methods based on time-domain and frequency-domain analysis are inept to capture transient phenomena of rubbing process due to inherent constraints in these techniques and the way they process a non-stationary and nonlinear signal with globally linear assumptions.

To overcome the non-linearity and non-stationary constraints in rubbing fault diagnosis, various time-frequency signal processing methods, such as short-time Fourier transform (STFT), wavelet transform (WT), and empirical mode decomposition (EMD), have been used in rubbing fault diagnosis [4, 5]. Huang et al. introduced another state-of-the-art time-frequency analysis technique, which is called empirical mode decomposition (EMD) [6].

However, the traditional EMD method has some limitations, which may cause problems while this method is applied for fault diagnosis purposes. Specifically, the abnormal presence of dissimilar oscillations in one IMF, or the presence of similar oscillations in multiple IMFs can be observed in conventional EMD. This phenomenon is called the “mode-mixing” problem that causes difficulty to obtain a clear physical interpretation of each IMF, which is necessary for fault diagnosis purposes. The variant of EMD is called ensemble EMD (EEMD), which successfully resolves the “mode-mixing” problem by applying EMD over an ensemble of the signal with the additive white noise [7]. Another critical issue in EEMD or EMD is to find the informative IMF set, which is noiseless, and contain explicit and informative information about the mechanical faults, such as rubbing phenomena in rotating machinery.

To address the above shortcomings in the existing EMD-based diagnosis methods, we propose an intelligent rub-impact fault diagnosis methodology using GA-based most meaningful IMFs selection procedure in EEMD and hybrid feature extraction process from selected components. As EEMD generates a finite set of IMFs, it is essential to decide which subset contains essential information about rubbing defect. In practices, all the IMFs components are not equally important for fault diagnosis, and these can be either signal dominant or noise dominant. Thus, it is essential to select the discriminative IMFs set, which is useful for fault features extraction and allow to perform the diagnosis of rubbing faults with high accuracy. To use GA [8] for selecting autonomously a set of signal-dominant and relevant to rubbing faults IMFs, a new fitness function which utilizes the ratio of the mean-peak ratio (MPR) of rub impact and mutual information (MI)—a statistical similarity metric [9]—is introduced in this study. Finally, various features [10] which are stemmed from time-domain, are extracted directly from the reconstructed signal of the selected IMFs, and frequency-domain features calculated from envelope power spectrum (EPS) [11] of this signal. Since this reconstrued signal is a clear rubbing fault signal, our diverse feature models distinctly represent rub-impact fault conditions, and this feature vector is further utilized with a state-of-art classifier— k -nearest neighbors (k -NN) [12]—to classify rubbing faults at various intensities. The efficacy of the proposed scheme is validated using data collected from benchmark rub-impact faults simulator.

The remaining sections are outlined as follows. An intelligent rub-impact fault diagnosis methodology in Sect. 2. Experiment results and relevant discussions are given in Sect. 3. Finally, Sect. 4 concludes this paper.

2 Proposed Methodology of Rub-Impact Fault Diagnosis

Figure 1 presents an intelligent rub-impact fault diagnosis scheme that used in this paper. As depicted in Fig. 1, the proposed methodology consists of three major steps: namely ensemble EMD (EEMD)-based signal preprocessing and GA-based IMF selection with an appropriate fitness function, feature extraction, and fault classification.

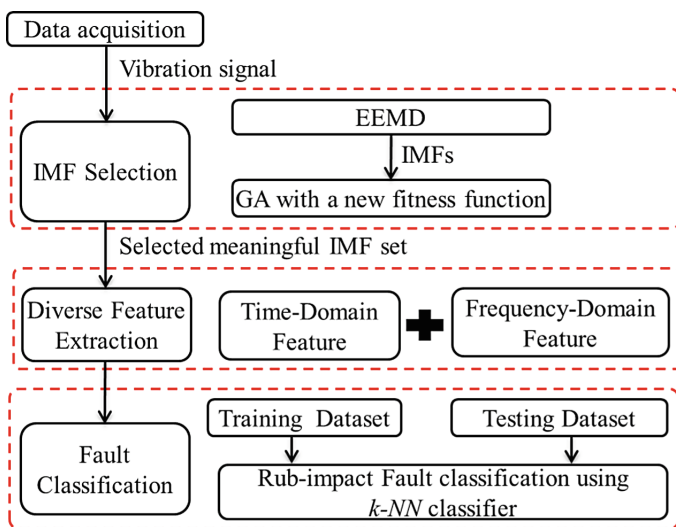


Fig. 1. An intelligent rub-impact fault diagnosis scheme including GA-based IMF section

2.1 Effectiveness of Ensemble Empirical Mode Decomposition (EEMD)

In this subsection, we describe the effectiveness of EEMD over traditional EMD. EEMD is noise assisted signal analysis method that solves the “mode mixing” problem by applying a uniformly distributed reference frame using the white Gaussian noise in the input signal [7, 8]. More ensemble in the average will improve confidence of EMD results; the only persistent part of the signal is the true and physically meaningful IMFs.

Figure 2 shows the result of extracted IMFs that obtained by the traditional EMD and EEMD for an original 2.8 g rub-impact signal. According to result in Fig. 2, it can be seen EMD fails to obtain sufficient numbers of IMFs, and IMF 9 and IMF 10 show a very similar oscillation. On the other hand, EEMD solves this mode-mixing problem where the IMFs containing the same frequency bands are extracted in different order for various groups of signals.

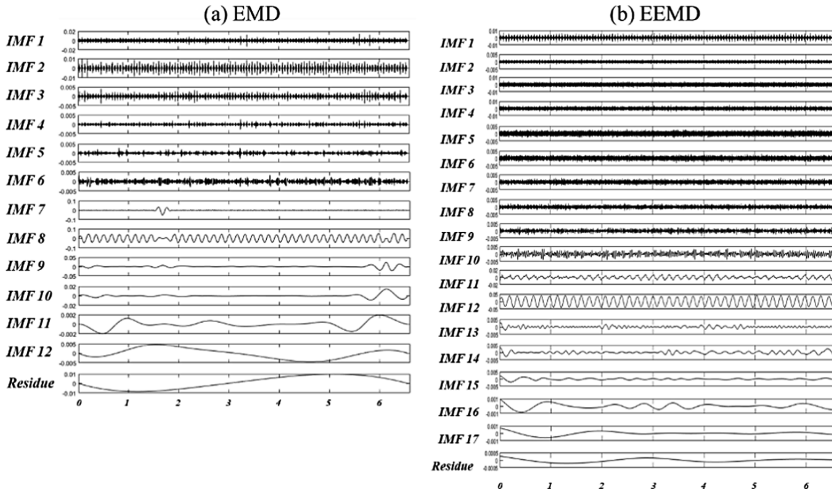


Fig. 2. Decomposed IMFs of a one-second original rubbing signal by (a) EMD and (b) EEMD.

2.2 Meaningful IMF Selection Using GA with a New Fitness Function

This paper applies GA for discriminant IMF subset selection. GA provides a good tradeoff between the quality of selected IMFs subset and the computational complexity [8]. The GA is used to produce a high-quality solution in optimization problems based on natural selection, which works with specific discrete steps namely problem representation (encoding), parent selection, crossover, mutation, and replacement.

In the proposed GA-based IMF selection, we generate initial population using the binary encoding scheme, and the length of each chromosome is equivalent to the number of initially extracted IMFs, and each chromosome represents a set of zeros and ones, where one is assigned for randomly selected oscillating components, and zeros are assigned to not selected IMFs. For instance, the chromosome of view 001000000100000000 means, which the 3rd and 10th IMFs are selected in the current solution. The uniform crossover and one-point mutation are chosen to reduce the change of separating the closely located genes in selected parent chromosome during the recombination process.

In this paper, we use a total of 2000 generations. In each generation, 50 offspring are created, and 50 chromosomes with the worst-valued fitness function in the population are replaced with those newly generated. These parameters are defined experimentally in which system performance is high. One of the significant tasks is to evaluate each subset with an appropriate fitness function in GA optimization for selecting the best IMF subset. To define a fitness, we carefully analyze the rubbing phenomena that include the ratio of the mean-peak ratio (MPR) of rub impact in each IMF to mutual information (MI), a statistical similarity metric, among IMFs.

MPR quantifies the degree of defective of rub-impact by taking the ratio between the sum of peak values of fundamental frequencies (FF) and its fractional, and the average power of the spectral values of the EPS. To ensure that MPR evaluation metric

can provide informative and most meaningful IMFs, the essential FF frequencies and their harmonics were defined based on various studies about rubbing processes and are as follows: $1/3X$, $1/2X$, $2/3X$, $1X$, $4/3X$, and $3/2X$. Once EPS can be calculated by [9], the proposed MPR evaluation metric is defined to precisely quantify the fundamental frequency (FF) and their harmonics in the EPS signal as follows,

$$MPR = 10 \log_{10} \left[\sum_{i=1}^h (FF_i - A_p) / A_p \right] (dB) \quad (1)$$

where n represents the number of FF (e.g., $n = 3$ in this study) and A_p defines the average power of the spectral components.

The next step is to reconstruct the signal using selected components as follows:

$$x_{rec}(t) = \sum_{l=1}^N IMF_l. \quad (2)$$

MPR is now ready, so our next step is to define MI since fitness function is the combined effect of both MPR and MI. Once, the reconstructed signal is obtained, we use the mutual information (MI) in this paper, which is commonly used information theoretic-based similarity measures between probability density functions (PDFs) [8, 9]. The main idea to use such a similarity measure is to find discriminant IMFs and penalize similar and noise-dominant IMFs. Let say, $x(t)$ and $x_{rec}(t)$ are two random variables, MI is calculated as follows [9],

$$MI_k(x(t); IMF_k(t)) = \iint p(x(t), IMF_k(t)) \log_2 \frac{p(x(t), IMF_k(t))}{p(x(t))p(IMF_k(t))}, \quad (3)$$

where $p(x(t))$ and $p(IMF_k(t))$ represent the marginal PDFs, and $p(x(t), IMF_k(t))$ are the joint PDF. Now MPR and MI are at hand; we define a fitness function in Eq. (4) that is used by GA to select the best combination of IMFs for each group of signals:

$$fitness = MPR_i / MI_i. \quad (4)$$

Note, this fitness function ensures that for whatever the number of IMFs included in solution is assigned, the GA will find an optimal or relatively optimal solution.

2.3 Diverse Feature Extraction Models

Another main idea is to extract diverse features to represent rub-impact fault uniquely that are used with k -NN classifier for higher accuracy for any rubbing fault diagnosis application. As GA-based IMF selection process generates a set of the most informative IMFs, this paper utilizes these selected IMFs for feature extraction. We extract most widely used features [10] that are stemmed from various signal processing domains such as time-domain and frequency-domain. Thus, we extract three time-domain features such as root mean square (RMS), kurtosis value (KV), and skewness value (SK).

Three frequency-domain, RMS frequency (RMSF), frequency standard deviation (FSD), mean frequency (MF) that are well corroborated to represent rub-impact uniquely.

3 Experiment Results and Discussion

To verify the effect of GA-based IMF selection for rubbing fault feature extraction and its application on fault diagnosis are presented in this section.

A self-designed machinery simulator is used to conduct research on rub-impact diagnosis, as can be seen in Fig. 3. To capture vibration data, most widely used two general purpose displacement sensors (model 3300 XL NSv) are installed in the drive end (DE) and non-drive end (NDE) respectively. In this study, the rub-impact fault is simulated as a shaft imbalance by adjusting extra weights. In total, ten different weights are added to the shaft, namely 0, 0.5, 1, 1.5, 1.6, 1.7, 1.8, 2, 2.4, and 2.8 grams (g). Data recording was performed at a constant speed with 2580 RPM, and data sampling rate was 56 Hz. A 59-s long signal is acquired for each weight. So, the created dataset contains 590 signal samples in total. The training subset consists of 30 signals, whereas the testing subset consists of 29 samples for each rubbing case.

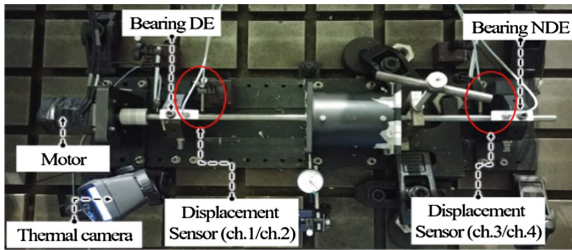


Fig. 3. A self-designed testbed for recording rub-impact vibration signals for fault diagnosis.

Table 1 summaries the result of selected valuable IMFs of each signal class based on GA with defined appropriate fitness function. To evaluate the efficacy of the proposed framework in classifying rub-impact faults of rotor blades with different intensity levels and confirm the advantages of the GA-based IMF selection process, we compared our approach with the state-of-the-art technique that utilizes time-varying and multiresolution envelope analysis for the optimal sub-band selection for further fault feature extraction [11] with degree-of-defectiveness ratio (DDR) metrics. This method

Table 1. The most meaningful IMFs subsets selected by GA

	Classes									
	0 g	0.5 g	1.0 g	1.5 g	1.6 g	1.7 g	1.8 g	2.0 g	2.4 g	2.8 g
Selected IMFs	7, 8, 10	2,7,8	3,7,10	8,9,10	3,9,10	3,7, 9	3,7	3,7,10	3,7	2,7,10

will be referred to as TVMR+DDR. Another approach for the comparison is a diverse feature model directly applied to the raw vibration signal containing rubbing faults (is referred to as RAW+DFEAT). The diagnosis accuracy is computed through the true positive rate evaluation (TPR) and classification accuracy (CA) as in [11].

Table 2. Experimental results. STD, standard deviation; average classification accuracy (ACA)

Method	Average TPR (STD) (%)										ACA (STD)
	0 g	0.5 g	1 g	1.5 g	1.6 g	1.7 g	1.8 g	2.0 g	2.4 g	2.8 g	
Proposed	98.28 (0.9)	99.31 (3.0)	97.6 (2.2)	97.93 (1.2)	92.41 (3.0)	95 (2.8)	96.38 (2.8)	87.07 (2.0)	93.45 (2.3)	92.59 (2.9)	94.75 (1.52)
TVMR+DDR	94.31 (3.0)	88.45 (5.9)	98.97 (3.3)	76.03 (2.6)	91.72 (5.8)	83.79 (3.7)	94.66 (4.5)	96.72 (4.5)	85.69 (8.6)	91.03 (7.4)	90.13 (6.10)
RAW+FEAT	46.90 (7.2)	88.45 (11.9)	64.14 (12.1)	92.41 (11.4)	91.72 (9.1)	67.59 (11.8)	83.28 (10.8)	86.55 (12.0)	80.86 (5.6)	91.03 (6.9)	79.29 (8.15)

Table 2 presents the experimental results, in which the proposed rub-impact fault diagnosis method using EEMD and GA-based IMF selection outperforms the referenced methods in terms of the average CA with the value of 94.75% achieved over 20 experiments. An interesting observation is that the average TPR values of the proposed method are above the value of 87%, and the standard deviations (STD) of TPR values for the proposed method do not exceed 3%, whereas referenced models suffer from as low as 76.03% with TVMR+DDR and 64.14% with RAW+DFEAT.

In addition, Fig. 4 presents the confusion matrices of the proposed and referenced methods. According to results in Fig. 4, the proposed method in Fig. 4(c) correctly identifies all fault types with a very low miss-classification rate in comparison with its counterparts, RAW+DFEAT in Fig. 4(a) and TVMR+DDR in Fig. 4(b).

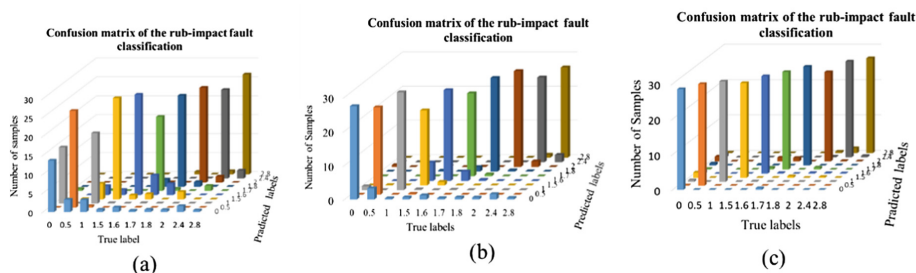


Fig. 4. The confusion matrices for (a) RAW+DFEAT, (b) TVMR+DDR, and (c) Proposed

4 Conclusions

This paper proposed an intelligent rub-impact fault diagnosis algorithm based on GA-based IMF selection in EEMD and heterogonous feature extraction models. The proposed algorithm addresses the problem of selecting proper and meaningful IMFs for rub-impact fault diagnosis. First, EEMD was applied to generate well-behaved and well-separated IMFs and to lessen the effect of the “mode-mixing” problem of traditional EMD algorithm. Second, the meaningful signal-dominant IMFs were selected using GA with the developed fitness function employing the new MPR evaluation method and MI distance metric. Selected IMFs were utilized for feature extraction that is further applied with a k-NN classifier for diagnosis. The results demonstrated that the proposed approach outperforms its referenced methods regarding TPR and classification accuracy.

Acknowledgements. This work was supported by the Korea Institute of Energy Technology Evaluation and Planning (KETEP) and the Ministry of Trade, Industry & Energy (MOTIE) of the Republic of Korea (Nos. 20181510102160, 20162220100050, 20161120100350, 20172510102130). It was also funded in part by The Leading Human Resource Training Program of Regional Neo-Industry through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and future Planning (NRF-2016H1D5A1910564), and in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2016R1D1A3B03931927).

References

1. Cong, F., Chen, J., Dong, G., Huang, K.: Experimental validation of impact energy model for the rub–impact assessment in a rotor system. *Mech. Syst. Signal Process.* **25**, 2549–2558 (2011)
2. Zhang, Y., Wen, B., Leung, A.Y.: Reliability analysis for rotor rubbing. *J. Vib. Acoust.* **124**, 58–62 (2002)
3. Rubio, E., Jáuregui, J.C.: Time-frequency analysis for rotor-rubbing diagnosis. In: *Advances in Vibration Analysis Research*. InTech (2011)
4. Lu, Y., Meng, F., Li, Y.: Research on rub impact fault diagnosis method of rotating machinery based on wavelet packet and support vector machine. In: *International Conference on Measuring Technology and Mechatronics Automation, ICMTMA 2009*, pp. 707–710 (2009)
5. Deng, L., Zhao, R.: Fault feature extraction of a rotor system based on local mean decomposition and Teager energy kurtosis. *J. Mech. Sci. Technol.* **28**, 1161–1169 (2014)
6. Huang, N.E., et al.: The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, pp. 903–995. The Royal Society (1998)
7. Wu, Z., Huang, N.E.: Ensemble empirical mode decomposition: a noise-assisted data analysis method. *Adv. Adapt. Data Anal.* **1**, 1–41 (2009)
8. Nguyen, P., Kim, J.-M.: Adaptive ECG denoising using genetic algorithm-based thresholding and ensemble empirical mode decomposition. *Inf. Sci.* **373**, 499–511 (2016)

9. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**, 1226–1238 (2005)
10. Rauber, T.W., de Assis Boldt, F., Varejão, F.M.: Heterogeneous feature models and feature selection applied to bearing fault diagnosis. *IEEE Trans. Ind. Electron.* **62**, 637–646 (2015)
11. Kang, M., Kim, J., Wills, L.M., Kim, J.M.: Time-varying and multiresolution envelope analysis and discriminative feature analysis for bearing fault diagnosis. *IEEE Trans. Ind. Electron.* **62**, 7749–7761 (2015)
12. Kang, M., Islam, M.R., Kim, J., Kim, J.M., Pecht, M.: A hybrid feature selection scheme for reducing diagnostic performance deterioration caused by outliers in data-driven diagnostics. *IEEE Trans. Ind. Electron.* **63**, 3299–3310 (2016)



Anomaly Detection in Spatial Layer Models of Autonomous Agents

Marie Kiermeier^(✉), Sebastian Feld, Thomy Phan,
and Claudia Linnhoff-Popien

LMU Munich, Munich, Germany
marie.kiermeier@ifi.lmu.de

Abstract. For describing the complete state of complex environments with multiple mobile and autonomous agents, spatial layer models (SLM) are popular data structures. These models consist of several planes describing the spatial structure of selected features. Those SLM can directly be used for deep reinforcement learning tasks. However, detecting anomalies in such SLM poses two major challenges: the state space explosion in such settings and the spatial relations between the features. In this paper, we present a method for anomaly detection in SLM which solves both challenges by first extracting significant sub-patterns from training data and storing them in a dictionary. Afterwards, the entries of this dictionary are used for reconstructing SLM, which have to be validated. The resulting covering rate is an indicator for the (ab)normality of the given SLM. We show the applicability of our approach for a simple multi-agent scenario, and more complex smart factory scenarios with autonomous agents.

Keywords: Anomaly detection · Autonomous agents · Spatial data

1 Introduction

Spatial layer model (SLM) are commonly used to describe the current over-all state of a single/multi-agent setting, e.g., the agents' positions, and the environment they are interacting with [2, 6, 8]. For doing this, features are defined and spatially represented in relation to the surface area. A set of such feature planes is similar to a multi-channel image. The resulting SLM are then used as input for convolutional neural networks (CNN) for deep reinforcement learning tasks [5]. The advantage of this state representation is that it preserves the spatial relations between the features (e.g., self/all/opponent agent, obstacles, etc.). However, new challenges for anomaly detection arise: First, since the agents can move almost freely within their environment, the state space becomes intractable, which is known as *state space explosion*. As a consequence, classical anomaly detection approaches are overstrained. Instead, a method is required, which first reduces the state space, and then enables reliable anomaly detection within this reduced state space. Second, since SLM are used for state description

while *preserving spatial relations* between different features, these spatial relations need to be retained by the anomaly detection approach. Accordingly, we present in this paper an anomaly detection approach for SLM, which is able to handle both challenges. For doing this, we will transfer the approach presented in [4], which addresses the challenge of state explosion for trajectories (1D lines) to SLM (2D surface areas). We use two multi-agent settings to show that suitable sub-pattern dictionaries can be built to allow the validation of SLM before inputting them into CNN.

2 Related Work

Pimentel et al. provide in [7] an exhaustive review of existing novelty detection methods, which can be used to detect data deviating significantly from the data used for training. In general, every anomaly detection approach can be classified according to the technical strategy (probabilistic, distance-based, etc.), and the data to which it can be applied (high-dimensional data, time-series, images, etc.). However, to the best of our knowledge, there exists no approach in the literature so far, which handles both challenges – state space explosion and spatial data –, which arise in settings with mobile and autonomous agents. Kiermeier et al. address in [4] the state space explosion problem for trajectory data. In contrast to SLM, trajectories are 1D lines. For 2D surface areas of SLM, however, the approach of [4] is not directly applicable, but has to be modified.

3 Anomaly Detection in Spatial Layer Models

In this section, the basic concepts and methodologies for anomaly detection in SLM are introduced.

3.1 Spatial Layer Model

In [2, 6, 8] the idea of using SLM to represent the state of a multi-agent systems is introduced. By using this representation spatial relations between features are preserved. A simple example for such a SLM is visualized in Fig. 1a. The general structure for the considered simple multi-agent setting is to mark obstacles in the undermost plane, which have to be circumvented by the agents. For each agent in the system a further plane is set on top for recording its current position. In the concrete case three agents (feature planes 2 – 4) are moving in an environment. The environment is represented by a 4x4 grid and contains one obstacle (feature plane 1). In this case *one-hot encoding* is used, i.e., each grid cell is either 1 or 0. Accordingly, positions of obstacles or agents are described by setting the value of the corresponding grid cell to 1 (marked in red in Fig. 1a). In contrast, a possible SLM in a smart factory scenario, where the agents have to work-off a task list by moving to the corresponding machines, could be more complex. Here, features are the positions and types of machines, the agents' states, and

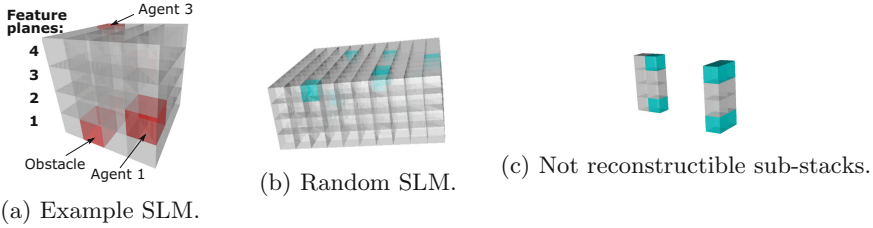


Fig. 1. Example for a SLM and a result of the anomaly detection procedure. (Color figure online)

for each machine the positions (and numbers) of agents inquiring it (see [6] for more details). In this scenario, the feature values are not limited to $\{0, 1\}$, but to \mathbb{N} . The machine inquiries or agents’ states are aggregated. This means, that if a machine is inquired by four agents, for example, the corresponding grid cell is set to 4. In this case, the state space increases even more, since the number of possible states increases.

3.2 Anomaly Detection Using Sub-Pattern Dictionary

The idea of the anomaly detection method presented in [4] is to model the huge state space of an autonomous agent setting by combining sub-patterns. Therefore, significant sub-patterns are extracted from training data and stored in a dictionary. The (ab)normality of data can then be assessed by the degree of which the data can be reconstructed by the dictionary entries. In case of trajectory data the sub-paths can be extracted using *pathlet learning* [1]. The idea is to find the “optimal” decomposition for each trajectory such that we get a compact sub-path dictionary, which can represent all training data with as few entries as possible. It has to be noted, that for 1D lines (trajectories), it is possible to check all possible decompositions to find the “optimal” one. However, this is infeasible for 2D surface areas of SLM. Accordingly, in the following we propose to use *recursive coordinate bisection* (RCB) to decompose the spatial area (x-/y-direction) of the SLM. RCB is a graph partitioning heuristic for recursively dividing a domain into sub-domains [9]. In each recursion step the domain is subdivided into two sub-domains by splitting the dimension with the longest expansion. We will use this division procedure to approximate the “optimal” decomposition of a SLM. To preserve the spatial relations, the SLM are only partitioned regarding their spatial area (x-/y-direction), but not horizontally (z-direction). The resulting partitions are candidate sub-patterns for the dictionary. The advantage of RCB is that the division procedure can be represented by a binary tree. For each recursion step, two children are added, which correspond to the two sub-areas. To find the “optimal” decomposition, weights $f(p)$ are added to the tree for every partition. These weights are defined by a recursive formulation of the original lower bound for pathlet learning [1]:

$$f(p) = \begin{cases} \min\left(\frac{1}{|D(p)|}, \lambda + f(p_1) + f(p_2)\right), & \text{if } p = p_1 \cup p_2 \\ \frac{1}{|D(p)|}, & \text{otherwise} \end{cases} \quad (1)$$

where p_1 and p_2 are the children of pattern p , $D(p)$ the set of training samples which contain p . Based on this, nodes/partitions are aggregated from bottom to top, if $\lambda + f(p_1) + f(p_2) > f(p)$. λ can be seen as a penalty, which assures, that we only decompose, if the two sub-partitions are clearly used more often, thus being more suited as dictionary entries. Obviously, using RCB is quite restrictive: It limits the set of candidates for the sub-patterns to those resulting from bisecting the spatial area. Other possibly more meaningful sub-patterns are left out. However, as we will show in Sect. 4, this partitioning heuristic suffices for building adequate sub-pattern dictionaries. The advantage of RCB, being fast and simple, outweighs this restriction clearly in this case.

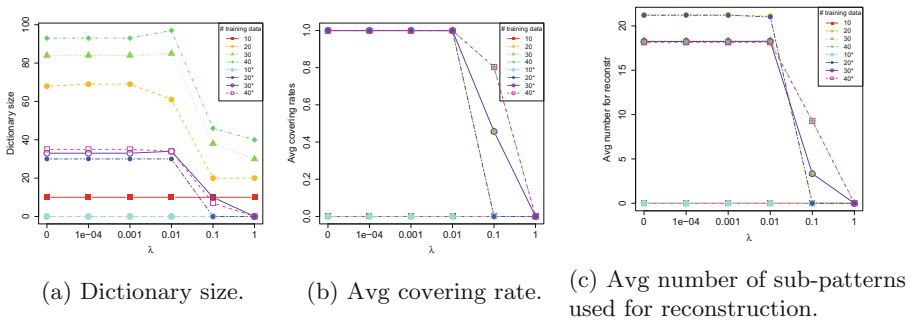


Fig. 2. Evaluation results for the simple multi-agent scenario.

3.3 Dictionary Optimization

We also propose an additional optimization step, which eliminates entries, which are not absolutely required. For this, we reconstruct a set of test data, whereby entries covering a large area are preferred for reconstruction. This way, entries can be identified which are subsets of other ones, and therefore being obsolete. In Sect. 4, we show in detail, that this optimization step reduces the dictionary size once again, but without remarkable loss in quality. In addition, by preferring entries covering a large area the reconstruction step becomes more efficient, since the number of required sub-patterns decreases.

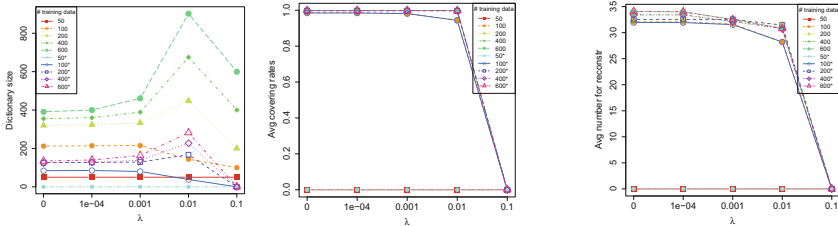
4 Evaluation

In this section we evaluate our approach for anomaly detection in SLM. We will first have a look at a simple multi-agent scenario, which uses one-hot encoding and then apply our approach to more complex smart factory scenarios. In each case, the data is generated by simulation models, which generate valid states according to the specifications.

4.1 Simple Multi-agent Scenario

Following the setting of [2] we have generated SLM for a simple multi-agent scenario. The environment is a 10×10 grid with two obstacles and three mobile and autonomous agents. Agents can occupy the same grid cell, while grid cells with obstacles cannot be occupied. An example for an invalid state, which should be detected by the anomaly detection process, would be, if an agent is at an obstacle’s position. For that, we need a suitable dictionary of sub-patterns. In Fig. 2 relevant performance indicators for the simple multi-agent scenario are plotted (results of the optimized sub-pattern dictionary are denoted with (*)). Overall, there are two parameters, which have to be adjusted correctly according to the scenario specifications: the amount of training data, and the trade-off-parameter λ . For various combinations of them, the size of the resulting dictionary, the average covering rate when reconstructing 1000 test states, and the average number of sub-patterns used for reconstructing a given SLM are shown. Based on this, one can see, that for the given scenario, 20 training states are sufficient to get an adequate average covering rate for the test data (0.99951). Less training data does not produce a meaningful dictionary; more training data does not achieve any significant increase in covering rate (see Fig. 2b), but requires more computation. Regarding λ , in this case $\lambda = 0.01$ is a good compromise between efficiency (see Fig. 2c), and a compact dictionary (see Fig. 2a) while having good covering rates (see Fig. 2b). In case of $\lambda > 0.1$ the average covering rate drops to 0 independently of the amount of training data, as $\lambda + f(p_1) + f(p_2)$ (see Eq. 1) becomes too large. Consequently, the nodes of the RCB tree are always completely aggregated when going from bottom to top for decomposition. Accordingly, the resulting dictionary consists of all (not decomposed) training states, but no useful sub-patterns for reconstruction. For this scenario the optimization step (see Sect. 3.3) reduces the dictionary size from 61 to 30 entries, when using 20 data samples for training and setting $\lambda = 0.01$. Similar holds when using more training data (see Fig. 2a). On average, the dictionary size is reduced by about 40%, while the average covering rates stay nearly constant (see Fig. 2b). This shows, that the proposed optimization step can be used for making the anomaly detection procedure more efficient without significant loss in quality. To illustrate the identification of invalid SLM, we generated one-hot encoding random states. In Figs. 1b and c one example state and the corresponding result of the anomaly detection process are visualized. Overall, two anomalous sub-stacks are identified (see Fig. 1c). The sub-stacks are invalid, since the third agent (feature plane 4) is at an obstacle’s position (feature plane 1). It is also remarkable, that for reconstructing a given SLM from the test data, 18-22 sub-patterns are required on average (see Fig. 2c). This means that (*dict. size * # of replcmt*) = $30 * 20 = 600$ checks are required for reconstruction in the worst case. In contrast, taking the naive way and checking every feature vector individually would result in $8 * 100 = 800$ checks, as there are 8 valid feature configurations in total. Accordingly, by having spatially connected groups of feature vectors instead of single feature vectors, the worst case complexity of the reconstruction/validation step can be reduced. Besides the reconstructability, the occurrence of dictionary

entries for the reconstruction can be observed. With that, one can assure that there is, for example, always just one 1-entry at each agent’s position feature plane. As soon as a dictionary entry, which describes an agent’s position, is used more than once for reconstruction, the anomaly detection procedure will report this. For more details see [3], where *sub-pattern (pathlet) occurrence* is introduced as monitoring feature when dealing with sub-pattern dictionaries for anomaly detection.



(a) Dictionary size (scenario 1). (b) Avg covering rate (scenario 1). (c) Avg number of sub-patterns used for reconstruction (scenario 1).

			1e-04	0.001	0.01				1e-04	0.001	0.01				1e-04	0.001	0.01	
100	0.9862	0.9827	200	0.9865	0.9855	0.9684	2000	0.9763	0.9759	0.9694	100	0.9961	0.9956	0.9843	100*	0.9960	0.9955	0.9843
200	0.9973	0.9973	400	0.9926	0.9924	0.9915	4000	0.9916	0.9914	0.9890	200*	0.9994	0.9994	0.9993	400*	0.9994	0.9994	0.9993
200*	0.9969	0.9969	400*	0.9924	0.9922	0.9914	4000*	0.9911	0.9910	0.9886	100*	0.9994	0.9994	0.9993	200	0.9994	0.9994	0.9993
400	0.9987	0.9987																

(d) Scenario 1. (e) Scenario 2. (f) Scenario 3. (g) Scenario 4.

Fig. 3. Evaluation results for smart factory scenarios.

4.2 Smart Factory Scenarios

In this section, we want to show the applicability of our approach to smart factory scenarios, which are complex multi-agent settings. The structure of SLM describing the states of a smart factory is adapted from [6]. Overall, we evaluated our approach on 4 different scenarios. The basic scenario is a manufacturing plant with a surface area of 10×10 grids, where 5 machines are placed, and 8 agents are moving around working-off their tasks. Similar to the simple multi-agent scenario in Sect. 4.1, in Fig. 3a–c the relevant performance indicators are plotted, while in Fig. 3d the covering rates are given in detail. It can be seen, that using 200 training data and setting $\lambda = 0.001$ results in a good trade-off between dictionary size (see Fig. 3a) and efficiency (see Fig. 3c), while having high covering rates anyway (see Fig. 3d). Again, the optimized dictionary variant can be used without significant loss in quality. For the second scenario the number of agents is doubled from 8 to 16. As shown in Fig. 3e, the number of training

data has to be increased from 200 to 400 to get similar covering rates (see middle rows of Fig. 3e). This is because more sub-patterns are required now. For example, the range of the feature values specifying the machine inquiries may expand, since more agents in the smart factory are able to queue up at the machines. In this case, the difference between SLM using one-hot encoding and those without comes into effect. For different inquiry feature values different sub-patterns have to be stored in the dictionary. In other words, the sub-pattern dictionary is adjusted to specific scenarios – having also high machine inquiry rates, or only less. As a consequence, a dictionary trained for a scenario with low inquiry values, would alarm if suddenly much more agents are within the smart factory, inquiring more machines, or if one machine breaks down, and the corresponding inquiries dam. In this case, λ should again be set to 0.001 to achieve a good compromise between dictionary size (see upper rows of Fig. 3e) and efficient reconstruction (see bottom rows Fig. 3e). In the third scenario, the number of machines is doubled from 5 to 10. As shown in Fig. 3f, in this case, the number of training states has to be increased significantly from 200 to 4000 (see upper rows of Fig. 3f), while λ can be left to 0.001. This change regarding the amount of training data is because of the increasing number of the feature planes from 10 to 15 due to doubling the number of machines. With this, the state space grows. Accordingly, the amount of training data has to be increased, too, to receive adequate covering rates for the test data (see middle of Fig. 3f). In the fourth scenario the area of the smart factory is quadrupled from $10 \times 10 = 100$ to $20 \times 20 = 400$. In this case, the number of training data can be even reduced from 200 to 100 while reaching similar covering rates for the test data (see middle of Fig. 3g). λ should, again, be set to 0.001, as already done in all previous scenarios. In this scenario it is remarkable, that the dictionary size and the average number of sub-patterns used for reconstruction increases only marginally compared to the basic scenario, in spite of the quadrupled area (see upper and lower rows of Fig. 3g). This indicates, that the sub-patterns have grown, too, and now cover more area. This is possible, since the number of machines and agents in the smart factory remain constant, while only the spaces between them become larger. Accordingly, larger sub-patterns are learned, and the number of entries required for reconstruction stays nearly constant. From an efficiency perspective, this is an enormous advantage, since the number of sub-patterns does not increase linearly with the area of the SLM. In addition, one can see, that independent of the concrete scenario structure the proposed optimization step clearly reduces the dictionary size, without meaningful loss in quality. On average, the number of entries is again reduced by about 40%, while the average covering rate is affected only marginally. Accordingly, this optimization step can also be used in more complex scenarios for further efficiency improvements within the whole anomaly detection procedure. Overall, the experiments show that our approach is also applicable to more complex multi-agent settings, like smart factories. For each of the four scenarios suitable sub-pattern dictionaries can be built to detect invalid SLM. Only the amount of training data and the λ parameter have to be

adjusted to the given scenario to get a compact, but efficient dictionary, which also yields high covering rates.

5 Conclusion and Future Work

In this paper, we have presented an anomaly detection method for SLM. An existing method for 1D lines (trajectories) is transferred to 2D surface areas of the SLM. For that purpose, a RCB procedure is used for building a binary tree, which recursively subdivides the surface area. After adding weights to each partition, this binary tree is then used to find the “optimal” decomposition for each training data sample individually. The resulting set of sub-patterns can be used to evaluate SLM by checking their reconstructibility. We have applied our approach to a simple multi-agent scenario, and more complex smart factory scenarios with even greater state spaces. Our evaluation has shown, that for each scenario compact, but efficient sub-pattern dictionaries for anomaly detection can be built after having adjusted only two parameters accordingly.

Regarding further work, we think about additional pruning strategies to make the reconstruction step more efficient. Another interesting aspect for further work are update strategies for the sub-pattern dictionary in case of changing scenario settings.

References

1. Chen, C., Su, H., Huang, Q., Zhang, L., Guibas, L.: Pathlet learning for compressing and planning trajectories. In: Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 392–395. ACM (2013)
2. Gupta, J.K., Egorov, M., Kochenderfer, M.: Cooperative multi-agent control using deep reinforcement learning. In: Sukthankar, G., Rodriguez-Aguilar, J.A. (eds.) AAMAS 2017. LNCS (LNAI), vol. 10642, pp. 66–83. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71682-4_5
3. Kiermeier, M., Sauer, H., Wieghardt, J.: Monitoring self-organizing industrial systems using sub-trajectory dictionaries. In: IEEE 15th International Conference on Industrial Informatics (INDIN), pp. 665–670. IEEE (2017)
4. Kiermeier, M., Werner, M., Linnhoff-Popien, C., Sauer, H., Wieghardt, J.: Anomaly detection in self-organizing industrial systems using pathlets. In: IEEE International Conference on Industrial Technology (ICIT), pp. 1226–1231. IEEE (2017)
5. Mnih, V., et al.: Playing atari with deep reinforcement learning. arXiv preprint [arXiv:1312.5602](https://arxiv.org/abs/1312.5602) (2013)
6. Phan, T., Belzner, L., Gabor, T., Schmid, K.: Leveraging statistical multi-agent online planning with emergent value function approximation. In: Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems, pp. 730–738 (2018)
7. Pimentel, M.A., Clifton, D.A., Clifton, L., Tarassenko, L.: A review of novelty detection. *Signal Process.* **99**, 215–249 (2014)
8. Silver, D., et al.: Mastering the game of go with deep neural networks and tree search. *Nature* **529**(7587), 484–489 (2016)
9. Simon, H.D.: Partitioning of unstructured problems for parallel processing. *Comput. Syst. Eng.* **2**(2–3), 135–148 (1991)



Deep Learning-Based Approach for the Semantic Segmentation of Bright Retinal Damage

Cristiana Silva¹, Adrián Colomer²(✉), and Valery Naranjo²

¹ Campus Gualtar, University of Minho, 4710 Braga, Portugal

² Instituto de Investigación e Innovación en Bioingeniería (I3B),
Universitat Politècnica de València,
Camino de Vera s/n, 46022 Valencia, Spain
adcogra@i3b.upv.es

Abstract. Regular screening for the development of diabetic retinopathy is imperative for an early diagnosis and a timely treatment, thus preventing further progression of the disease. The conventional screening techniques based on manual observation by qualified physicians can be very time consuming and prone to error. In this paper, a novel automated screening model based on deep learning for the semantic segmentation of exudates in color fundus images is proposed with the implementation of an end-to-end convolutional neural network built upon U-Net architecture. This encoder-decoder network is characterized by the combination of a contracting path and a symmetrical expansive path to obtain precise localization with the use of context information. The proposed method was validated on E-OPHTHA and DIARETDB1 public databases achieving promising results compared to current state-of-the-art methods.

Keywords: Semantic segmentation · Deep learning · Fundus images
Exudates · U-Net

1 Introduction

According to the World Health Organization (WHO), diabetic retinopathy (DR), a complication of diabetes manifested in the retina, is a major cause of blindness within the working age population in the developed world. It occurs as a result of accumulated damage to the retinal small blood vessels [1]. Due to a common absence of symptoms in early stages, this disease can go unnoticed until the changes in the retina have progressed to a level where treatment is nearly impossible or irreversible vision loss has occurred.

The risk of blindness in diabetic patients could be significantly reduced through regular screening by suitably trained observers for the development of DR, since an early detection and timely treatment can halt or reverse the

progression of the disease [2]. However, the number of qualified physicians available for direct examinations of the population at risk is limited in most countries. Moreover, the conventional retina examination techniques based on manual observation can be highly subjective, very time consuming and prone to error. These facts highlight the need for automated DR diagnosis techniques based on color fundus retinal photography with high accuracy and quick convergence rate for them to be suitable for real-time applications.

One of the primary signs of DR is the development of retinal exudates (Fig. 1(a)) which consist of lipid and protein accumulations in the retina of various shapes, locations, and sizes, according to the stage of the disease. Its accurate detection can be seriously affected by several factors related to the acquisition process with fundus cameras as well as retina's anatomy. The presence of other bright elements (drusen, optic disk, and optic nerve fibers), dust spots, random brightness, noise presence, uneven illumination, low contrast, and color variation represent a challenge for the task at hand, as it can be seen in Fig. 1(b), making this an extensively studied topic.

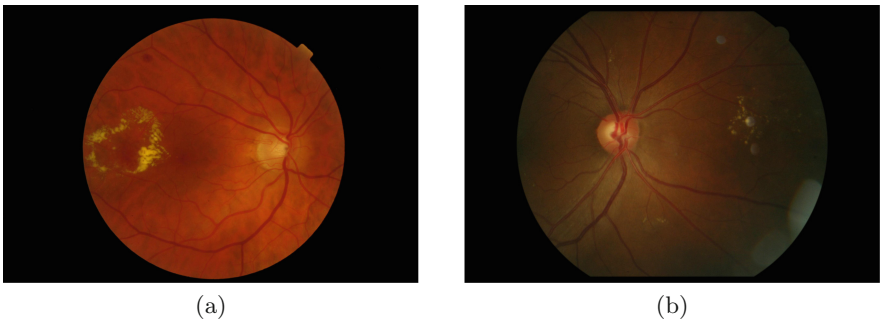


Fig. 1. Fundus images. (a) Image with the presence of exudates, (b) Noisy image with artefacts and uneven illumination.

Most of the classical approaches developed so far involve an image preprocessing stage, followed by a candidate extraction step where structures with similar characteristics as the lesion are selected. Finally, several features are extracted for each lesion candidate and a classification algorithm (usually, a machine learning classifier [3]) is applied to eliminate false positives. The main implemented techniques for extracting lesion candidates can be categorized into dynamic thresholding [4], mathematical morphology [5–7], and clustering [8]. A hand-crafted feature extraction requires domain expertise and effort for it to be optimized to specific problems. This process is deemed unobjective since the researcher has to manually decide on the features to be used in a classifier with knowledge obtained through specialized clinicians. This motivates the development of a novel framework capable of automatically learn the most relevant features and accurately segment exudates.

Recently, convolutional neural networks (CNNs), a branch of deep learning, have emerged as a powerful tool for making automatic image recognition tasks more successful. Its great performance in biomedical applications [9, 10] can be explained through its capability of hierarchically extract features from raw image pixel intensities by learning and formulating the appropriate filters for the task at hand. The first attempts in CNN-based approaches for DR evaluation emerged in a Kaggle¹ competition [11, 12] where images were classified by the severity of the disease. To the best of the author’s knowledge, only a single work has been developed towards the segmentation of exudates by using deep neural networks. In [13] a patch-based CNN architecture is proposed with the aim of providing a pixel-wise classification by returning the probability of each pixel belonging to one of two classes: exudate or non-exudate. Two fully-connected layers are responsible for the binary classification of image pixels. The resulting map is then combined with the output of optic disc and vessel detection procedures. Despite the fact that the network’s input includes optic disk pixels, potentially affecting its results, this architecture is not best suited for a pixel-level classification.

The main contribution of this paper is a novel deep learning-based approach for the automatic semantic segmentation of exudates in color fundus images. An encoder-decoder CNN built on top of the U-Net architecture [14] is implemented for this application. The model uses labelled pixels to learn the connection between local features and the associated specific classes, and then classify each pixel based on which class presents the highest probability for that pixel. Compared to most exudate segmentation methods, the algorithm undertaken in this work is more robust due to the fact that it is end-to-end, in other words, it is almost entirely trainable and free of hand designed and fixed modules. To the best of the author’s knowledge, this is the first attempt of adapting an encoder-decoder CNN architecture to retinal images’ semantic segmentation.

2 Methods

CNNs have been successfully applied to semantic segmentation [15], specifically fully convolutional networks (FCNs) which follow the encoder-decoder architecture. This image-to-image structure emerged as a solution for pixel-wise predictions since it outputs high resolution segmentation maps with localization as well as semantics information, performing very well in biomedical image segmentations [16, 17]. Following these nets, a novel neural network structure was introduced in [14], the so-called “U-Net” for its U-shaped architecture. It differs from FCNs in its extended decoding branch by taking into account useful global context information in higher resolution layers, being able to work with small sets of training images and still provide more precise segmentations. It has proven its effectiveness in biomedical image segmentations [14, 18, 19] as it outperforms existing methods on biomedical challenges. In this paper, a CNN is built on top of U-Net for the semantic segmentation of retinal images, as shown in Fig. 2.

¹ <https://www.kaggle.com/c/diabetic-retinopathy-detection>.

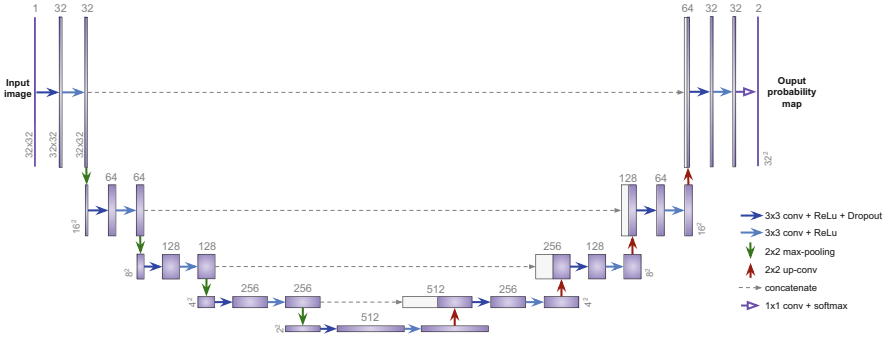


Fig. 2. Architecture of the proposed network built upon U-Net.

2.1 Image Pre-processing

Due to fundus images' heterogeneity, they aren't adequate to be used directly as input to the network. First of all, images belonging to the same dataset often present different resolutions. A standardized image resolution is required by the developed framework. Thus, a spatial normalization is performed using a reference image as a size invariant to resize all images in the dataset.

To reduce memory consumption and training time, a conversion from RGB to grayscale images is performed. This is done by extracting the green channel which is commonly used to segment the lesions [20]. While the red channel is often saturated and with low contrast and the blue channel usually very noisy and with poor dynamic range, the green channel shows the maximum contrast between lesions and background.

Fundus images commonly suffer from non-uniform illumination and poor contrast caused by different lightning conditions in the acquisition rooms as well as retina's anatomical variability. To solve this problem, an image contrast enhancement is carried out by performing a contrast limited adaptive histogram equalization (CLAHE). This window-based technique provides a uniform distribution of grey values across an established 8×8 pixels-sized window, improving local contrast and, thus, raising the visibility of some hidden features.

To prevent the network from learning retinal images' inherent background noise, a 5×5 median filter is applied. This filter smooths image data by performing a spatial filtering on each pixel using the grey level values present in a square window surrounding that pixel. Finally, the intensity values of the images are scaled to $[0,1]$.

2.2 Network Architecture

The core element of this method is the convolutional neural network built upon U-Net architecture [14]. Unlike the usual CNN architectures with only contracting layers for image classification, U-Net is an image-to-image framework as it takes an image as input and returns a probability map as output. This is possible

thanks to the addition of an expansive path (decoder) symmetrical to the typical CNN’s contracting path (encoder) to obtain pixel-wise labeling. Precise localization with the use of context is achieved in this model through the combination of high resolution feature maps from the contracting path with upsampled outputs from the expansive path.

In similarity to the original U-Net, each contracting block of the architecture implemented in this work (see Fig. 2) is composed by two 3×3 convolution, each followed by a rectified linear unit (ReLU) activation function ($f(x) = \max(0, x)$). Afterwards, 2×2 max pooling is applied, reducing image resolution by 2. At each block, the number of filters is doubled. On the other hand, in the expansive path, the opposite happens. The same pair of convolutional layers are applied in each block, preceded by an “up-convolution”, that is, an up-sampling of the feature map (increasing image resolution by 2) followed by a 2×2 convolution. Then, the resulting feature map is concatenated with the corresponding feature map from the contracting path, size-wise. Finally, a 1×1 convolution and a pixel-wise softmax activation function are applied to obtain the desired number of classes and final probabilities for each pixel.

The modifications carried out in this implementation involve the addition of a dropout layer between two consecutive convolutional layers to avoid overfitting and the reduction of filters for all convolutional layers along the network to simplify the architecture and reduce training time while maintaining the same level of performance. Moreover, all convolutions are implemented with zero-padding to preserve the spatial size of the input image. Therefore, it becomes unnecessary to crop the feature maps from the contracting path for them to be concatenated with the feature maps from the expansive path, as established in [14], since they already present the same resolution.

2.3 Training and Testing

Like any other deep learning approach, this work involves training the network first, and then test the resulting model on new images. In most cases, lesions compose less than one percent of the total number of pixels in a retinal image. For this reason, the network computes the probability of a pixel being an exudate using local features in a square window centered on the pixel itself. Moreover, this patch-based approach is carried out to substantially increase the amount of training data, improving model’s performance.

Each time the network is trained, input images are subjected to the aforementioned preprocessing techniques and split into patches using a square sliding window with overlap. While the sliding window goes through the full images, patches partially or completely outside the FOV or containing optic disk pixels, previously detected by means of [21], are excluded. Still, the resulting patches present unbalanced classes, that is, the number of patches classified as healthy is substantially higher than the ones classified as pathological in a retinal image, which can overwhelm the net and result in overfitting to the majority class. Given N pathological patches and M healthy patches where $M \gg N$, a random selection of N healthy patches is applied to balance the classes.

At testing phase, to improve performance and obtain smoother predictions, consecutive overlapping patches with a stride of 8 pixels are used to obtain the lesion probability of each pixel by averaging probabilities over all predicted patches covering that pixel. Once again, patches partially or completely outside the FOV or containing optic disk pixels are excluded. Hence, once the predictions are generated, the resulting overlapped patches are recomposed considering the missing patches and the overlapping technique, and the final probability maps are obtained as images in the original resolution. These images can then be used for further performance evaluation.

3 Experiments

The proposed model was trained and validated on E-OPHTHA [22] public database which contains two subsets, depending on the lesion type. The subset selected to be used in this implementation contains forty-seven retinal images with the presence of exudates. These lesions are manually annotated by ophthalmologists at a pixel level. The dataset presents four different image resolutions, ranging from 1440×960 pixels to 2544×1696 pixels. After the first pre-processing technique where a spatial normalization is performed, the images were scaled down to a final resolution of 1440×960 pixels.

3.1 Implementation Details

The framework was developed using Python 3.5 and OpenCV 3.0, from the pre-processing techniques to the attainment of output probability maps. The resizing of the images as well as the model evaluation were performed using Matlab[®]R2016a. An Intel Core i7-7700K@4.20 GHz processor with 32 GB of RAM and Ubuntu 16.04 LTS as operating system was used throughout the all process. A NVIDIA GeForce[®]TITAN Xp with 12 GB of GDDR5X RAM was the GPU used. The CNN model was designed recurring to Keras framework with Theano as backend.

3.2 Training Parameters

The framework's design allows a fast and easy adjustment of parameters and datasets to be used for training and testing. Several tests were performed in which tunable parameters were adjusted according to its impact in the model's performance. In order to obtain predictions for all the images in the dataset and, thus, provide robustness to the proposed method, cross-validation was applied. For this purpose, the forty-seven images were randomly partitioned into $k = 5$ folds. In each fold iteration, out of the k partitions, a single partition is retained to test the model, while the remaining $k - 1$ partitions are used as training data.

The retinal input images and their corresponding labelled segmentations were used to train the network with the employment of stochastic gradient descent for optimization and a cross-entropy loss function. The network was trained with

a momentum of 0.9, a weight decay of $1e^{-6}$ and a fixed learning rate of $1e^{-3}$. It was set to continue its training for 2000 epochs with a batch size of 32. The input of the net were 32×32 pixels patches which were extracted with a stride of 16 pixels for both width and height.

3.3 Results

In order to obtain binary segmentation maps from the probability maps, the optimal threshold for each fold was determined by computing the best trade-off between sensitivity and specificity. At this stage, five performance evaluation metrics - accuracy (Acc), sensitivity (Sen), specificity (Spe), area under the ROC curve (AUC), and Standard Deviation (Std) - based on the number of true positive (TP), false positive (FP), true negative (TN), and false negative (FN) pixels, were used to quantify segmentation results taking into account the labelled pixels from the ground-truth provided by experts.

A pixel-level segmentation of exudates could only be accurately validated on E-OPHTHA, since it is the first public database to provide pixel-level annotations by experts. For this reason, a comparison between the proposed method and existing classical methods which carried out a validation of their pixel-level classification on this database is shown in Table 1.

Table 1. Comparative exudate segmentation results for the validation of different methods at pixel-level on E-OPHTHA database.

	Accuracy	Sensitivity	Specificity	AUC
Haloj et al. [5]	-	0.9582	-	0.9620
Imani et al. [4]	-	0.8032	0.9983	0.9370
Proposed	0.9936	0.8941	0.9931	0.9927

To further evaluate the model’s ability to generalize to heterogeneous fundus images with different acquisition methods, the proposed model was also validated on DIARETDB1 public database. This database consists of 89 retinal images acquired with the same 50° FOV digital fundus camera and, consequently, with a fixed resolution of 1500×1152 pixels. A spatial normalization revealed to be unnecessary since retinal structures are fairly comparable. Because this database contains images with the presence or absence of exudates, a subset of 42 images containing this type of lesion was selected for this experiment. Exudates are, once again, manually annotated by experts but not at a pixel-level which is equivalent to a wide amount of false positives around the lesions. For this reason, this database isn’t suitable for a semantic segmentation validation. However, this test was performed specifically to validate the model’s performance in different databases and a wider set of images.

In Table 2, the proposed method (tested in both DIARETDB1 (dtdb) and E-OPHTHA (eoph) databases) is compared with several algorithms which present

measurements for their performance at the pixel-level on private datasets or recurring to private ground-truth annotations for public databases. Even though these works don't use a common dataset or segmentation approach, this comparison is made to show the advantages of the proposed method in terms of the aforementioned measurements.

Table 2. Comparative exudate segmentation results for the validation at pixel-level of different methods on distinct datasets.

	Accuracy	Sensitivity	Specificity	AUC
Welfer et al. [6]	-	0.7048	0.9884	-
Sopharak et al. [8]	0.9910	0.8720	0.9920	-
Sopharak et al. [3]	0.9841	0.9228	0.9852	-
Harangi et al. [7]	-	0.86	-	-
Prentasić et al. [13]	-	0.78	-	-
Proposed (dtdb)	0.9701	0.8451	0.9809	0.9535
Proposed (eoph)	0.9936	0.8941	0.9931	0.9927

As it can be seen in Tables 1 and 2, the proposed method outperforms existing algorithms in most evaluation metrics. Haloi et al. [5] and Imani et al. [4] present higher values for sensitivity and specificity, respectively, on E-OPHTHA database, while Sopharak et al. [3] exceeds sensitivity values on a private dataset. Even though the proposed model's performance on DIARETDB1 isn't higher than most methods, it is still a great outcome taking into consideration the nature of this database, as it was previously explained.

Figure 3 illustrates the qualitative segmentation results on E-OPHTHA. The validation approach presents some drawbacks due to the nature of its performance evaluation method. Manual segmentation performed by humans at a

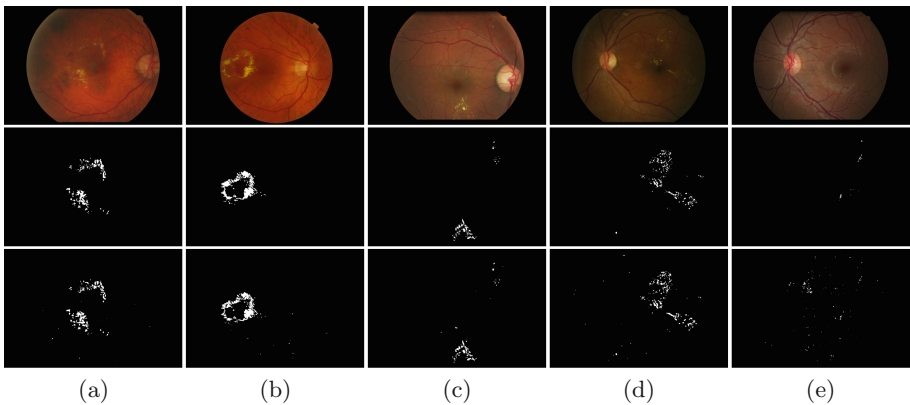


Fig. 3. Qualitative results of the exudate semantic segmentation. First row: original images; Second row: ground-truth annotations from experts; Third row: resulting segmentation maps from the proposed method.

pixel-level is prone to small-scale errors. In Fig. 3(e) it is noticeable that ground-truth annotations can be slightly misleading, originating a considerable amount of FP pixels which are, in fact, TP pixels. The remaining pixels misclassified as exudates are caused by the presence of noise and bright reflections along the main retinal vessels. There might be also some ambiguous regions where faint exudates are not considered by experts but accurately identified by the network. Furthermore, quantitative results are severely penalized in a pixel-level classification due to the small amount of pixels that are labelled as exudates in a retinal image. This means that the ratio between FN and TP pixels is inevitably lower, decreasing sensitivity values significantly. Nevertheless, the resulting segmentation maps are overall extremely similar to the expert's annotations, demonstrating the model's ability to accurately segment exudates, even in challenging situations such as in Fig. 3(e), where the image presents a lot of noise and uneven illumination. Prediction time is alongside segmentation accuracy when it comes to the major requirements for automated screening methods. Note that segmentation map takes around 36 s to be computed, allowing real-time feedback in clinical use.

4 Conclusions

In this work, a novel end-to-end network built on top of U-Net for semantic segmentation of exudates in fundus images is proposed. The preliminary experimental results show clear advantages of the proposed method over classical exudate segmentation algorithms.

In future work, bright reflections along the main retinal vessels will be the subject of post-processing techniques to reduce noise in segmentation results. In addition, grayscale images will be replaced by RGB images as input to the model. Finally, the proposed method will be applied to the detection of other kinds of DR related lesions.

Acknowledgements. This paper was supported by the European Union's Horizon 2020 research and innovation programme under the Project GALAHAD [H2020-ICT-2016-2017, 732613]. The work of Adrián Colomer has been supported by the Spanish Government under a FPI Grant [BES-2014-067889]. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

1. World Health Organization: Diabetes fact sheet. *Sci. Total Environ.* **20**, 1–88 (2011)
2. Verma, L., Prakash, G., Tewari, H.K.: Diabetic retinopathy: time for action. No complacency please! *Bull. World Health Organ.* **80**(5), 419–419 (2002)
3. Sopharak, A.: Machine learning approach to automatic exudate detection in retinal images from diabetic patients. *J. Mod. Opt.* **57**(2), 124–135 (2010)

4. Imani, E., Pourreza, H.R.: A novel method for retinal exudate segmentation using signal separation algorithm. *Comput. Methods Programs Biomed.* **133**, 195–205 (2016)
5. Haloi, M., Dandapat, S., Sinha, R.: A Gaussian scale space approach for exudates detection, classification and severity prediction. arXiv preprint [arXiv:1505.00737](https://arxiv.org/abs/1505.00737) (2015)
6. Welfer, D., Scharcanski, J., Marinho, D.R.: A coarse-to-fine strategy for automatically detecting exudates in color eye fundus images. *Comput. Med. Imaging Graph.* **34**(3), 228–235 (2010)
7. Harangi, B., Hajdu, A.: Automatic exudate detection by fusing multiple active contours and regionwise classification. *Comput. Biol. Med.* **54**, 156–171 (2014)
8. Sopharak, A., Uyyanonvara, B., Barman, S.: Automatic exudate detection from non-dilated diabetic retinopathy retinal images using fuzzy C-means clustering. *Sensors* **9**(3), 2148–2161 (2009)
9. Havaei, M., Davy, A., Warde-Farley, D.: Brain tumor segmentation with deep neural networks. *Med. Image Anal.* **35**, 18–31 (2017)
10. Liskowski, P., Krawiec, K.: Segmenting retinal blood vessels with deep neural networks. *IEEE Trans. Med. Imag.* **35**(11), 2369–2380 (2016)
11. Pratt, H., Coenen, F., Broadbent, D.M., Harding, S.P.: Convolutional neural networks for diabetic retinopathy. *Procedia Comput. Sci.* **90**, 200–205 (2016)
12. Gulshan, V., Peng, L., Coram, M.: Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *JAMA* **316**(22), 2402–2410 (2016)
13. Prentašić, P., Lončarić, S.: Detection of exudates in fundus photographs using deep neural networks and anatomical landmark detection fusion. *Comput. Methods Programs Biomed.* **137**, 281–292 (2016)
14. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) *MICCAI 2015*. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
15. Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Garcia-Rodriguez, J.: A review on deep learning techniques applied to semantic segmentation, pp. 1–23. arXiv preprint [arXiv:1704.06857](https://arxiv.org/abs/1704.06857) (2017)
16. Deng, Z., Fan, H., Xie, F., Cui, Y., Liu, J.: Segmentation of dermoscopy images based on fully convolutional neural network. In: *IEEE International Conference on Image Processing (ICIP 2017)*, pp. 1732–1736. IEEE (2017)
17. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440. IEEE (2014)
18. Li, W., Qian, X., Ji, J.: Noise-tolerant deep learning for histopathological image segmentation, vol. 510 (2017)
19. Chen, H., Qi, X., Yu, L.: DCAN: deep contour-aware networks for object instance segmentation from histology images. *Med. Image Anal.* **36**, 135–146 (2017)
20. Walter, T., Klein, J.C., Massin, P., Erginay, A.: A contribution of image processing to the diagnosis of diabetic retinopathy-detection of exudates in color fundus images of the human retina. *IEEE Trans. Med. Imaging* **21**(10), 1236–1243 (2002)
21. Morales, S., Naranjo, V., Angulo, U., Alcaniz, M.: Automatic detection of optic disc based on PCA and mathematical morphology. *IEEE Trans. Med. Imaging* **32**(4), 786–796 (2013)
22. Zhang, X., Thibault, G., Decencière, E.: Exudate detection in color retinal images for mass screening of diabetic retinopathy. *Med. Image Anal.* **18**(7), 1026–1043 (2014)



Comparison of Local Analysis Strategies for Exudate Detection in Fundus Images

Joana Pereira¹, Adrián Colomer²(✉), and Valery Naranjo²

¹ Campus Gualtar, University of Minho, 4710 Braga, Portugal

² Instituto de Investigación e Innovación en Bioingeniería (I3B),
Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain
adcogra@i3b.upv.es

Abstract. Diabetic Retinopathy (DR) is a severe and widely spread eye disease. Exudates are one of the most prevalent signs during the early stage of DR and an early detection of these lesions is vital to prevent the patient's blindness. Hence, detection of exudates is an important diagnostic task of DR, in which computer assistance may play a major role. In this paper, a system based on local feature extraction and Support Vector Machine (SVM) classification is used to develop and compare different strategies for automated detection of exudates. The main novelty of this work is allowing the detection of exudates using non-regular regions to perform the local feature extraction. To accomplish this objective, different methods for generating superpixels are applied to the fundus images of E-OPHTA database and texture and morphological features are extracted for each of the resulting regions. An exhaustive comparison among the proposed methods is also carried out.

Keywords: Exudates · Superpixels · LBP · Granulometries · SVM

1 Introduction

Diabetic Retinopathy (DR) is a common complication of diabetes, which is among the major causes of vision loss in the world. However, at the initial phase of the disease, the vision impairment is not easily realized by the patient [1, 2]. Exudates are one of the most prevalent signs during the early stage of DR. These lesions are formed due to the leakage of blood and its early detection can improve patients' chances to avoid blindness [3]. In Fig. 1 it is possible to observe the difference between a healthy fundus image and a retinal image containing exudates as a consequence of DR.

Manual detection of exudates by ophthalmologists is laborious and time-consuming. Therefore, automated screening techniques for exudate detection have great significance in saving cost, time and labour, allowing the ophthalmologists to make the treatment decision timely [4]. In this sense, one of the main objectives of this work is to develop and compare different strategies to locally extract information of fundus images for detecting exudates.

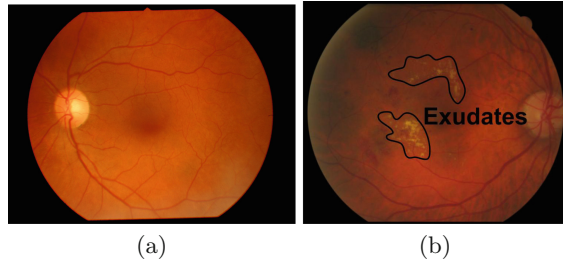


Fig. 1. Fundus images. (a) Healthy eye, (b) Pathological eye with exudates.

Several methods related to the automatic detection of exudates have been proposed in the literature, these can be grouped in: thresholding-based [5,6], region growing-based [3,7] and morphological-based classification [6,8]. These methods focus their efforts in the exudates segmentation, however, these approaches are characterised by presenting a high false-positive rate. For this reason, in the methodology proposed in this work, the characterisation of healthy and damaged retinal areas is performed by applying image descriptors in a local way, avoiding the segmentation step. Methods involving feature extraction and classification of the retinal tissue have been studied in the literature. The most common procedure requires the extraction of features from a lesion candidate map generated by different techniques, such as: mathematical morphology [4,6]; background subtraction [9]; clustering [10]; or using banks of filters and applying a low adaptive threshold [11,12]. Subsequently, the candidates are classified as exudate or non-exudate elements, making use of the extracted features and classification algorithms.

The strategies proposed in this work do not require the previous segmentation of exudates or the generation of candidate maps. Exudates usually represent less than one percent of the total number of pixels that compose the retinal image. For this reason, in the methodology presented in this paper, the image is divided in regions or patches and, during the feature extraction step, features vectors are extracted for each region. Different methods to generate superpixels are presented as a strategy to create non-regular regions.

Superpixels are regions resulting from a low-level segmentation of an image and are typically used as primitives for further analysis such as detection, segmentation, and classification of objects. The underlying idea is that this first low-level partition decreases the computational complexity of the following processing steps and improves their robustness [13]. Methods for generating superpixels have been studied in the literature. Those can be broadly categorized as: graph-based algorithms [14,15]; gradient ascent methods [16,17]; or clustering methods [18]. Of these, the latter has particular interest since the Simple Linear Iterative Clustering (SLIC) algorithm [18], which is an adaptation of k -means for superpixel generation, will play an important role during the evaluation of this work. Machairas et al. [19] also proposed a strategy to create superpixels by applying the watershed transformation to a spatially regularized gradient

to achieve a tunable trade-off between superpixels regularity and adherence to object boundaries. With this approach, the term “waterpixels” was introduced in the literature. The strategies proposed in this paper rely on the marker-controlled watershed transformation, in order to efficiently generate waterpixels.

The main objective of this work is to develop and compare different strategies to obtain regions of interest with the purpose of locally describing healthy and pathological retinal areas. Allowing the detection of exudates using non-regular regions to perform the local feature extraction is the main novelty of this work. Therefore, two different strategies for generating waterpixels are applied to the images of the E-OPHTA database. Subsequently, texture and morphological features are locally extracted and, finally, each region is classified according to healthy and pathological classes, during the classification stage. In the end, an elaborated comparison between the proposed strategies for generating waterpixels and the SLIC superpixels is also performed.

2 Methods

This paper’s methodology allows the characterisation of healthy and damaged retinal areas by applying image descriptors in a local way. Therefore, it is possible to summarise the methodology proposed into three principal steps: computation of waterpixels; local feature extraction; and, finally, classification of the retinal tissue. These steps will be developed in the following subsections.

2.1 Computation of Waterpixels

Creation of the Grid and Gradient Definition. The first step consists in computing a grid of regular cells C_i . A grid of hexagons is created with the size of the input image. Note that the hexagon side s is a tunable parameter. The smaller s is, the smaller the resulting waterpixels and the higher the number of resulting regions. The distance between adjacent hexagon centres is denoted as σ and it plays a normalization role in the waterpixel computation process.

At this stage, the computation of the morphological gradient g from the input image $f : D$ (where D is a rectangular subset of Z^2) is performed. The morphological gradient can be defined as:

$$g(f) = \delta_B(f) - \epsilon_B(f) \quad (1)$$

where δ and ϵ are, respectively, the operators dilation and erosion and B is a unitary structuring element.

Selection of the Markers. The selection of markers allows us to control the number and regularity of the resulting waterpixels: the number of markers is equal to the number of waterpixels in the final partition; to obtain regions which are similar in shapes and sizes, it is necessary to select the markers in a way that they are regularly spaced out over the image.

Taking this into account, a unique marker per cell is selected to obtain total control over the number of waterpixels, and a strong impact on their size and shape. During this procedure, it is necessary to have in mind that the ideal is to find a marker that enables to obtain the best performance in terms of boundary adherence and regularity. Therefore, two approaches are tested: selecting the centres of the grid cells as markers (c-Waterpixels) and selecting one minimum of the gradient per cell as markers (m-Waterpixels). In the second approach, each cell C_i of the grid defines a region of interest where the content of g is analysed to select a unique marker, taking into account the following considerations:

- if there exist more than one minimum of g inside C_i , the one with the highest surface extinction value [13] is selected.
- if there is no minimum of g inside C_i , the centre of C_i is defined as marker (to guarantee regularity).
- if there is a unique minimum of g inside C_i , this minimum is obviously selected as marker.

Spatial Regularisation of the Gradient and Watershed Computation.

A spatially regularised gradient g_{reg} is computed to guarantee a compromise between boundary adherence and regularity:

$$g_{reg} = g + kd_Q \quad (2)$$

where g is the gradient of the image, k is the spatial regularisation parameter, which, in this work, is set to one. Finally, let $Q = q_{i|1 \leq i \leq N}$ be a set of N connected components of the image f . For all $p \in D$, we can define a distance d_Q with respect Q as follows:

$$\forall p \in D, d_Q(p) = \frac{2}{\sigma} \min_{i \in [1, N]} d(p, q_i) \quad (3)$$

where σ is the grid step defined in the previous section.

Finally, the watershed transformation is performed on the spatially regularised gradient g_{reg} , starting the flooding from the markers. This allowed the partition of the image into waterpixels regions. Figure 2 shows the resulting c and m-Waterpixels and Fig. 3 illustrate the described steps to create this regions.

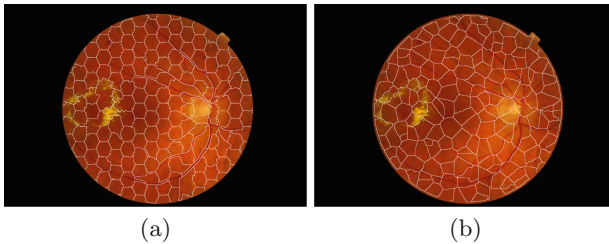


Fig. 2. Comparison of the two proposed strategies to generate waterpixels. (a) c-Waterpixels and (b) m-Waterpixels.

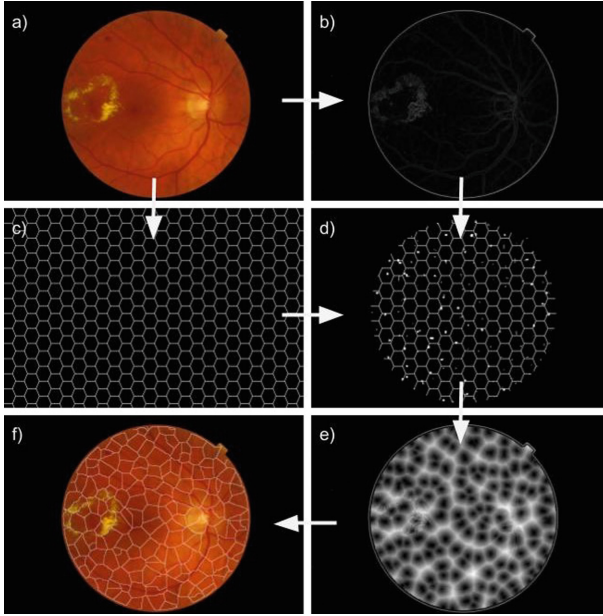


Fig. 3. Illustration of m-Waterpixels generation. (a) Original image, (b) Morphological gradient of the original image, (c) Regular grid of hexagonal cells ($s = 32$), (d) Selected markers within the regular grid, (e) Spatially regularised gradient, (f) m-Waterpixels.

2.2 Feature Extraction

Local Binary Pattern (LBP) is a simple yet very efficient texture descriptor which labels the pixels of an image by thresholding the neighborhood of each pixel:

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c) \cdot 2^p, \quad s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \quad (4)$$

where P represents the number of samples on the symmetric circular neighbourhood of radius R , g_c is the grey value of the pixel (i, j) and g_p is the grey value of each neighbour.

When LBP is used for texture description it is common to include a contrast measure by defining the Rotational Invariant Local Variance (VAR) as:

$$VAR_{P,R} = \frac{1}{P} \sum_{p=0}^{P-1} (g_p - \mu)^2, \quad \mu = \frac{1}{P} \sum_{p=0}^{P-1} g_p \quad (5)$$

$LBP_{P,R}^{riu2}$, defined in [20], and $VAR_{P,R}$ are complementary measures and the combination of both is expected to be a powerful descriptor for detecting abnormal retinal patterns. $LBP_{P,R}^{riu2}$ and $VAR_{P,R}$ operators are locally computed for each pixel of the green channel of the retinal image. As a result, LBP and

VAR images are obtained. These resulting images are divided into waterpixels regions and normalised histograms are computed for each region combining the information provided by both images using the method proposed in [21].

Granulometries. Mathematical morphology has become increasingly relevant in the image processing field essentially due to its versatility and rigorous mathematical description. Granulometry is one of the most interesting techniques based on mathematical morphology.

Let f be a grey-level image, $f \in F(E, T)$ such that $f(x) : E \rightarrow T$ where $x = (x, y) \in E$ is the pixel position and $E \subseteq \mathbb{Z}^2$. T is an ordered set of grey-levels. After the selection of a fixed set $B \subseteq E$, the two elementary operations of erosion ($\epsilon_B(X)$) and dilation ($\delta_B(X)$) can be composed together to generate a new set of grey-level operators given by the grey-level opening (γ_B) and closing (φ_B):

$$\gamma_B(f)(x) = (f \circ B)(x) = \delta_B(\epsilon_B(f))(x) \tag{6}$$

$$\varphi_B(f)(x) = (f \bullet B)(x) = \epsilon_B(\delta_B(f))(x) \tag{7}$$

When a series of openings with SE of increasing size (λ) are sequentially computed on the image, a morphological opening pyramid is obtained and it can be defined as:

$$\Pi_\gamma(f) = \{\Pi_{\gamma_\lambda} : \Pi_{\gamma_\lambda} = \gamma_\lambda(f), \forall \lambda \in [0, \dots, n_{max}]\} \tag{8}$$

where n_{max} represents the maximum size of the structuring element. By duality, the morphological closing pyramid Π_φ is defined in the same way.

A shape descriptor can be defined using the morphological pyramids above described. Let $m(f)$ be the Lebesgue measure of a discrete image f . The granulometry curve, or pattern spectrum of f with respect to Γ is defined as:

$$PS_\Gamma(f, n) = PS(f, n) = \frac{m(\Pi_{\gamma_n}(f)) - m(\Pi_{\gamma_{n+1}}(f))}{m(f)}, n \geq 0 \tag{9}$$

By duality, this concept extends to the anti-granulometry curve $PS_\Phi(f)$.

Granulometry is used in this work by applying a series of morphological opening (closing) operations with increasing-size structuring elements (SE) defined by a specific step ($s = 2$) and a maximum value ($n_{max} = 22$), to obtain a local description of the shape and size of the retinal exudates. SE is a disk or a line giving place to an isotropic or angular granulometry, respectively. In the end, four feature vectors are obtained: two regarding the isotropic granulometry (opening and closing operations) and two regarding the angular granulometry. Thus, these pattern spectrums are combined giving place to the morphological feature vector composed by 44 elements. Note that angular granulometry is computed in the directions $0^\circ, 45^\circ, 90^\circ$ and 135° . The regions containing optic disk pixels are not considered during the feature extraction stage. To detect the optic disk, the method proposed in [22] is used. Figure 4 illustrates the results obtained by applying this method.

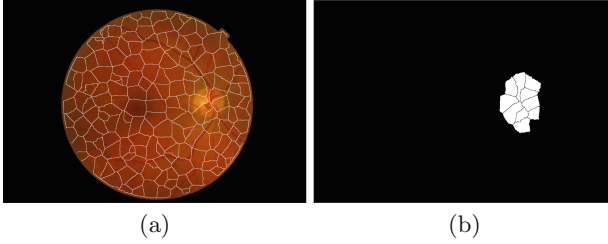


Fig. 4. Optic disk detection results. (a) Waterpixels, (b) Optic disk regions.

2.3 Retinal Tissue Classification

After the feature extraction step, the Support Vector Machines (SVM) classifier [23] is used to classify each region in healthy or pathological. This classifier generates a classification model, based on the information of features and labels of the training set, able to predict the correct class of new samples belonging to the test set. With this purpose, SVM maximises the distance between the hyperplanes defined by the support vectors with the aim of finding the optimal separation between classes. The Radial Basis Function (RBF) kernel is employed when the SVM algorithm is applied. The library for support vector machines (libSVM) used was introduced by Chang in [23].

An external K -fold cross validation is performed to provide robustness to the decision system and to guarantee that all the instances are used in the creation of the model and in the prediction stage. A different partition is selected as test subset while the rest of the partitions are used to train the model.

As it was previously discussed, exudates represent only a small percentage of the total number of pixels that compose the retinal image, which results in a very unbalanced dataset when the feature extraction is performed. Training the classifier with this dataset can result in an overfitting to the class “healthy”.

To avoid this problem, the set of all healthy samples is randomly permuted and partitioned into $T = \text{round}(M/N)$ subsets, being M and N the number of healthy and pathological samples, respectively. After that, a set of T classifiers is trained using all pathological training samples and each partition of healthy training samples. Finally, during the test stage, testing samples are evaluated for each of the T models and soft majority voting is applied to the output probabilities as the final criterion. If the obtained probability is higher than a given threshold (δ), the region is assigned to the class “pathological”.

3 Results

The database chosen to validate the method proposed in this project was the E-OPHTHA public database [24]. This database is divided in two subsets and the one used in this paper is composed by 47 images containing exudates. These lesions are manually annotated by experts. It was necessary to perform a spatial normalization of the fundus images since they presented different resolutions.

Two different strategies to obtain waterpixels were presented in this paper. These regions were used to perform a local feature extraction in fundus images for detecting pathological areas. To evaluate these strategies, the procedure explained in Sect. 2.3 was used to train different classification models.

In order to compare the strategies proposed in this paper with another state-of-the-art method, the same procedure was applied using the SLIC algorithm to generate superpixels. This algorithm adapts a k -means clustering approach to efficiently generate superpixels. By default, the only parameter of the algorithm is k , which is the desired number of approximately equally sized superpixels. Several tests using the original images (OI) and the images without the blood vessels (W/V) were performed for each of the strategies mentioned. The images without blood vessels were obtained through the inpainting technique [25].

Table 1 contains the Area Under ROC Curve (AUC), accuracy, sensitivity and specificity (as well as the standard deviation associated with each of these metrics) of the resulting classification processes, taking into account the ground-truth provided by the ophthalmologists. The results of accuracy, sensitivity and specificity were obtained using a decision threshold of $\delta = 0.5$.

As it can be observed through the analysis of Table 1, the values of accuracy improve, for each of regions of interested developed, when the inpainting technique is applied in order to remove the blood vessels from the original images. During the evaluation of the different techniques it is important to remember the importance of obtaining a good trade-off between the sensitivity and specificity. Additionally, in the context of this paper, sensitivity plays a major role since

Table 1. AUC, accuracy, sensitivity and specificity related to the exudate detection on the original images (OI) and images without vessels (W/V) for each of the region of interest developed: m-Waterpixels, c-Waterpixels and SLIC superpixels.

	m-Waterpixels		c-Waterpixels		SLIC Superpixels	
	OI	W/V	OI	W/V	OI	W/V
AUC	0.7998 \pm 0.0317	0.8287 \pm 0.0246	0.7937 \pm 0.0325	0.8277 \pm 0.0199	0.7987 \pm 0.0311	0.8169 \pm 0.0299
Accuracy	0.6514 \pm 0.1093	0.7585 \pm 0.0676	0.6650 \pm 0.0870	0.7545 \pm 0.0679	0.7635 \pm 0.0238	0.8037 \pm 0.0222
Sensitivity	0.7779 \pm 0.0987	0.7216 \pm 0.0834	0.7651 \pm 0.0477	0.7496 \pm 0.0295	0.6567 \pm 0.0824	0.6656 \pm 0.0634
Specificity	0.6363 \pm 0.1295	0.7595 \pm 0.0789	0.6546 \pm 0.0990	0.7539 \pm 0.0757	0.7736 \pm 0.0305	0.8175 \pm 0.0273

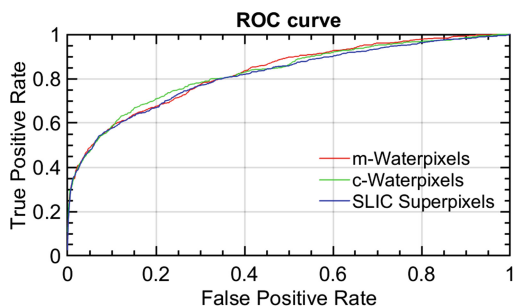


Fig. 5. ROC curves for the different superpixels methods.

this measure represents the ability of our system to detect exudates. Having that in mind, and through the analysis of Table 1, it is possible to conclude that the waterpixels approaches outperform the SLIC [18] method when it comes to generate the most appropriate regions to detect exudates.

The ROC curves for each of the tests performed with the images without vessels are represented in Fig. 5. As it can be observe, the c-Waterpixels outperform the other methods for lower false positive rates while the m-waterpixels exceed the c-Waterpixels and SLIC superpixels for highest false positives rates.

4 Conclusions

In this paper, a system based on local feature extraction and SVM classification is used to develop and compare different strategies of exudates automated detection. These strategies involve performing the local feature extraction using three different methods to generate non-uniform regions: m-Waterpixels, c-Waterpixels and SLIC superpixels. Analysing the results obtained for each of the proposed methods, it is possible to conclude that the watershed-based approaches lead to a better detection of pathological areas. The results also demonstrate that applying the inpainting technique to remove the blood vessels is essential to obtain a more accurate detection of exudates. Future work will allow the detection of other lesions related to DR, such as microaneurisms and hemorrhages.

Acknowledgements. This paper was supported by the European Union's Horizon 2020 research and innovation programme under the Project GALAHAD [H2020-ICT-2016-2017, 732613]. The work of Adrián Colomer has been supported by the Spanish Government under a FPI Grant [BES-2014-067889]. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

1. Sidibé, D., Sadek, I., Mériaudeau, F.: Discrimination of retinal images containing bright lesions using sparse coded features and SVM. *Comput. Biol. Med.* **62**, 175–184 (2015)
2. Zhou, W., Wu, C., Yi, Y., Du, W.: Automatic detection of exudates in digital color fundus images using superpixel multi-feature classification. *IEEE Access* **5**, 17077–17088 (2017)
3. Sinthanayothin, C., et al.: Automated detection of diabetic retinopathy on digital fundus images. *Diabet. Med.* **19**(2), 105–112 (2002)
4. Walter, T., Klein, J.C., et al.: A contribution of image processing to the diagnosis of diabetic retinopathy-detection of exudates in color fundus images of the human retina. *IEEE Trans. Med. Imaging* **21**(10), 1236–1243 (2002)
5. Ali, S., et al.: Statistical atlas based exudate segmentation. *Comput. Med. Imaging Graph.* **37**(5–6), 358–368 (2013)
6. Zhang, X., Thibault, G., Decencièrre, E., Marcotegui, B., et al.: Exudate detection in color retinal images for mass screening of diabetic retinopathy. *Med. Image Anal.* **18**(7), 1026–1043 (2014)

7. Li, H., Chutatape, O.: Automated feature extraction in color retinal images by a model based approach. *IEEE Trans. Biomed. Eng.* **51**(2), 246–254 (2004)
8. Welfer, D., Scharcanski, J., Marinho, D.R.: A coarse-to-fine strategy for automatically detecting exudates in color eye fundus images. *Comput. Med. Imaging Graph.* **34**(3), 228–235 (2010)
9. Giancardo, L., et al.: Exudate-based diabetic macular edema detection in fundus images using publicly available datasets. *Med. Image Anal.* **16**(1), 216–226 (2012)
10. Amel, F., Mohammed, M., Abdelhafid, B.: Improvement of the hard exudates detection method used for computer-aided diagnosis of diabetic retinopathy. *Int. J. Image Graph. Signal Process.* **4**(4), 19 (2012)
11. Akram, M.U., Khalid, S., Tariq, A., Khan, S.A., Azam, F.: Detection and classification of retinal lesions for grading of diabetic retinopathy. *Comput. Biol. Med.* **45**, 161–171 (2014)
12. Akram, M.U., Tariq, A., Khan, S.A., Javed, M.Y.: Automated detection of exudates and macula for grading of diabetic macular edema. *Comput. Methods Programs Biomed.* **114**(2), 141–152 (2014)
13. Machairas, V.: Waterpixels and their application to image segmentation learning. Ph.D. thesis, Université de recherche Paris Sciences et Lettres (2016)
14. Shi, J., Malik, J.: Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(8), 888–905 (2000)
15. Veksler, O., Boykov, Y., Mehriani, P.: Superpixels and supervoxels in an energy optimization framework. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) *ECCV 2010. LNCS*, vol. 6315, pp. 211–224. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15555-0_16
16. Comaniciu, D., Meer, P.: Mean shift: a robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(5), 603–619 (2002)
17. Levinshtein, A., Stere, A., Kutulakos, K.N., Fleet, D.J., Dickinson, S.J., Siddiqi, K.: TurboPixels: fast superpixels using geometric flows. *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(12), 2290–2297 (2009)
18. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(11), 2274–2282 (2012)
19. Machairas, V., Faessel, M., Cárdenas-Peña, D., Chabardes, T., Walter, T., Decencière, E.: Waterpixels. *IEEE Trans. Image Process.* **24**(11), 3707–3716 (2015)
20. Ojala, T., Pietikainen, M., Maenpää, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(7), 971–987 (2002)
21. Guo, Z., Zhang, L., Zhang, D.: Rotation invariant texture classification using LBP variance (LBPV) with global matching. *Pattern Recognit.* **43**(3), 706–719 (2010)
22. Morales, S., Naranjo, V., Angulo, J., Alcañiz, M.: Automatic detection of optic disc based on PCA and mathematical morphology. *IEEE Trans. Med. Imaging* **32**(4), 786–796 (2013)
23. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. *ACM Trans. Intell. Syst. Technol. (TIST)* **2**(3), 27 (2011)
24. Decencière, E., Cazuguel, G., Zhang, X., Thibault, G., Klein, J.C., Meyer, F., et al.: TeleOphta: machine learning and image processing methods for teleophthalmology. *IRBM* **34**(2), 196–203 (2013)
25. DErrico, J.: inpaint_nans, matlab central file exchange (2004). <http://kr.mathworks.com/matlabcentral/fileexchange/4551-inpaint-nans>. Accessed 13 Aug 2012



MapReduce Model for Random Forest Algorithm: Experimental Studies

Barbara Bobowska^(✉) and Dariusz Jankowski^(✉)

Department of Systems and Computer Networks,
Wrocław University of Science and Technology,
Wyb. Wyspiańskiego 27, 50-370 Wrocław, Poland
{barbara.bobowska,dariusz.jankowski}@pwr.edu.pl

Abstract. In the world where technology has largely dominated almost every aspect of human life the amount of data generated each minute grows at a rapid rate. The need to analyse massive volumes of data poses new challenges for researchers and specialists around the world. The MapReduce model became a center of interest due to offering a way of execution that allowed a parallelization of tasks. Machine learning also gained a significantly more attention due to many applications where it can be used. In this paper, we discuss the challenges of Big Data analysis and provide an overview of the MapReduce model. We conducted experiments to examine the performance of the MapReduce on the example of a Random Forest algorithm to determine its effect on the overall quality of the analysis. The paper ends with remarks on the strengths and pitfalls of using MapReduce as well as ideas on improving its potential.

Keywords: Big Data · MapReduce · Spark · Random Forest
Classification

1 Introduction

Data collection computer systems are used in virtually all areas of life, including technology, medicine, astronomy, economics, and sport. Collected data is stored as databases or data warehouses, which can be analysed in order to identify interesting or important information. Typical tasks connected with data mining [12] include searching for associations, information grouping and classification (which we will focus on in the following sections of this work). The first attempts to define the concept of Big Data dates back to 2001, when Doug Laney defined a coherent model characterized by three factors: the data are numerous, the data has great diversity of types, data are generated and captured very quickly. Nevertheless, Big Data processing produces a number of problems, including the following: huge amounts of data to store, large demand for computing power (CPU performance and RAM size) required for the analysis. Therefore, it is often the case that state-of-the-art data processing approaches cannot be simply applied for the purpose of Big Data.

The main contributions of this paper are as follows.

© Springer Nature Switzerland AG 2018

H. Yin et al. (Eds.): IDEAL 2018, LNCS 11314, pp. 184–194, 2018.

https://doi.org/10.1007/978-3-030-03493-1_20

1. Introduce the current state of parallel data processing with MapReduce and Random Forest.
2. Comparison of traditional algorithm design methods with MapReduce parallel programming model.
3. Comprehensive experimental evaluation of the MapReduce model on the example of a Random Forest algorithm.

The rest of the paper is organized as follows. In Sect. 2, we survey related work. Section 3 describes MapReduce as a programming model. Section 4 presents the methodology used for the parallel implementation of Random Forest Algorithm. Section 5 discusses the results of the experiment while Sect. 6 presents final conclusions and scope for future work.

2 Related Work

MapReduce is a framework that supports parallel execution and allow users to focus on data processing. Because of simplicity, high scalability and fault tolerance, MapReduce model is becoming the dominant solution in business and academic applications. One of the most popular implementation of MapReduce is Apache Hadoop [16]. In response to limitations in the MapReduce cluster computing paradigm at the University of California Spark system [11] was developed in 2012. Machine learning library that runs on the top of the Spark is Spark MLlib. Other projects worth mentioning are: Storm [7], a real-time computing system for unbounded streams of data; Dremel [4], a system for analysing read-only datasets; Apache Drill [9], a distributed system for interactive ad-hoc analysis of large-scale datasets. The major limitations of MapReduce (e.g. low efficiency, no schema, a single fixed dataflow) and methods of dealing with them can be found in [1, 5, 13].

In this work, we use Random Forest (RF) algorithm [3, 10] as base classifier. Random Forest based on decision trees in combination with aggregation and bootstrap ideas. Ensembles are generally known for high performance and robustness in relation with other classification methods. On practical point of view, RF are widely used [17]. Recent reviews and useful references for different strategies that can be used to scale Random Forest to Big Data one can find in [8]. The main proposed strategies are based on: subsampling, parallelization, adaptation of bootstrapping schemes, divide-and-conquer approach and online learning. In [2] an Apache Spark implementation called ReForeSt was introduced which employed mechanism such as random rotations in order to improve the prediction accuracy as well as deployed a local computations of trees in order to reduce the computational time of the algorithm. The changes proved to be successful as the results presented in the work indicate that the ReForeSt is more efficient in comparison with standard MLlib implementation.

3 Parallel Processing with MapReduce

Programs using MapReduce are automatically paralleled and can be run on a cluster consisting of desktop-class computers. The run-time system is responsi-

ble for partitioning of an input data, and for distributing the program among individual units. All calculations are implemented based on `Map()` and `Reduce()` functions, where:

Map() – the user-defined function assumes certain input values and produces a temporary key-value pair from them. However, what should be noted is that the resulting temporary pair belongs to a different domain than the pair of input data. What's more, these pairs may differ in the type of data:

$$\text{Map}(k1, v1) \rightarrow \text{List}(k2, v2)$$

The MapReduce library groups all temporary pairs according to the same key and gives them to the `Reduce()` function, which includes them as its arguments. This process is also known as shuffling.

Reduce() – the function accepts the given key and assigns values to it. It then merges these values to create a possibly smaller set of values. Temporary values are passed to the `Reduce()` function with the help of the iterator, thanks to which it is possible to handle a list of values that are too large to be stored in memory. The temporary and result values produced by the `Reduce()` function belong to the same domain:

$$\text{Reduce}(k2, \text{List}(v2)) \rightarrow \text{List}(v2)$$

MapReduce tasks are executed in the following steps:

1. The MapReduce library divides input data into M parts, typically 16 MB to 64 MB for each part (it is possible to control the size of the divided parts using an optional parameter). Then, multiple copies of the program are created on a cluster of many physical machines.
2. One of the copies of the program is particularly important and is called Master. The remaining copies are called Workers. The task of the Master is to assign work to individual Workers. The Master must assign M mapping tasks and R reduce tasks. The Master chooses the type of Workers, who are at rest at the current moment and assigns them specific tasks.
3. The Worker, who has been assigned the map task, reads its contents from the corresponding part of the input data. Then it creates the key/value pairs from the received set to pass them to the user-defined `Map()` function. The temporary key/value pairs produced by the `Map()` are loaded into memory.
4. Every now and then, buffered pairs are saved on the local disk, divided into R regions by the partitioning function. Locations of buffered pairs on the local disk are transferred to the Master, who is responsible for passing them to the Workers to whom the reduction tasks were assigned.
5. When a location information arrives to a Worker, it uses a remote call to read the cached data from the disks of the Workers responsible for the mapping tasks. When the Worker working on the reduction tasks has all the data about the temporary pairs, it starts sorting them according to the key so that all occurrences for the given key are in one set. Sorting is needed because usually, one Worker is responsible for data reductions coming from many mapping tasks. If temporary data is larger than disk space, external sorting is used.

6. Workers responsible for the reduction task iterate through the sorted temporary data, and for each unique key they encounter, they pass them along with the corresponding temporary data values to the Reduce() function, which, like the Map() function, is defined by the user. The output of this function is added to the output file.
7. When all mapping and reduction tasks are completed, the Master wakes up the user program. Notification from MapReduce returns to the user's program.

The programming paradigm described above is used by the infrastructure for requesting and processing large data, including Apache Spark. All operations in the Spark are performed using resilient distributed datasets (RDD). Spark only provides tools for data processing. Therefore, it must be integrated with a file system such as HDFS or another noSQL database and a YARN management system. The previously mentioned RDD is a key concept in understanding the operation of the Spark platform. RDDs are collections of read-only objects partitioned into individual machines. These objects can be rebuilt using lineage, thanks to which they are resistant to execution errors, available due to parallel operation, and can be saved and read from distributed file systems, e.g. HDFS. Most importantly, they can be stored in the cache of individual machines responsible for carrying out single tasks (workers nodes), thus enabling immediate access to them. The next important concepts are transformations and actions performed on RDDs. Transformations determine the actions performed on the data in order to obtain subsequent RDD objects, transformed according to expectations. Actions, unlike transformations, return a specific value. The lazy evaluation mechanism is based on the use of a structure called Direct Acyclic Graph (DAG), in which the whole history of transformations made on RDDs is recorded. The key components of Apache Spark are: Spark Core - responsible for basic system functions (e.g. API RDDs), Spark SQL - supporting API, allowing communication with Spark using Apache Hive, together with the SQL variant called Hive Query Language (HiveQL), Spark Streaming - allowing for real-time processing of stream data, MLLIB - a library containing the most important machine learning algorithms and GraphX - a library containing the implementation of algorithms and graph processing tools [6, 14, 15].

4 Parallel Implementation of Random Forest Algorithm

One of the most popular projects that uses YARN [16] is Apache Spark which introduces a more robust framework based on the Map Reduce paradigm. With Spark it is possible to run interactive queries as well as iterative algorithms. Spark consists of several components (that extends its usability above Hadoop) like [2]: Spark Streaming used for streaming data analysis and Mllib framework that includes machine learning algorithms. Due to our reasoning being creating a simple yet effective application to deal with large data sets analysis we decided to use Sprak's Mllib library to implement the Random Forest algorithm. Random Forest is a decision tree-based ensemble method used for both classification

and regression tasks. An ensemble classifier makes a prediction based on the combined results from base classifiers (decision trees). Decision trees use tree-like decision model where nodes represent a condition that splits the tree into branches that could either lead to another node or if the branch does not split anymore with a leaf. For classification task, the leaf is simply a class label. Decision trees use cost function to calculate which of the candidate splits is best. One of the measures of calculating the benefit of a split is Gini index [12].

To adhere to the internal workings of the framework the objects in the data sets were represented as follows - the first value represented the label and the following the features of an object preceded by its index and separated by a colon. The presented algorithm utilities structures and mechanism native to Spark i.e. resilient distributed datasets (RDDs) and Direct Acyclic Graph (DAG). The steps of the algorithm are shown in Table 1.

Table 1. Random Forest algorithm implemented in Spark

Algorithm 1. Random_Forest_Spark

```

1: Initialize SparkContext
2: Loading the data set
3: Splitting the data set into training_data and test_data
4: Training the Random_Forest model:
5:     Set the starting parameters of the algorithm
6:     Use bootstrapping to generate a subset to train every
       tree in the ensemble on
7:     Train each tree in parallel
8:     Calculate the prediction using majority vote
9: Calculate the classification error
10: end;

```

5 Experiments

The application was run on Apache Spark 2.1.0 using HDFS as the file system and YARN as a resource manager. We tested several data sets with number of 100k, 500k, 1M, 10M and 20M instances using seven different cluster configurations build by utilizing the Google Cloud Platform service. The clusters composed of standard 2 and 4 core commodity PC's. For more detailed description refer to Table 2, where:

- No. of cores - the number of cores in a single PC in the cluster;
- No. of Worker Nodes - the number of Workers excluding the Master Node;
- RAM (single machine) - the RAM memory of a single PC;
- RAM (cluster) - the RAM memory of all the machines used as Worker Nodes; for configuration 1 and 5 the RAM memory of a cluster is simply the memory of a single machine since there's only one machine that will run task from both Master and Worker nodes;

- Disk capacity - the amount of disk storage for each machine;
- YARN - the number of YARN instances counted as $No. of Workers \cdot No. of cores$;
- YARN mem. - the amount of memory allocated to YARN counted as $RAM(cluster) \cdot 0.8$;

As a measure of comparison, we implemented an additional ensemble using SKlearn library’s Random Forest classifier to showcase the performance of the algorithm implemented using parallelized MLib framework against a method developed using the traditional approach. Each instance was run given different parameters of the Random Forest algorithm, meaning the tree depth and number of trees in the ensemble. For configurations of a cluster with no worker node computations for instance of 10M and 20M were omitted due to too long for the objective designs of the project. Tests were performed for each configuration and data set using four different values for the number of trees in the ensemble as well as three different values for the depth of a single tree and then repeated 5 more times, to ensure the stability of obtained results. The metrics used to measure the performance were *time* (in seconds) and *accuracy* defined as the number of correctly classified instances divided by all the instances in a data set.

Table 2. Cluster configurations

Conf.	No. of cores	No. of worker nodes	RAM (single machine)	RAM (cluster)	Disk capacity	YARN	YARN mem.
1	2	0	7.5 GB	7.5 GB	10 GB	2	6 GB
2	2	2	7.5 GB	15 GB	10 GB	4	12 GB
3	2	4	7.5 GB	30 GB	10 GB	8	24 GB
4	2	6	7.5 GB	45 GB	10 GB	12	36 GB
5	4	0	15 GB	15 GB	10 GB	4	12 GB
6	4	2	15 GB	30 GB	10 GB	8	24 GB
7	4	4	15 GB	60 GB	10 GB	16	48 GB

Our aim was to not only analyse the performance of the framework based on a MapReduce paradigm itself but also confront it against traditional methods of designing an algorithm to showcase the potential as well as drawbacks of such solution. In order to achieve it we used different configuration to gauge how different parameters of the cluster affect its overall work.

One of the questions was how significant of an impact does the processing power and memory capacity of a single machine has on the cluster as opposed to the number of Workers included in it. The results can be observed on Fig. 1. Comparing two clusters one made of two core PCS and consisting of four Workers and the other made of four core PCs and only two Worker machines. The obtained results indicate that those two qualities of the cluster affects its execution in a similar manner i.e. the cluster composed of two core PCs with a number of four Workers can perform just as well if not better than a cluster made of PCs with four core CPUs but two times fewer Workers.



Fig. 1. Comparison between execution time for a cluster made of PCs with 2 core CPUs and 4 Workers and a cluster with 2 Workers and PCs with 4 core CPUs

It's of utmost importance to note that while both Hadoop and Spark are great tools when dealing with large data sets they are not an especially right choice when performing an analysis of data sets of smaller volume. What one has to consider is that the execution time in MapReduce framework is dependent on the time in which the slowest map and reduce tasks are completed. Balancing the workload in between the phases in the framework may very well be the most challenging part of designing the algorithm the hardest especially managing the input for the reduce tasks so that the workload for this specific phase is as even as possible. Additionally the management of distribution of the tasks by YARN causes additional delay. Both the strengths and pitfalls of a solution based on MapReduce paradigm can be observed on Fig. 2 where it's noticeable that for smaller data sets e.g. 100k or 500k instances the benefits of using MapReduce is very scarce. The advantage really shows for data sets of 1M and more instances where the execution of MapReduce based solution outperforms the traditional approach based on Fig. 3. Not taking into consideration the aforementioned issues of MapReduce framework can lead to a situation where simply upgrading i.e adding more machines to the cluster may not generate much gain.

Comparing traditional approach with MapReduce implementation demonstrated that for data sets consisting of 100k instances the method former i.e. application run locally using SKlearn performed slightly better than the former using Spark's MLlib - Fig. 4. That discrepancy can be of course attributed to the implementation of the classifier itself. For data set of 500k the situation however becomes much more clear as the performance of the SKlearn-based algorithm decreases sharply as opposed to the MLlib implementation. Once again its worth noticing that while for that particular set the use of solution utilizing the MapReduce model can be beneficial it is still obvious that deciding on proper

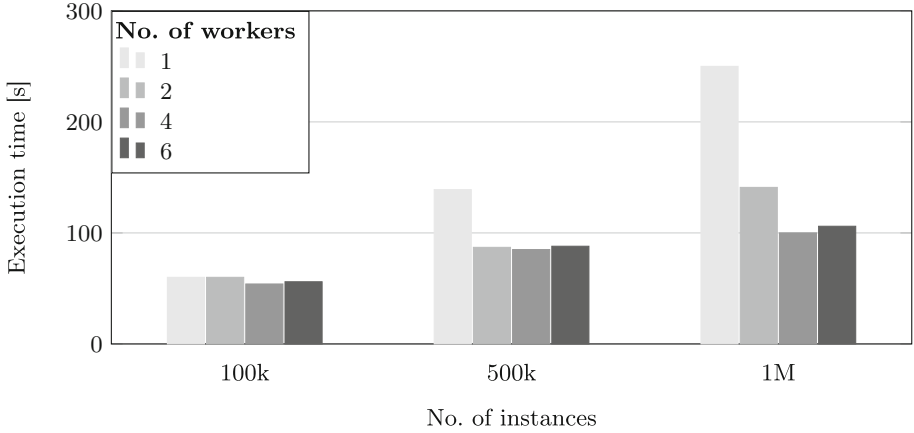


Fig. 2. Execution time for data sets of 100k, 500k and 1M instances on PCs with 2 core CPUs. Tree depth = 3, number of trees = 100

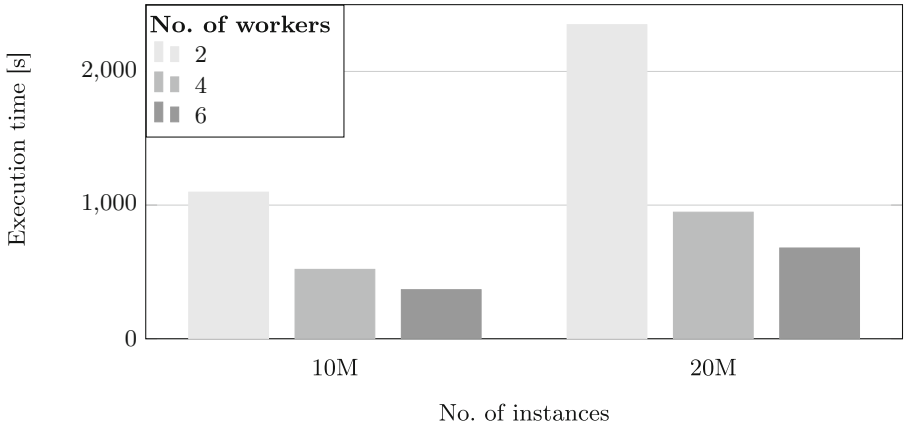


Fig. 3. Execution time for data sets of 10M- and 20M instances on PCs with 2 core CPUs. Tree depth = 3, number of trees = 100

configuration of the cluster is an issue that should be addressed carefully. A general assumption can be made that the bigger the data set the more beneficial it is to use more workers. However the issue is far more complex due to the need to balance the workload on each of the workers as well as manage the resources of YARN.

Another important aspect that needed to be addressed was the influence of using different number of clusters on the predictive abilities of the classifier. Tables 3 and 4 present the value of classification error in relation to the number of instances in the data set and the number of workers in the cluster. There is no relation on the quality of prediction and the number of workers, so that parameter cannot affect the accuracy of the classifier.

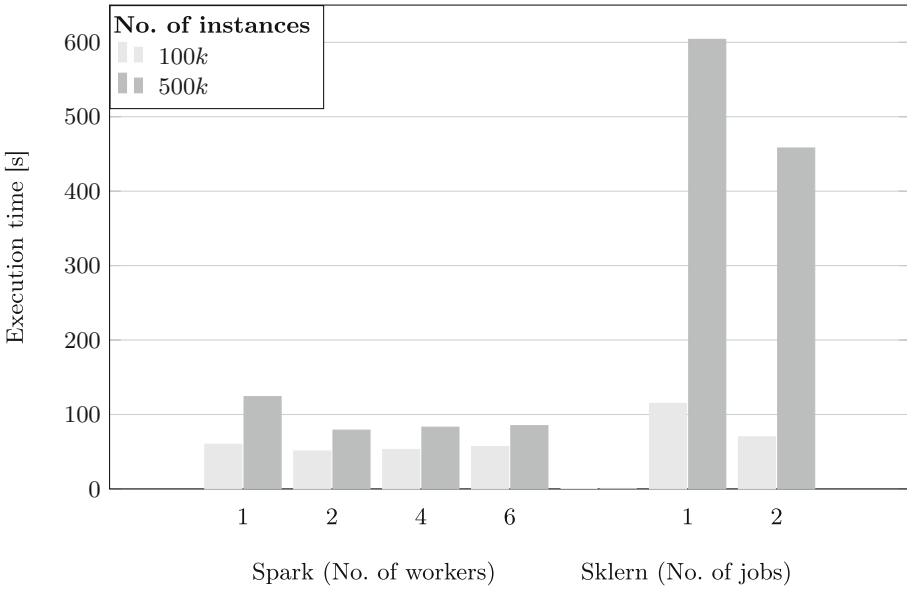


Fig. 4. Comparison between execution time of MLlib implementation and Sklearn on PCs with 2 cores. Tree depth = 3, number of trees = 50.

Table 3. The classification error for MLlib implementation. Tree depth = 4, number of trees = 100

No. of workers/No. of instances	100k	500k	1M	10M	20M
2	0.09	0.09	0.07	0.07	0.07
4	0.06	0.07	0.08	0.08	0.07
6	0.07	0.07	0.07	0.07	0.08

Table 4. The classification error for MLlib implementation. Tree depth = 5, number of trees = 100

No. of workers/No. of instances	100k	500k	1M	10M	20M
2	0.06	0.06	0.06	0.06	0.05
4	0.06	0.05	0.05	0.05	0.06
6	0.05	0.05	0.05	0.05	0.06

Table 5. The execution time and speed-up values for different cluster configurations. Tree depth = 3, number of trees = 50

No. of workers	100k		500k		1 mln	
	Time [s]	Speed-up	Time [s]	Speed-up	Time [s]	Speed-up
1	60	1	124	1	210	1
2	51	1.17	78	1.56	134	1.56
4	53	1.13	83	1.53	91	2.3
6	57	1.05	85	1.49	100	2.1

Table 6. The execution time and speed-up values for different cluster configurations. Tree depth = 3, number of trees = 50

No. of workers	10 mln		20 mln	
	Time [s]	Speed-up	Time [s]	Speed-up
1	n/d	n/d	n/d	n/d
2	928	1	1855	1
4	448	2.07	800	2.32
6	301	3.08	586	3.17

6 Conclusion and Future Works

In this paper we discussed the current state of Big Data analysis presenting a quick overview of challenges and connected areas of research. We also propose an extension of the standard Random Forest algorithm for the distributed framework Apache Spark. The method has been adapted to handle Big Data.

The introduction of MapReduce model enabled a new way of tackling some of the difficulties of processing large data sets. However, one must take into consideration that not only the advantages but the potential pitfalls of the MapReduce paradigm are present. It is not in any way a golden solution for every situation and prior knowledge of the inner-workings of tools based on this approach is a must. The advantages of using MapReduce can be easily observed on datasets of larger volume, while for the smaller datasets the profit is negligible. Tables 5 and 6 showcase the speed-up obtained for datasets of five different volumes: 100 000, 500 000, 1 million, 10 million and 20 million for various configuration of the cluster, consisting of one, two, four and six worker machines. The gain in case of datasets below 1 million while present, is much less significant than in datasets consisting of 10- and 20 million of instances. It's worth noticing that there seems to be a threshold after which enlarging the number of machines in the cluster fails to improve the performance. We have demonstrated that technologies such as Apache Spark making use of the MapReduce paradigm can significantly reduce the execution time. One of the most crucial point stressed in the simulations is that for small data sets using solutions based on MapReduce model









could be ineffective in comparison with the traditional (serial) approach to data processing. The discussion sketched about implemented Random Forest can be extended. In the future not only single data streams, but also entail unbounded data should be processed in an online fashion to obtain real-time answers.

References

1. Anderson, E., Tucek, J.: Efficiency matters!. *ACM SIGOPS Oper. Syst. Rev.* **44**(1), 40–45 (2010)
2. Assefi, M., Behraves, E., Liu, G., Tafti, A.P.: Big data machine learning using apache spark MLlib. In: 2017 IEEE International Conference on Big Data (Big Data), pp. 3492–3498, December 2017
3. Breiman, L.: Random forests. *Mach. Learn.* **45**(1), 5–32 (2001)
4. Cao, Y., Sun, D.: A parallel computing framework for large-scale air traffic flow optimization. *IEEE Trans. Intell. Transp. Syst.* **13**(4), 1855–1864 (2012)
5. DeWitt, D., Stonebraker, M.: MapReduce: a major step backwards. *Database Column* **1**, 23 (2008)
6. Dittrich, J., Quian, J.: Efficient big data processing in Hadoop MapReduce. *Proc. VLDB Endow.* **5**(12), 2014–2015 (2012)
7. Foster, I., Zhao, Y., Raicu, I., Lu, S.: Cloud computing and grid computing 360-degree compared. In: Grid Computing Environments Workshop, GCE 2008, pp. 1–10. IEEE (2008)
8. Genuer, R., Poggi, J.-M., Tuleau-Malot, C., Villa-Vialaneix, N.: Random forests for big data. *Big Data Res.* **9**, 28–46 (2017)
9. Hausenblas, M., Nadeau, J.: Apache drill: interactive ad-hoc analysis at scale. *Big Data* **1**(2), 100–104 (2013)
10. Ho, T.K.: Random decision forests. In: 1995 Proceedings of the Third International Conference on Document Analysis and Recognition, vol. 1, pp. 278–282. IEEE (1995)
11. Meng, X., et al.: MLlib: machine learning in apache spark. *J. Mach. Learn. Res.* **17**(1), 1235–1241 (2016)
12. Mitchell, T.M.: *Machine Learning*. McGraw-Hill Education, New York City (1997)
13. Pavlo, A., et al.: A comparison of approaches to large-scale data analysis. In: Proceedings of the 2009 ACM SIGMOD International Conference on Management of data, pp. 165–178. ACM (2009)
14. Pol, U.R.: Big data analysis: comparison of Hadoop MapReduce and apache spark. *Int. J. Eng. Sci. Comput.* **6**(6), 6389–6391 (2016)
15. Salloum, S., Dautov, R., Chen, X., Peng, P.X., Huang, J.Z.: Big data analytics on Apache Spark. *Int. J. Data Sci. Anal.* **1**(3–4), 145–164 (2016)
16. Vavilapalli, V.K., et al.: Apache Hadoop yarn: yet another resource negotiator. In: Proceedings of the 4th Annual Symposium on Cloud Computing, p. 5. ACM (2013)
17. Ziegler, A., König, I.R.: Mining data with random forests: current options for real-world applications. *Wiley Interdisc. Rev.: Data Min. Knowl. Discov.* **4**(1), 55–63 (2014)



Specifics Analysis of Medical Communities in Social Network Services

Artem Lobantsev^(✉), Aleksandra Vatian, Natalia Dobrenko,
Andrei Stankevich, Anna Kaznacheeva, Vladimir Parfenov,
Anatoly Shalyto, and Natalia Gusarova

ITMO University, 49 Kronverkskiy prosp, Saint-Petersburg 197101, Russia
lobantseff@gmail.com

Abstract. Social networks contain a lot of useful medical information in users' and communities' posts especially about adverse drug reactions. But before processing of the medical communities, it is important to be aware of their implicit features, which could affect the reliability of the information retrieved. We use the principal component centrality evaluation to reveal features of the distribution of influence of community members. Cosine similarity was used to compare vocabularies and structural indicators of communities of different types. As a result of the research, it was found that the medical communities have significant similarities with the communities of mothers of young children, so they can be used as an extension of the information database on the collection of the drug response. In addition, medical communities may have an atypical structure with several users who have high influence in a particular group, which shows that is necessary to verify the reliability of the information retrieved.

Keywords: Social network analysis · Eigenvector centrality
Principal component centrality · Medical network · ADR
Pharmacovigilance

1 Introduction

Modern medicine is moving toward person-oriented treatment. Patient-Centered Care is an actual global trend in pharmacovigilance researches [12, 14, 17]. According to [17], with a growth in per capita prescriptions medication, it is important to detect adverse drug reactions (ADR) by reliable pharmacovigilance systems.

In the developed countries, mostly there are pharmacovigilance centers that formally collect ADR from drug manufacturers and patients. For instance, in the USA there is a US FAERS¹ maintained by US Food and Drug Administration. In

¹ Food and Drug Administration Adverse Event Reporting System: <https://www.fda.gov/Drugs/GuidanceComplianceRegulatoryInformation/Surveillance/AdverseDrugEffects>.

France, ADR collecting is maintained by the ANSM². ADR spontaneous reports can be made by professionals, and for consumers have become available since 2011.

In some developing countries, there are also formal centers for collecting ADR, but they may not focus on patient reporting. For example, as concerning to official information, in 2016, Russia entered the Good Pharmacovigilance Practice rules. However, this system is aimed not so much at fixing individual cases of ADR as on withdrawing poor-quality batches of medicines from the turnover. The practice of individual doctors and, especially, the opinion of individual patients do not fall in this database.

In this regard, the role of practitioners and patients as information providers for the pharmacovigilance system increases [6,19]. Social networks can become a promising additional source of ADR in developed countries and essential for developing countries. An analysis of social networks in which patients share experience in treatment and drug use could facilitate information gathering process by increasing the available information base.

The goal of this paper is to identify the specifics of medical communities in social networks that can affect the interpretation of users' reports, namely specifics of community structure regards to nodes with the highest level of influence (opinion leaders) [15,18]. As will be seen later, some types of communities do not have groups with a specific distribution of influence. Therefore, the objective of identifying groups with a distinct influence distribution is not as straightforward as it might seem.

Some communities can be considered as an additional source for collecting ADR in case of they discuss similar topics. Basically, the general theme of the community is reflected in its name. And communities of a close general theme can be used as an extension of the database. In order to confirm this hypothesis we compare medical communities with communities of mothers of young children and communities of trade and rent real estate.

This paper is organized as follows. In Sect. 2 we provide works related to patient-oriented treatment state-of-the-art, medical social network analysis tasks and methods used for social network analysis. Section 3 presents methods we used for comparing medical communities with non-medical ones and communities we chosen to study. The results of performed work presented in Sect. 4, followed by conclusion and future work.

2 Related Works

Borgatti et al. [3] propose several task types of network analysis, including similarity analysis (location, membership, attribute), social relations analysis (kinship, social role, affective, cognitive), interactions and flows.

² French Medicines Agency – Agence nationale de sécurité du médicament et des produits de santé.

Loscalzo and Yu [10] present various methods of social network assessment, e.g. they given common notations of the most applicable centralities, including eigenvector centrality, which we use in this paper.

Review of patient-centered pharmacovigilance state-of-the-art is made by Saleh et al. [14]. Authors selected several of the most popular resources for searching scientific publications, and looked for papers corresponding to the participation of patients in ADR reporting.

To compare several users in their drug reaction, Jiang and Yang [9] propose using cosine similarity. In their *ProfileNet* method of estimating similarity, they analyzed online medical forums to collect users posts with comments about taking drugs and compared all posts of individual users in pairs in order to estimate the similarity of each two users.

Ilyas and Radha [8] study methods for determining the influence of nodes in massive social networks. The authors propose using the principal component centrality (PCC) to enhance eigenvector centrality (EVC) and obtain more accurate result of the impact evaluation. In the study, they compare the EVC and PCC methods in determining the most influential nodes of specific networks.

Nevertheless, none of the works mentioned above discuss the specifics of processing particular communities. In the next section, it will be said that some communities may have a type of structure that can affect the results of the data analysis and should be taken into account when developing the analysis algorithm.

3 Methods and Materials

In this paper, we implement three methods for comparing social communities to assess the specificity of medical groups. These three methods can be divided into two sections: structural methods and the semantic method. Using the structural methods, we are trying to find out the structural properties of medical communities. The semantic method is represented by an analysis of the similarity of community vocabularies. The purpose of this method is to identify similar topics of discussion in different communities.

For analysis, we select groups in the VK³ social network service with mainly Russian-speaking users. Each group is a community, where users read posts on the public wall of the community, made by moderator/administrator of a particular group. Users are able to suggest their own posts and discuss every post in comments. For comparison, we chose 15 groups of different categories: medicine, real estate rental and childcare. We call these categories “medicine”, “realty” and “mommies”. In each category, five groups are selected.

In the category of medicine, the range of users in a group is from 5,000 to 3,000 users, and the number of messages on the wall is from 500 to 4,000. For groups of category of mommies, the number of users in the group is from 25,000 to 30,000, and the number of messages is between 2,000–10,000. And for groups

³ <https://vk.com>.

of the realty category, the number of users by groups is from 15,000 to 30,000, and the number of messages is between 2,000–10,000.

3.1 Structure Features

Principal Component Analysis. To estimate structural specificity, we used Principal Component Centrality evaluation (PCC). Ilyas and Radha [8] propose to calculate the eigenvector centrality (EVC) for estimating the influence of nodes, and then use PCC to recalculate the distribution of influence taking into account several influential neighborhoods.

According to this approach, we calculated EVC for each group, then took the 50 most influential nodes and calculated PCC using Eq. (1) in matrix form.

$$\mathbf{C}_P = \sqrt{(\mathbf{X}_{N \times P} \odot \mathbf{X}_{N \times P})(\Lambda_{P \times 1} \odot \Lambda_{P \times 1})} \quad (1)$$

\mathbf{C}_P denotes vector of all centralities of group nodes calculated with principal components, $\mathbf{X}_{N \times P}$ is a matrix of eigenvectors according to top P eigenvalues for N nodes; $\Lambda_{P \times 1}$ is a vector of P top eigenvalues.

Similarity Analysis of Groups Indicators Vectors. For each group in the sample, we calculated the indicators which reflect the structure properties of the communities according to [13] (descriptions are given in the Table 1). For the calculation, we used the Gephi⁴ software and the *igraph* R package [5].

Structural similarity is assessed by comparing the values of group indicators compiled in vectors \mathbf{I} . To determine the similarity of two vectors a cosine similarity commonly used [16]. Cosine similarity ranges from -1 to 1 , meaning exactly opposite vectors and absolute identical ones respectively. For example, structural similarity between group g_i and g_j is presented in (2).

3.2 Semantic Features

Vocabulary Similarity. As proposed by Jiang and Yang [9], the similarity of group vocabulary can also be estimated from the cosine similarity. The vocabulary of the group denotes a range of words used by users of a specific group.

For every group, we have prepared the word list consisting of all the words from the group public wall. After cleansing the word set, we compare the word coincidence rates for each groups pair. Thus, the calculated similarity of vocabularies of groups pair is performed as a calculation of the cosine similarity between the two word vectors \mathbf{w}_i and \mathbf{w}_j . Words in vectors are sorted such that each row in the vector \mathbf{w}_j stands for the same word as the same row in the vector \mathbf{w}_i . Vocabulary similarity between group g_i and g_j is presented in (3).

$$S(g_i, g_j) = \frac{\mathbf{I}_i \cdot \mathbf{I}_j}{\|\mathbf{I}_i\| \|\mathbf{I}_j\|} \quad (2)$$

$$S(g_i, g_j) = \frac{\mathbf{w}_i \cdot \mathbf{w}_j}{\|\mathbf{w}_i\| \|\mathbf{w}_j\|} \quad (3)$$

⁴ <https://gephi.org/>.

Table 1. Indicators used for group comparison

Indicator	Description
Active users	Number of users who left any activity on the community wall
Population	The total number of members of the group and users who left any activity on the wall
Clear population	<i>Population</i> without deactivated or deleted users
Members	The number of formal members among the <i>Clear population</i>
Share active members	The proportion of active users who are formal members of the group
Connected users	Number of users within a group who have at least one connection with another member
Connected users share	The proportion of <i>Connected users</i> in the total amount of members
Isolates	Number of isolates within the group
Isolates share	The proportion of isolates in the <i>Clear population</i>
Edges	Number of edges of the “friendship” network
Connected components	Number of connected components in the “friendship” network [4]
Vertex giant component	The proportion of vertices in the biggest connected component of the “friendship” network
Density	Density of the network of “friendship”. The density of a network is the ratio of the existing connections of the graph to all possible for a graph with the same number of vertices
Density without isolates	<i>Density</i> only among <i>connected users</i>
Modularity	The modularity value (from 0 to 1) [11]
Clusters	Number of detected clusters for a network without isolates [1]
Mean geodesics	Average geodesic distance of the “friendship” network. Geodesic distance is the shortest path between any pair of network nodes
Diameter	The diameter of the “friendship” network (the maximum geodesic distance in the graph)
Mean degree	The average value of the degree centrality of the network of “friendship” for a complete network [7]
Female share	The proportion of female users among the <i>clear population</i> of the group
Writer share	The proportion of participants who create content in the group in the form of posts or comments
Liker share	The proportion of participants who only put “likes” on the content of the group
Passive share	The share of passive users who do not “like” and do not create content. They are a subset of the formal members of the group

4 Results and Discussion

Principal Component Analysis. The results of the PCC analysis are shown in Fig. 1. The calculated centrality is regarded as the influence of the node. For most groups, the distribution of influence is similar, and close to the power law distribution. The number of the most influential node usually close to 1. Realty groups are all the similar and have a distribution of power law. However, two groups: one of medical, the other from the mommies category, have a distinctive pattern (Fig. 2). As it will be seen in the following sections, such similarity has a specific validity.

A typical scree plot, depicted in most of the distribution plots in Fig. 1 refers to the most common types of communities with a centralized structure. In these communities, leadership and hierarchical distribution of influence are clearly expressed, where the next less influential node in the range of nodes sorted by influence has much less influence than the previous one [2].

The study shows that medical and mommies groups can have a distinct distribution of the influence. Figure 2, presents two plots for medical and mommies

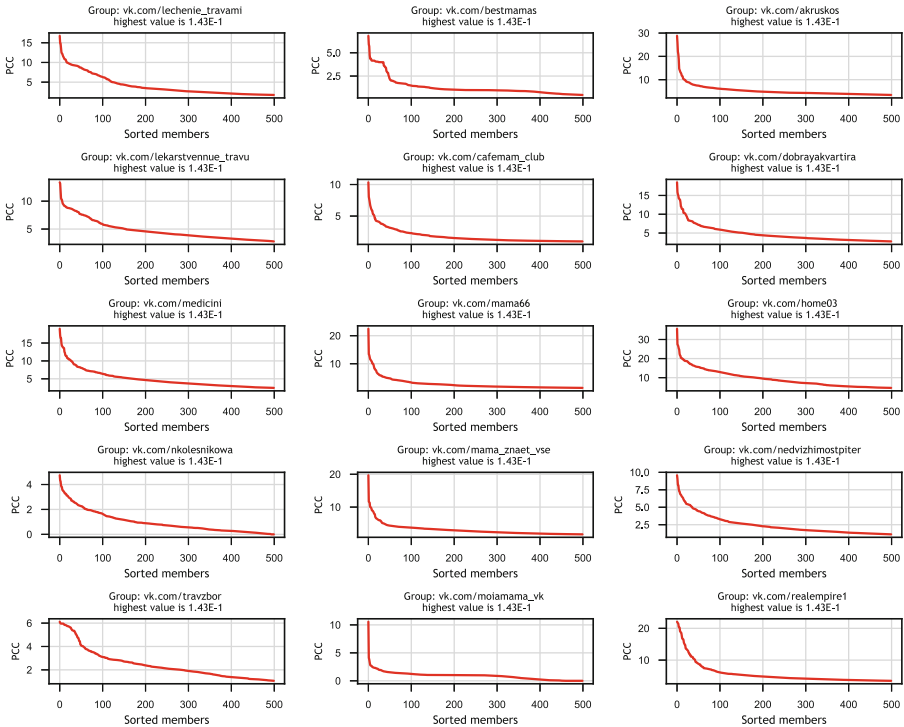


Fig. 1. PCC distribution for top 500 nodes in group. Each column refers to specific category: medical, mommies and realty.

unordinary groups. Reducing the influence of the top 25 authoritative users has a gentle slope.

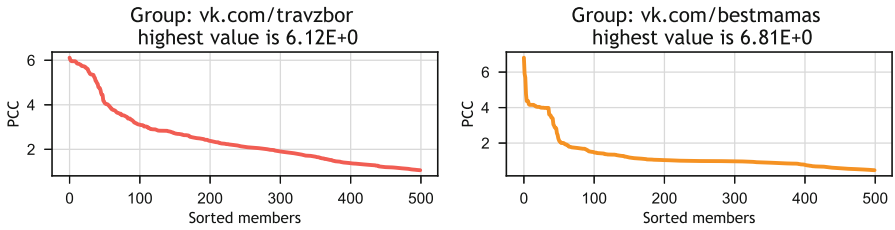


Fig. 2. Specificity of influence distribution in medical and mommies groups.

Similarity Analysis of Indicators Vectors. We decided to perform the results of the similarity analysis for community indicators in the form of a boxplot (see Fig. 3).

As shown in the plot, the medical groups have the lowest variation in the values of the indicators in this sample. In addition, using this method, we again find that the medical and mommies group have a fairly high similarity. The values of the similarity of real estate groups vary significantly, which may indicate a high dispersion of group structures in this category.

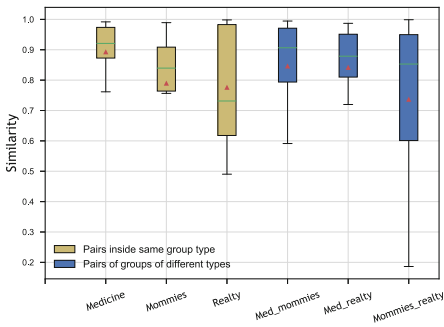


Fig. 3. Groups pairwise structure similarity. Boxplot shows the scatter of similarity values of the structure indicators vectors, obtained by pairwise comparison. The red triangle is the mean. (Color figure online)

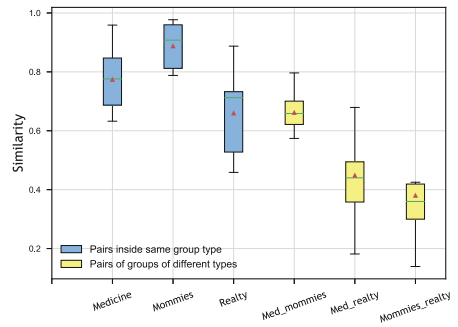


Fig. 4. Groups pairwise vocabulary similarity. Boxplot shows the scatter of similarity values of the vocabulary, obtained by pairwise comparison. The red triangle is the mean. (Color figure online)

Vocabulary Similarity. The plot in Fig. 4 shows that the groups of mommies category have the highest vocabulary similarity among all categories. The realty category has the highest variation in similarity values, which may confirm that

groups within the realty category can be quite different both in structure and vocabulary. In addition, the category of medicines and the category of mommies have the highest pairing similarity in the data. That indicates that these categories have comparable or even the same topics of interest.

Discussion. The results of the analysis of groups in given sample reveal that Russian-speaking communities in the VK.com social network service, related to medicine, can have a specific features:

- The PCC analysis showed the possibility of having a specific structure, where there is a small number of the most influential users, who have a comparable weight in the hierarchy of the community;
- The similarity analysis of indicators vectors revealed a high structural similarity of medical groups with groups in their category (groups of realty category show, that the opposite option is possible);
- The similarity analysis of indicators vectors and vocabulary similarity disclose considerable similarity of the vocabulary and structure with groups of mommies category.

Conclusion. Our study showed, that medical communities have a specific structure features, that are expressed in the presence of structures, where there is a small number of the most influential users with comparable weight in the hierarchy of the community. The presence of a such specific structure indicates the expediency of additional checking of groups of this category on the reliability of the objectivity of the opinions and proposals received. In addition, the study found that medical communities are highly similar to groups in their category and groups in the “mommies” category. That fact confirms our hypothesis made in the introduction and allows us to consider groups of mothers as an extension of the information base for collecting ADR from social networks. This work was supported by Grant 08-08.

References

1. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. *J. Stat. Mech.: Theory Exp.* **2008**(10), P10008 (2008). <https://doi.org/10.1088/1742-5468/2008/10/P10008>
2. Bodin, Ö., Crona, B.I.: The role of social networks in natural resource governance: what relational patterns make a difference? *Glob. Environ. Chang.* **19**(3), 366–374 (2009)
3. Borgatti, S.P., Mehra, A., Brass, D.J., Labianca, G.: Network analysis in the social sciences. *Science* **323**(5916), 892–895 (2009)
4. Catanese, S., De Meo, P., Ferrara, E., Fiumara, G., Proveti, A.: Extraction and analysis of facebook friendship relations. In: Abraham, A. (ed.) *Computational Social Networks*, pp. 291–324. Springer, London (2012)
5. Csardi, G., Nepusz, T.: The igraph software package for complex network research. *InterJournal Complex Syst.* **1695**(5), 1–9 (2006)

6. Foster, J.M., van der Molen, T., de Jong-van den Berg, L.: Patient-reporting of side effects may provide an important source of information in clinical practice. *Eur. J. Clin. Pharmacol.* **63**(10), 979–980 (2007)
7. Freeman, L.C.: Centrality in social networks conceptual clarification. *Soc. Netw.* **1**(3), 215–239 (1978)
8. Ilyas, M.U., Radha, H.: Identifying influential nodes in online social networks using principal component centrality. In: 2011 IEEE International Conference on Communications (ICC), pp. 1–5. IEEE (2011). <https://doi.org/10.1109/icc.2011.5963147>
9. Jiang, L., Yang, C.C.: User recommendation in healthcare social media by assessing user similarity in heterogeneous network. *Artif. Intell. Med.* **81**, 63–77 (2017)
10. Loscalzo, S., Yu, L.: Social network analysis: tasks and tools. In: Liu, H., Salerno, J.J., Young, M.J. (eds.) *Social Computing, Behavioral Modeling, and Prediction*, pp. 151–159. Springer, Boston (2008)
11. Newman, M.E.: Mixing patterns in networks. *Phys. Rev. E* **67**(2), 026126 (2003). <https://doi.org/10.1103/PhysRevE.67.026126>
12. Rathert, C., Wyrwich, M.D., Boren, S.A.: Patient-centered care and outcomes: a systematic review of the literature. *Med. Care Res. Rev.* **70**(4), 351–379 (2013). <https://doi.org/10.1177/1077558712465774>
13. Rykov, J.: *Struktura social'nyh svjazej v virtual'nyh soobshhestvah: sravnitel'nyj analiz onlajn-grupp social'noj seti "VKontakte"* (in Russian). Ph.D. thesis, Higher School of Economics - National Research University (2016)
14. Saleh, H.A., Fourrier-Réglat, A., Diogène, E.: Patient-centered pharmacovigilance: a review. *Trop. J. Pharm. Res.* **17**(1), 179–188 (2018)
15. Sela, A., Milo-Cohen, O., Ben-Gal, I., Kagan, E.: Increasing the flow of rumors in social networks by spreading groups. arXiv preprint [arXiv:1704.02095](https://arxiv.org/abs/1704.02095) (2017)
16. Sidorov, G., Gelbukh, A., Gómez-Adorno, H., Pinto, D.: Soft similarity and soft cosine measure: similarity of features in vector space model. *Computación Y Sist.* **18**(3), 491–504 (2014)
17. Smith, M.Y., Benattia, I.: The patient's voice in pharmacovigilance: pragmatic approaches to building a patient-centric drug safety organization. *Drug Saf.* **39**(9), 779–785 (2016). <https://doi.org/10.1007/s40264-016-0426-9>
18. Valente, T.W.: Opinion leader interventions in social networks. *BMJ: Br. Med. J.* **333**(7578), 1082–1083 (2006)
19. Van Grootheest, K., de Jong-van den Berg, L.: Patients' role in reporting adverse drug reactions. *Expert. Opin. Drug Saf.* **3**(4), 363–368 (2004)



PostProcessing in Constrained Role Mining

Carlo Blundo¹(✉), Stelvio Cimato², and Luisa Siniscalchi¹

¹ Università degli Studi di Salerno, Fisciano, Italy
{cblundo,lsiniscalchi}@unisa.it

² Università degli Studi di Milano, Milan, Italy
stelvio.cimato@unimi.it

Abstract. Constrained role mining aims to define a valid set of roles efficiently representing the organization of a company, easing the management of the security policies. Since the associated problems are NP hard, usually some heuristics are defined to find some sub-optimal solutions. In this paper we define two heuristics for the *Permission Distribution* and *Role Usage Cardinality Constraints* in the post processing framework, i.e. refining the roles produced by some other algorithm. We discuss the performance of the proposed heuristics applying them to some standard datasets showing the improvements w.r.t. previously available solutions.

1 Introduction

Role based access control (RBAC) is one of the most common approach to regulate system access to authorized users. In an organization, each user has assigned a set of permissions, allowing her to access some of the resources of the organization. In the RBAC model these permissions are organized in different sets called *roles* and one or more roles can be assigned to a user. Since the number of roles is smaller than the number of permissions RBAC is often used in the organizations with a huge number of users.

The process to define the roles and their assignment to the users is called *role mining*. In this work we are interested in the bottom-up approach, where starting from the permissions assigned to the users it is possible to automatically generate the roles and assign them to the users. There are already different works [2, 3, 8, 10] that explain the importance of considering constraints in the role mining process. In this work we analyze two different types of cardinality constraints, namely *role-usage cardinality constraint* (RUCC) and *permission-distribution cardinality constraint* (PDCC). In the former case the number of roles that can be assigned to a user is limited, while in the latter case a permission cannot be included in more than a fixed number of roles. In this paper, we consider the post-processing framework [8], where roles are first mined without considering the constraints, and then constraint satisfaction is imposed correcting the roles or the assignments when constraints are violated.

In literature, different approaches have considered constrained role mining, often using different terminologies and different set of constraints, see [12] for a

complete survey. We focus on the considered classes of constraints and evaluate our heuristics applying them to standard datasets considering standard metrics. The results are then compared with the ones obtained after the application of the available heuristics in literature. In particular we consider CPA [10], RPA [10], and FRUC [8] for the RUCC case, and FPDC, described in [8] for the PDCC case. In all cases, the presented heuristics improve over the previous proposals, both considering for example the size of the role set and the execution time.

2 Constrained Role Mining Problem

In this section we briefly review the basic definitions of CORE Role Based Access Control (Core RBAC, or RBAC 0) and of role mining (see [7, 16] for more details). Given a set of users $\mathcal{U} = \{u_1, \dots, u_n\}$ and a set of permissions $\mathcal{P} = \{p_1, \dots, p_m\}$, the relation $\mathcal{UPA} \subseteq \mathcal{U} \times \mathcal{P}$ maps users to permissions and $(u, p) \in \mathcal{UPA}$ describes the fact that user u possesses permission p (i.e., user u is allowed to execute an operation identified by permission p – user u can *use* the printer). A set of roles $\mathcal{R} = \{r_1, \dots, r_s\}$ is a family of subsets of permissions such that, for any $r \in \mathcal{R}$, $r \subseteq \mathcal{P}$. We also define the relations $\mathcal{UA} \subseteq \mathcal{U} \times \mathcal{R}$ and $\mathcal{PA} \subseteq \mathcal{R} \times \mathcal{P}$ that, respectively, describe user-to-role and role-to-permission assignments. Informally, given \mathcal{U} , \mathcal{P} , and \mathcal{UPA} , the role mining problem consists in finding the relations \mathcal{UA} and \mathcal{PA} *satisfying* \mathcal{UPA} . That is the relations \mathcal{UA} and \mathcal{PA} must ensure that for any $(u, p) \in \mathcal{UPA}$ there must exist at least a role $r \in \mathcal{R}$ satisfying $(r, p) \in \mathcal{PA}$ (i.e., permission p belongs to role r) and $(u, r) \in \mathcal{UA}$ (i.e., role r is assigned to user u). Moreover, if $(u, p) \notin \mathcal{UPA}$, then it cannot be that both $(u, r) \in \mathcal{UA}$ and $(r, p) \in \mathcal{PA}$. To simplify the description of our heuristics, we will represent the assignment relations by binary matrices. For instance, by \mathbf{UPA} we denote the \mathcal{UPA} 's matrix representation. The binary matrix \mathbf{UPA} satisfies $\mathbf{UPA}[i][j] = 1$ if and only if $(u_i, p_j) \in \mathcal{UPA}$. This means that user u_i possesses permission p_j . In a similar way, we define the matrices \mathbf{UA} and \mathbf{PA} . In terms of the previously defined matrices, the role mining problem consists in finding a \mathbf{UPA} 's *decomposition* into matrices \mathbf{UA} and \mathbf{PA} satisfying, for $i \in [n]$ and $j \in [m]$, $\bigvee_{h=1}^k (\mathbf{UA}[i][h] \wedge \mathbf{PA}[h][j]) = \mathbf{UPA}[i][j]$, where k is the number of columns (rows) in \mathbf{UA} (\mathbf{PA}) – k is the number of roles in the system. Such *equation* is synthesized as $\mathbf{UPA} = \mathbf{UA} \otimes \mathbf{PA}$. In other words, for any $u_i \in \mathcal{U}$, the roles assigned to u_i must exactly cover the permissions assigned to u_i in matrix \mathbf{UPA} .

As done in [3, 4, 8, 10], we can define the problem of role mining under cardinality constraints. First we assume that there is an upper bound on the number of roles that can be assigned to each user, defining the ROLE-USAGE CARDINALITY CONSTRAINT ROLE MINING problem (RUCC). Next, we set a limit on the number of roles that can include a given permission, defining the PERMISSION-DISTRIBUTION CARDINALITY CONSTRAINT problem (PDCC).

Problem 1. (RUCC) Given \mathbf{UPA} , find a decomposition $(\mathbf{UA}, \mathbf{PA})$ for which the following conditions hold: (1) $\mathbf{UPA} = \mathbf{UA} \otimes \mathbf{PA}$; (2) the number k of roles is minimized; (3) $\forall u_i \in \mathcal{U}$, we have $\sum_{j=i}^k \mathbf{UA}[i][j] \leq t$, where $t \geq 1$.

Problem 2. (PDCC) Given UPA, find a decomposition (UA, PA) for which the following conditions hold: (1) $\text{UPA} = \text{UA} \otimes \text{PA}$; (2) the number k of roles is minimized; (3) $\forall p_j \in \mathcal{P}$, we have $\sum_{i=1}^k \text{PA}[i][j] \leq t$, where $t \geq 1$.

In [8,10], the previous problems have been proved to be NP-Hard. The computational complexity of the Role Mining problem (and of some of its variants) was also considered in several papers (see, for instance, [5,6,17,18]). Other problems considering different types of constraints have been defined in [3,8,9,11].

3 Heuristics

Since finding an optimal solution to the constrained role mining problem is NP-hard, we have to resort to some heuristics to get a (not-optimal) solution. Our heuristics fall within the post-processing framework where roles are first mined irrespectively of the constraint using any known role mining algorithm. Then, the post-processing heuristic takes as input a decomposition of UPA into UA and PA and *manage* to fix the cases violating the constraint by deleting (unassigning) roles and/or adding (assigning) new roles. In the following sections we present post-processing heuristics to mine roles satisfying RUCC and PDCC constraints.

Post-processing RUCC. In the following we present an heuristic for the post-processing framework referred to **postRUCC**.

Algorithm **postRUCC** takes as input an initial decomposition of UPA into UA and PA, that not necessarily satisfies the given constraints and tries to cover all the permissions of each user by using the minimum number of roles described by PA. If the number of needed roles exceeds the threshold t , then we select the *first* $t - 1$ roles, we *transfer* the remaining uncovered permissions to a new role, assigning the $t - 1$ selected roles plus the new one to the user. Notice that, we decided to select the *first* $t - 1$ roles, but any other strategy could be used, such as random selection. Indeed, we experimentally observed that results are similar and sometimes the random choice produced worse results. Selecting, for example, the $t - 1$ roles belonging to the greatest number of users (covering then a larger part of UPA) could be convenient, but the process of determining such t roles could take a prohibitively large amount of time. More in detail, **postRUCC** algorithm, in line 3, computes a compact representation of the roles described by the PA matrix. Then, in lines 4–20 it roles are re-assigned to each user. In particular, in line 6, the algorithm **approxCover** returns a covering of user's u_i permissions using the roles in APR. Since computing the minimum number of roles needed to cover u_i 's permissions is an NP-Hard problem, **postRUCC** uses the *classical* greedy approximation algorithm solving the Set-Covering Problem [20]. Once a covering has been obtained, **postRUCC** checks whether it contains at most t roles (lines 7 and 8). If this is the case, all those roles are assigned to user u_i ; otherwise, only the *first* $t - 1$ roles returned by **approxCover** are assigned to u_i (see lines 9–12). Such an assignment is done by **update** algorithm that assigns role r to user i by appropriately modifying the **newUA** and **newPA** matrices (if r is a *new* role, then it is added to both **newUA** and **newPA**). In line 12, **postRUCC**

Algorithm 1. `postRUCC`

```

input : The  $n \times m$  matrix UPA, its decomposition into UA and PA, and the
         threshold  $t$ 
output: A new decomposition newUA and newPA of UPA satisfying the RUCC
         constraint
1 newUA  $\leftarrow [n][\cdot]$ , newPA  $\leftarrow [\cdot][m]$ 
2  $k \leftarrow$  number of rows in PA // Number of roles
3 for  $j \leftarrow 1$  to  $k$  do APR[ $j$ ]  $\leftarrow \{\ell : \text{PA}[j][\ell] = 1\}$ 
4 for  $i \leftarrow 1$  to  $n$  do
5   perms  $\leftarrow \{j : \text{UPA}[i][j] = 1\}$ 
6   COVER  $\leftarrow$  approxCover(perms, APR)
7   if  $|\text{COVER}| \leq t$  then  $\ell \leftarrow |\text{COVER}|$  else  $\ell \leftarrow t - 1$ 
8   if  $|\text{COVER}| \leq t$  then flag  $\leftarrow$  false else flag  $\leftarrow$  true
9   for  $j \leftarrow 1$  to  $\ell$  do
10     $r \leftarrow \text{COVER}[j]$  //  $j$ -th role in the cover
11    (newUA, newPA)  $\leftarrow$  update(newUA, newPA,  $i$ , APR[ $r$ ])
12    perms  $\leftarrow$  perms  $\setminus$  APR[ $r$ ]
13    if flag then
14      (newUA, newPA)  $\leftarrow$  update(newUA, newPA,  $i$ , perms)
15      if perms  $\notin$  APR then
16        | APR[ $k$ ]  $\leftarrow$  perms // Add the new role to the role-set
17        end
18      end
19    end
20 end
21 return (newUA, newPA)

```

keeps track of the u_i 's uncovered permissions. If `approxCover` returns more than t roles (see line 13), then the remaining uncovered permissions will form a role that, in line 14, is assigned to u_i . If such a role is a new one (i.e., the role represented by `perms` does not belong to the role-set represented by `APR`, see line 15), then in line 16 it is added to the role-set.

Post-processing PDCC. In the following we present an heuristic for the post-processing framework referred to as `postPDCC`. Such an heuristic, starting from a given decomposition of `UPA` into `UA` and `PA` mined irrespectively of the constraints, returns two new matrices, `newUA` and `newPA` having, respectively, n rows (one for each user) and m columns (one for each permission). The idea our heuristic relies on is to form, starting from the roles described by `UA` and `PA`, new roles including only permissions satisfying the PDCC constraint. Using such new roles the heuristic tries to cover as many as possible permissions in `UPA`. Then, it iteratively runs any role mining algorithm, generically referred to as `RM`, on an *updated* `UPA` containing only the remaining uncovered permissions. Such iteration is repeated until either the newly computed matrix `UPA` is empty (i.e., all permissions have been covered) or all the permissions in all mined roles violate the PDCC constraint. In the former case we have done; while, in the

latter case, for each permission the heuristic forms a role covering, in this way, all permissions in the *updated* UPA.

More in detail, at line 4, algorithm `postPDCC` computes the vector `NR` that keeps track of the number of roles a permission has been assigned to. If $NR[j] > t$, then permission p_j violates the PDCC constraint. After computing the vector `NR`, it verifies whether in the current role-set, represented by `PA`, there are some permissions not violating the PDCC constraint. Indeed, in lines 6–18, `postPDCC` examines one mined role r at time and forms, in line 7, another role (i.e., *candidateRole*) including all the permissions assigned to r that do not violate the PDCC constraint.

Algorithm 2. `postPDCC`

```

input : The  $n \times m$  matrix UPA, its decomposition into UA and PA, and the
         threshold  $t$ 
output: A new decomposition newUA and newPA of UPA satisfying the PDCC
         constraint
1 newUA  $\leftarrow [n][\cdot]$ , newPA  $\leftarrow [\cdot][m]$ , flag  $\leftarrow$  true
2 while flag do
3   flag  $\leftarrow$  false
4   for  $j \leftarrow$  to  $m$  do NR[j]  $\leftarrow$   $|\{i : PA[i][j] = 1\}|$  // For all permissions
5    $k \leftarrow$  number of rows in PA
6   for  $r \leftarrow$  to  $k$  do // For all roles
7     candidateRole  $\leftarrow$   $\{j : PA[r][j] = 1 \text{ and } NR[j] \leq t\}$ 
8     if candidateRole  $\neq \emptyset$  then
9       for  $i \leftarrow$  to  $n$  do // For all users
10        perms  $\leftarrow$   $\{j : UPA[i][j] = 1\}$ 
11        if candidateRole  $\subseteq$  perms then
12          (newUA, newPA)  $\leftarrow$  update(newUA, newPA, i, candidateRole)
13          foreach  $j$  in candidateRole do UPA[i][j]  $= 0$ 
14          flag  $\leftarrow$  true
15        end
16      end
17    end
18  end
19  if flag then // UPA has been modified
20    (UA, PA)  $\leftarrow$  RM(UPA)
21  else // Any permission  $p_j$  has  $NR[j] > t$ 
22    for  $i \leftarrow 1$  to  $n$  and  $j \leftarrow 1$  to  $m$  do
23      if UPA[i][j]  $= 1$  then (newUA, newPA)  $\leftarrow$  update(newUA, newPA, i, \{j\})
24    end
25  end
26 end
27 return (newUA, newPA)

```

Then, *candidateRole* is assigned to each user possessing all the permissions in *candidateRole* (see lines 9–16, the assignment is done in line 12) while all

permissions in *candidateRole* are *zeroed* in UPA (see line 13). The variable *flag* is set to **true** denoting that the matrix UPA has been modified. If so, **postPDCC** invokes the procedure **RM** on the modified UPA. If the variable *flag* is **false**, then any permission p_j has $\text{NR}[j] > t$. A role containing the permission p_j alone is formed and it is assigned to any user u_i such that $\text{UPA}[i][j] = 1$ (see lines 22–14). Notice that if a permission p_j has $\text{NR}[j] \leq t$, then p_j will appear in all new roles formed from the ones that originally contained it (see line 7). Since the *old* roles, along with the user-to-role assignment described by **UA**, covered p_j , then the new roles and the new assignments (see line 12) will cover p_j as well (this can be easily shown by contradiction). Therefore, since in line 13 we modify UPA, permission p_j will not appear in any role subsequently mined in lines 19–25. By iterating such a process, all permissions will be covered (eventually by roles possessing a single permission, see lines 24–22) and the resulting matrices **newUA** and **newPA** will represent a complete role-set satisfying the PDCC constraint.

4 Experimental Evaluation

In the following sections we present the experimental evaluation of our heuristics compared with state of the art procedures. All heuristics have been implemented in Java and tested on a MacBook Pro running OS X 10.10 on a 2.7 GHz Intel Core i5 CPU having 8 GB 1867 MHz DDR3 RAM. In order to evaluate our heuristics, we use nine real-world datasets that have been widely deployed in literature for the analysis of various role mining heuristics (see, for instance, [6, 8, 10, 11, 15]).

Table 1. Characteristics of the real-world datasets considered in this paper

Dataset	$ \mathcal{U} $	$ \mathcal{P} $	$ \mathcal{R} $	$ \mathcal{UPA} $	min#P	max#P	min#U	max#U
americas large	3485	10127	398	185294	1	733	1	2812
americas small	3477	1587	178	105205	1	310	1	2866
apj	2044	1164	453	6841	1	58	1	291
emea	35	3046	34	7220	9	554	1	32
healthcare	46	46	14	1486	7	46	3	45
domino	79	231	20	730	1	209	1	52
customer	10021	277	276	45427	1	25	1	4184
firewall 1	365	709	64	31951	1	617	1	251
firewall 2	325	590	10	36428	6	590	46	298

Table 1 summarizes the characteristics of the real-world datasets we use in this paper, they have been obtained from the datasets in our possession and from Tables 1 and 2 in [6]. More in detail, for each dataset Table 1 specifies the number of users $|\mathcal{U}|$, the number of permissions $|\mathcal{P}|$, the optimal number of roles

in an unconstrained setting $|\mathcal{R}|$, the number of user-to-permission assignments $|\mathcal{UPA}|$, the minimum and the maximum number of permissions assigned to a user (respectively, $\text{min}\#P$ and $\text{max}\#P$), and the minimum and the maximum number of users that have the same permission (respectively, $\text{min}\#U$ and $\text{max}\#U$).

To compare the heuristics on the real-world datasets of Table 1, we take into account the number of roles generated by the heuristics, the execution time of the heuristics, and a *reduced* version of the *Weighted Structural Complexity* (WSC) (see [14] for its *complete* definition). This is not a limitation as the heuristics we consider do not output a *role hierarchy* neither we allow a *direct user-permission assignment* (see [16] or [7] for their definitions). Given an UPA's decomposition into matrices \mathbf{UA} and \mathbf{PA} , the *Weighted Structural Complexity* (WSC) of $(\mathbf{UA}, \mathbf{PA})$ is defined as $|\mathcal{R}| + |\mathcal{UA}| + |\mathcal{PA}|$, where $|\cdot|$ denotes the size of the set or relation.

RUCC Heuristics Evaluation. In this section we compare four heuristics for the RUCC scenario, namely CPA [10], RPA [10], FRUC [8], and our `postRUCC`, testing them on the real-world datasets in Table 1. We start comparing the four heuristics on the *americas small* dataset. We considered six values for the constraint parameter t , in particular $t \in \{1, 3, 5, 7, 9, 11\}$. Table 2 presents three subtables comparing the heuristics with respect to role-set size (left side), execution time expressed in milliseconds (right side), and WSC (middle side). The best results are highlighted in bold face. From Table 2 one can see that, heuristic FRUC always returns a role-set bigger than the other heuristics and that CPA, except for the first two cases, is better than `postRUCC`. Our heuristic shows a negligible execution time, CPA, except for $t = 1$, is more than two thousand times slower than `postRUCC`, while the remaining two heuristics presents a low execution time. Table 2 states that CPA and RPA output a solution with WSC better than `postRUCC` that in turn has better WSC than FRUC.

Table 2. RUCC - *americas small* dataset

t	Role-set Size				WSC				Time			
	<code>postRUCC</code>	FRUC	RPA	CPA	<code>postRUCC</code>	FRUC	RPA	CPA	<code>postRUCC</code>	FRUC	RPA	CPA
1	259	463	259	259	25488	42695	25488	25488	13	364	8	28
3	235	282	279	265	21585	28116	15338	14718	5	65	73	2758
5	228	254	238	216	21512	25087	15355	14807	3	37	70	2860
7	225	242	224	209	21492	23559	15288	14830	2	24	55	2849
9	225	236	217	209	21492	22901	15260	14830	1	8	54	2830
11	225	226	214	209	21492	22116	15250	14830	1	14	60	2832

In the following, fixing the constraint value $t = 2$, we compare CPA, RPA, FRUC, and `postRUCC` on the all the real-world datasets described in Table 1. Our experiments are summarized by Tables 3a, b, and c. In Table 3a we report the number of roles generated by the four heuristics. Table 3b shows the execution time expressed in milliseconds, while the values of the WSC parameter are reported in Table 3c. In each table, for each column (i.e., dataset) the best

results are highlighted in bold face. Notice that, in Tables 3a, b, and c the entries associates to heuristics CPA and RPA, for the *customer* dataset are set to *n.a.* as, after three hours, both heuristics did not return any candidate role-set, so we decided to interrupt the execution. According to Table 3a, our heuristics always returns a role-set smaller than the other three. More precisely, heuristics CPA and RPA, for the datasets *firewall 1* and *healthcare*, and heuristic FRUC, for the dataset *emea*, return a role-set having the same size of the dataset computed by *postRUCC*. With respect to the execution time, from Table 3b one can see that our heuristic is much faster than the others. Notice that several entries of such a table are set to 1, due to the fact that the input decomposition of UPA into UA and PA *almost* satisfies the RUCC constraint. For our heuristic *postRUCC* we use the decomposition provided by SMA_R described in [1], FRUC uses the decomposition obtained by applying the minimum biclique cover based algorithm in [6], RPA starts from a decomposition based on the Optimal Boolean Matrix Decomposition algorithm [13], and CPA uses the decomposition generated by Fast Miner [19]. Table 3c shows that *postRUCC* generates in eight cases out of nine a solution with lower WSC than FRUC; while, in six cases out of nine, WSC obtained from *postRUCC* is worse than the one obtained from CPA and RPA. Only in one case, namely for the *customer* dataset, our heuristic shows a better value for the WSC parameter. For the *domino* dataset it is off by one from the minimum; while, in another case (i.e., *emea* dataset) *postRUCC* attains the minimum value as FRUC.

Table 3. RUCC Post-processing comparisons

	Datasets								
	cust	am. small	am. large	apj	fire1	fire2	emea	dom	hc
(a) Role-set size									
FRUC	5022	339	444	560	99	15	34	28	26
RPA	<i>n.a.</i>	302	529	486	90	10	36	23	17
CPA	<i>n.a.</i>	303	528	486	90	10	36	22	17
<i>postRUCC</i>	4682	237	430	485	79	10	34	21	17
(b) Execution time									
FRUC	7939	134	97	35	1	7	1	1	1
RPA	<i>n.a.</i>	79	533	63	1	1	1	1	1
CPA	<i>n.a.</i>	2428	43337	2454	57	9	28	1	1
<i>postRUCC</i>	27	2	2	2	1	1	1	1	1
(c) WSC									
FRUC	51278	32540	111359	6878	8049	2213	7280	1022	834
RPA	<i>n.a.</i>	14773	87461	5200	3256	1418	7284	753	432
CPA	<i>n.a.</i>	14840	87372	5220	3252	1418	7284	748	432
<i>postRUCC</i>	45995	21646	107458	5480	5065	1466	7280	749	496

PDCC Heuristics Evaluation. In this section we compare the heuristics *Fix Permission-Distribution Cardinality* described in Sect. 3.2 of [8] (in the following we will refer to such an heuristic as FPDC) and our heuristic **postPDCC** described in Sect. 3. In **postPDCC**, the algorithm RM can be any role mining algorithm returning a complete role-set. In our experiments we instantiate it with the role mining algorithm SMA_R described in [1]. We start comparing the heuristics FPDC and **postPDCC** on the *healthcare* dataset. We considered six values, i.e., $\{2, 4, 6, 8, 10\}$, for the constraint parameter t . Table 4 presents three subtables that compare the heuristics with respect to role-set size (left side), execution time (right side), and WSC (middle side). As done for previous experiments, the best results are highlighted in bold face. From Table 4 one can see that, heuristic **postPDCC** returns a role-set smaller than the one returned by FPDC in the first three cases, in the last two cases the role-set returned by **postPDCC** contains just one role more than the one computed by FPDC. For $t \in \{4, 6, 10\}$ both heuristics exhibit the same negligible running time, while for $t \in \{2, 8\}$ FPDC is faster than **postPDCC** (still both running times are negligible). Finally, the middle side of Table 4 states that state of the art heuristic FPDC outputs a solution with WSC from 30% to 65% larger than our heuristic **postPDCC**.

Table 4. PDCC Post-Processing Framework - *healthcare* dataset

t	Role-set size		WSC		Execution Time	
	postPDCC	FPDC	postPDCC	FPDC	postPDCC	FPDC
2	21	29	388	636	3	1
4	17	19	343	463	1	1
6	15	16	330	448	1	1
8	16	15	321	472	2	1
10	16	15	270	512	1	1

In the following, fixing the constraint t to the value 8, we compare the heuristics FPDC and **postPDCC** on all real-world datasets summarized in Table 1 Our experiments are summarized in Tables 5a, b, and c. In Table 5a we report the number of roles generated by the two heuristics. Table 5b shows the execution time expressed in milliseconds of the two heuristics, while the values of the WSC parameter are reported in Table 5c. In each tables, for each dataset the best results are highlighted in bold face. According to Table 5a, in the three cases out of nine both heuristics return equal sized role-sets. In the remaining six cases out heuristic FPDC returns a role-set smaller than our heuristic. In such cases, we noticed that the decomposition of UPA into UA and PA used in [8] as input to the heuristic FPDC *almost* satisfies the PDCC constraint (i.e., FPDC starts with a decomposition that is *almost* the sought solution). This effect is also reflected by the execution time taken by both heuristics summarized in Table 5b where most entries for FPDC are set to 1 and for the *emea* dataset the value is

Table 5. PDCC Post-processing comparisons

	Datasets								
	cust	am. small	am. large	apj	fire1	fire2	emea	dom	hc
(a) Role-set size									
FPDC	276	284	785	465	77	10	53	20	16
postPDCC	325	568	3082	476	79	10	70	20	16
(b) Execution time									
FPDC	1	156	5829	25	1	1	5	1	1
postPDCC	10892	1334	7165	192	26	7	69	2	1
(c) WSC									
FPDC	45978	20020	49421	5820	4086	1891	5939	818	472
postPDCC	46444	100924	183467	5204	2865	1554	6213	767	321

negligible. Table 5c shows that postPDCC, in five cases out of nine, generates an RBAC state with lower Weighted Structural Complexity than FPDC.

5 Conclusions

In this paper we studied the constrained role mining problem w.r.t. two types of cardinality constrains that are *role-usage cardinality constraint* (RUCC) and *permission-distribution cardinality constraint* (PDCC). For each of them we described a new heuristic in the post processing framework (i.e. refining the roles produced by some other algorithm). The last part of the work presents an evaluation of the performances of the implemented heuristics, in particular, we compare them with the solutions already available in the state of the art showing the improvements given by our algorithms. In the analysis of the performance we used standard real-world datasets and standard metrics.

Due to space limitation, we provided detailed results only for a couple of real-world datasets. We plan to run a comprehensive analysis of the proposed heuristics on all real-world datasets described in Table 1 and on synthetic ones as well. Moreover, our future objective is to provide heuristics for other kind of constraints such as *Permission-Usage Cardinality Constraint* and *User-Distribution Cardinality Constraint*.

References

1. Blundo, C., Cimato, S.: A simple role mining algorithm. In: SAC 2010, New York, NY, USA, pp. 1958–1962. ACM (2010)
2. Blundo, C., Cimato, S.: Constrained role mining. In: Jøsang, A., Samarati, P., Petrocchi, M. (eds.) STM 2012. LNCS, vol. 7783, pp. 289–304. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38004-4_19

3. Blundo, C., Cimato, S., Siniscalchi, L.: Mining roles in constrained RBAC. Submitted (2018)
4. Blundo, C., Cimato, S., Siniscalchi, L.: PRUCC-RM: permission-role-usage cardinality constrained role mining. In: COMPSAC 2017, pp. 149–154. IEEE (2017)
5. Chen, L., Crampton, J.: Set covering problems in role-based access control. In: Backes, M., Ning, P. (eds.) ESORICS 2009. LNCS, vol. 5789, pp. 689–704. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04444-1_42
6. Ene, A., Horne, W., Milosavljevic, N., Rao, P., Schreiber, R., Tarjan, R.E.: Fast exact and heuristic methods for role minimization problems. In: SACMAT 2008, pp. 1–10. ACM (2008)
7. Ferraiolo, D.F., Sandhu, R.S., Gavrila, S.I., Kuhn, D.R., Chandramouli, R.: Proposed NIST standard for role-based access control. ACM Trans. Inf. Syst. Secur. **4**(3), 224–274 (2001)
8. Harika, P., Nagajyothi, M., John, J.C., Sural, S., Vaidya, J., Atluri, V.: Meeting cardinality constraints in role mining. IEEE Trans. Dependable Secur. Comput. **12**(1), 71–84 (2015)
9. Hingankar, M., Sural, S.: Towards role mining with restricted user-role assignment. In: Wireless VITAE 2011, pp. 1–5 (2011)
10. John, J.C., Sural, S., Atluri, V., Vaidya, J.S.: Role mining under role-usage cardinality constraint. In: Gritzalis, D., Furnell, S., Theoharidou, M. (eds.) SEC 2012. IAICT, vol. 376, pp. 150–161. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30436-1_13
11. Kumar, R., Sural, S., Gupta, A.: Mining RBAC roles under cardinality constraint. In: Jha, S., Mathuria, A. (eds.) ICISS 2010. LNCS, vol. 6503, pp. 171–185. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17714-9_13
12. Mitra, B., Sural, S., Vaidya, J., Atluri, V.: A survey of role mining. ACM Comput. Surv. **48**, 4 (2016)
13. Lu, H., Vaidya, J., Atluri, V.: Optimal boolean matrix decomposition: application to role engineering. In: ICDE 2008, pp. 297–306 (2008)
14. Molloy, I., et al.: Mining roles with semantic meanings. In: SACMAT 2008, pp. 21–30. ACM (2008)
15. Molloy, I., Li, N., Li, T., Mao, Z., Wang, Q., Lobo, J.: Evaluating role mining algorithms. In: SACMAT, pp. 95–104. ACM (2009)
16. Sandhu, R., Ferraiolo, D., Kuhn, R.: The NIST model for role-based access control: towards a unified standard. In: Proceedings of the Fifth ACM Workshop on Role-based Access Control, RBAC 2000, New York, NY, USA, pp. 47–63. ACM (2000)
17. Vaidya, J., Atluri, V., Guo, Q.: The role mining problem: finding a minimal descriptive set of roles. In: SACMAT 2007, pp. 175–184. ACM (2007)
18. Vaidya, J., Atluri, V., Guo, Q.: The role mining problem: a formal perspective. ACM Trans. Inf. Syst. Secur. **13**(3), 27 (2010)
19. Vaidya, J., Atluri, V., Warner, J.: Roleminer: mining roles using subset enumeration. In: CCS 2006, pp. 144–153. ACM (2006)
20. Young, N.E.: Greedy set-cover algorithms. In: Kao, M.-Y. (ed.) Encyclopedia of Algorithms, pp. 886–889. Springer, Heidelberg (2016). <https://doi.org/10.1007/978-3-642-27848-8>



Linguistic Features to Identify Extreme Opinions: An Empirical Study

Sattam Almatarneh^(✉) and Pablo Gamallo

Centro Singular de Investigación en Tecnoloxías da Información (CITIUS),
Universidade de Santiago de Compostela, Rúa de Jenaro de la Fuente Domínguez,
15782 Santiago de Compostela, Spain
{sattam.almatarneh,pablo.gamallo}@usc.es

Abstract. Studies in sentiment analysis and opinion mining have examined how different features are effective in polarity classification by making use of positive, negative or neutral values. However, the identification of extreme opinions (most negative and most positive opinions) have overlooked in spite of their wide significance in many applications. In our study, we will combine empirical features (e.g. bag of words, word embeddings, polarity lexicons, and set of textual features) so as to identify extreme opinions and provide a comprehensive analysis of the relative importance of each set of features using hotel reviews.

Keywords: Sentiment analysis · Opinion mining
Linguistic features · Classification · Extreme opinion

1 Introduction

The information revolution is the most prominent feature of this century. The world has become a small village especially with the proliferation of social networking sites where anyone in the world can sell, buy or express their opinion. The vast amount of information on the Internet has become a source of interest for studies, as it offers an excellent opportunity to extract information and organize it according to the need. In the last two decade, an immense number of studies have been carried in the field of opinion mining and sentiment analysis. The main task in Opinion Mining is polarity classification, which occurs when a piece of text stating an opinion is classified into a predefined set of polarity categories (e.g., positive, neutral, negative). Reviews such as “like” versus “dislike” are examples of two-class polarity classification. An unusual way of performing sentiment analysis is to detect and classify extreme opinions, which represent the most negative and most positive opinions about a topic, an object or an individual. An extreme opinion is the worst or the best view, judgment, or appraisal formed in ones mind about a particular matter.

One of the main motivations for detecting extreme opinions is the fact that they actually stand for *pure* positive and negative opinions. As rating systems have no clear borderlines on a continuum scale, weakly polarized opinions

(e.g. those rated as 4 and 2 in a 1 to 5 rating system) may be in fact closer to neutral statements. According to Pang and Lee [11], “it is quite difficult to properly calibrate different authors’ scales, since the same number of *stars* even within what is ostensibly the same rating system can mean different things for different authors”. Given that rating systems are defined on a subjective scale, only extreme opinions can be seen as natural, transparent, and non ambiguous positive or negative statements. Extreme opinions only constitute a small portion of the opinions on Social Media. According to [11], only about 5% of all opinions are on the most extreme points of a scale, which makes the search for these opinions a challenge. We are then confronted with a challenging task.

It is not surprising that extreme views have a strong impact on product sales, since they influence customer decisions before buying. Previous studies analyzed this relationship, such as the experiments reported in [8], which found that as the high proportion of negative online consumer reviews increased, the consumer’s negative attitudes also increased. Another motivation for the identification of extreme opinions is the current use of bot technology by cyborgs on social networks. These bots are designed to sell products or attract clicks, amplifying false or biased stories in order to influence public opinion.

The main objective of this article is to examine the effectiveness and limitations of different linguistic features to identify extreme opinions in the hotels’ reviews. Our main contribution is to report an extensive set of experiments aimed to evaluate the relative effectiveness of different linguistic features for two binary classification tasks:

- very negative *vs.* not very negative opinions
- very positive *vs.* not very positive opinions

The rest of the paper is organized as follows. In the following Sect. 2, we describe the related work. Then, Sect. 3 describes the method. Experiments are introduced in Sect. 4, where we also describe the evaluation and discuss the results. We draw the conclusions and future work in Sect. 6.

2 Related Work

There are two main approaches to find the sentiment polarity at a document level. First, machine learning techniques based on training corpora annotated with polarity information and, second, strategies based on polarity lexicons. The success of both methods mainly depends on the choice and extraction of the proper set of features used to identify sentiments. There is a great number of surveys and books in sentiment analysis describing the main methods and comparing the usefulness of different linguistic and textual features. For instance, the most salient linguistic features for sentiment classification are listed in Chapter 3 of [9] book. [4] presented a systematic study of different sentence features for two tasks in sentiment classification: namely, polarity classification and subjectivity classification. [7] introduced a new approach to build fixed length vectors for paragraph, sentence, and document representation. [17] proposed an approach

to find the polarity of reviews by converting text into numeric matrices using *countvectorizer* and TF-IDF, and then using them as input in machine learning algorithms for classification. Moreover, sentiment words are the core component in opinion mining and have been used in many studies. [10] built a lexicon containing a combination of sentiment polarity (positive, negative) with one of eight possible emotion classes (anger, anticipation, disgust, fear, joy, sadness, surprise, trust) for each word. As far as we know, excerpted of our previous studies [2,3] no previous work has been focused on detecting extreme opinions. Our proposal, therefore, may be considered to be the first step in that direction.

3 Method

We deal with two document-level binary classification tasks: (1) very negative *vs.* not very negative, and (2) very positive *vs.* not very positive. These tasks can be achieved by automatic classifiers composed of training data in a supervised strategy. The characteristics of documents will be encoded as features in vector representation. These vectors and the corresponding labels feed the classifiers. In the experiments described later, we will examine the following sets of features:

- **N-grams Features:** We deal with n-grams based on the occurrence of unigrams and bigrams of words in the document. Unigrams (1g) and bigrams (2g) are valuable to detect specific domain-dependent (opinionated) expressions. The influence of this type of content features has been confirmed by several opinion mining studies [12,19]. We assign a weight to all terms by using two representations: Term Frequency-Inverse Document Frequency (TF-IDF) and CountVectorizer. TF-IDF is computed in Eq. 1.

$$tf/idf_{t,d} = (1 + \log(tf_{t,d})) \times \log\left(\frac{N}{df_t}\right). \quad (1)$$

where $tf_{t,d}$ in the term frequency of the term t in the document d , N is the number of documents in the collection and, df_t is the number of documents in the collection containing t . CountVectorizer transforms the document to token count matrix. First, it tokenizes the document and according to a number of occurrences of each token, a sparse matrix is created. In order to create the Matrix, all stop words are removed from the document collection. Then, the vocabulary is cleaned up by eliminating those terms appearing in less than 4 documents to eliminate those terms that are too infrequent. To convert the reviews to a matrix of TF-IDF features and to a matrix of token occurrences, we used sklearn feature extraction python library.¹

- **Doc2Vec:** We used the Doc2vec algorithm introduced in [7] to represent the reviews. This neural-based representation has been shown to be efficient when dealing with high-dimensional and sparse data [5,7]. Doc2vec learns features

¹ http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html.

from the corpus in an unsupervised manner and provides a fixed-length feature vector as output. Then, the output is fed into a machine-learning classifier. We used a freely available implementation of the doc2vec algorithm included in gensim,² which is a free Python library. The implementation of the doc2vec algorithm requires the number of features to be returned (length of the vector). So, we performed a grid search over the fixed vector length 100.

- **Set of Textual Features (SOTF):** Many textual features may be used as evidences to detect extreme views: both very positive or very negative alike. In this study, we have extracted some of them to examine to what extent they influence the identification of extreme views. Uppercase characters may indicate that the writer is very upset or affected, so we counted the number of words written in uppercase letters. Also, intensifier words could be a reliable indicator of the existence of extreme views. So, we considered words such as mostly, hardly, almost, fairly, really, completely, definitely, absolutely, highly, awfully, extremely, amazingly, fully, and so on. Furthermore, we took into account negation words such as no, not, none, nobody, nothing, neither, nowhere, never, etc. In addition, we also considered elongated words and repeated punctuation such as (*sooooo, baaaaaad, woooow, goood, ???, !!!!,...etc*). These textual features have been shown to be effective in many studies related to polarity classification such as [6,16].
- **Sentiment Lexicons:** Sentiment words also called opinion words are considered the primary building block in sentiment analysis as it is an essential resource for most sentiment analysis algorithms, and the first indicator to express positive or negative opinions. In our previous studies, we described a strategy to build sentiment lexicons from corpora [1,3]. In this study, we used the same method to create two lexicons of the most negative words and another one for the most positive for hotels domain. VERY-NEG is a lexicon made up of words classified as MN or NMN, while VERY-POS is another lexicon consisting of words classified as MP or NMP³. The new sentiment lexicons for hotels were built from the text corpora introduced in [14,15]. The corpora⁴ consist of online reviews collected from IMDB, Goodreads, OpenTable and Amazon/Tripadvisor. We only use the hotels and restaurants reviews from OpenTable and Tripadvisor. As shown in Table 1, we included lexicon-based features in the two classification tasks as follows. For MN *vs* NMN We represented the number of MN and the number of NMN terms in the document. We also included the proportion of MN and NMN terms. And the same way for the second classification task (MP *vs* NMP) We represented the number of MP and the number of NMP terms in the document. We also included the proportion of MP and NMN terms.

Table 1 summarizes all the features introduced above with a brief description for each one.

² <https://radimrehurek.com/gensim/>.

³ <https://github.com/citiususc/VERY-NEG-and-VERY-POS-Lexicons>.

⁴ <http://www.stanford.edu/~cgpotts/data/wordnetscales/>.

Table 1. Description of all the considered linguistic features in order to identify the most negative opinions (MN vs. NMN) and the most positive opinions (MP vs. NMP)

Features	Descriptions
N-grams	Unigram TF-IDF(1g) Unigram CountVectorizer(1g) Unigram and Bigram TF-IDF (1g 2g) Unigram and Bigram CountVectorizer (1g 2g)
Doc2Vec (100 feat.)	Generate vectors for the document
SOTF (8 feat.)	Number and proportion of negation words in the document Number and proportion of uppercase words in the document Number and proportion of elongated words and punctuations in the document Number and proportion of intensifiers words in the document
<i>VERY-NEG</i> (4 feat.)	Number and proportion of MN terms in the documents Number and proportion of NMN terms in the documents
<i>VERY-POS</i> (4 feat.)	Number and proportion of MP terms in the documents Number and proportion of NMP terms in the documents

4 Experiments

4.1 Data collection

In order to extract extreme opinions, we require to analyze document collections with scaled opinion levels (e.g. rating) and extract those documents associated with the lowest and highest scale. We obtained our dataset from Expedia crowd-sourced data. The HotelExpedia dataset⁵ originally contains 6030 hotels and 381941 reviews from 11 different hotel locations. The datasets are cleaned and prepared for analysis by applying the following three preprocessing steps: (1) data deduplication operation is performed in order to remove such duplicate reviews; (2) 3-stars reviews were deleted since they tend to contain neutral views; (3) all reviews containing less than three words and blank reviews were also removed. After the above three data cleansing operations, the final datasets consists of 20,000 reviews, being 5,000 for each category: 1, 2, 4 and 5 stars.

⁵ <http://ave.dee.isep.ipp.pt/~1080560/ExpediaDataSet.7z>.

4.2 Training and Test

Since we are facing a text classification problem, any existing supervised learning method can be applied. Support vector machines (SVMs) have been shown to be highly effective at traditional text categorization [12]. We decided to utilize *scikit*⁶ which is an open source machine learning library for Python programming language [13]. This library implements several classifiers, including regression and clustering algorithms. We chose SVMs as our classifier for all experiments, hence, in this study we will only summarize and discuss results for this learning model. The dataset was randomly partitioned into training (75 %) and test (25 %). In our analysis, we employed 5_fold cross_validation and the effort was put on optimizing F1 which is computed with respect to MN and MP (which is the target class). We also measured statistical significance with a paired, two-sided micro sign test [18]. This is a statistical method to test for consistent differences between pairs of observations based on their binary decisions on all the document/category pairs, and it applies the Binomial distribution to compute the p-values under the null hypothesis of equal performance.

5 Result

Table 2 shows the performance of very negative classification (MN vs. NMN) performed on our data collection. In these experiments, we combine each n-gram model with the rest of features. The n-gram models are unigrams (1g) and unigrams with bigrams (1g 2g), each one weighted with TF-IDF and CountVector. These models were considered as baselines. Then, combined each baseline with one of the rest of features: namely, Doc2vec, SOTF, *VERY-NEG*, (see Table 1). Moreover, we also combined all features with each baseline (All).

In Table 2, we also report the performance of very positive classification (MP vs. NMP) on our dataset. As we did with the most negative classification, n-gram-based classifiers were regarded as baselines, and we examined the association of various combinations of features into the baseline classifiers, including configurations combining all features.

The results depicted by Table 2 show the following trends. Concerning the classification of not very extreme opinions (NMN and NMP), the baseline approaches are already very accurate and, so, the use of the rest of features does not provide any significant improvement. By contrast, the classification of very extreme opinions is a more tough task in which the baselines are outperformed by some of the other features we have tested. The last column in both tables shows the significant differences concerning only MN and MP classifications. So, significant tests are shown for classification of extreme opinions. In the case of not extreme opinions, there are no significant improvements when we combine different features.

To detect extreme opinions (both very negative and very positive), the most valuable features are textual features (SOTF) and embeddings (Doc2Vec). However, Doc2Vec is more beneficial to detect the very negative reviews, while SOTF

⁶ <http://scikit-learn.org/stable/>.

Table 2. Polarity classification results, in terms of precision, recall, and F1 scores of (MN Vs. NMN) and (MP Vs. NMP). For each n-gram-based model the best performance for each metric is in bold. The symbol “ \gg ” and “ \ll ” indicates a significant improvement with respect to the n-gram-based baselines, with p-value ≤ 0.01 . The symbol “ $>$ ” or “ $<$ ” means that the $0.01 < \text{p-value} \leq 0.05$. “ \sim ” indicate that the difference was not statistically significant (p-value $> .05$).

Features	MN			NMN			s-test	MP			NMP			s-test
	P	R	F1	P	R	F1		P	R	F1	P	R	F1	
1g(TF-IDF)	0.75	0.64	0.69	0.89	0.93	0.91		0.87	0.83	0.85	0.94	0.96	0.95	
+ Doc2Vec	0.77	0.70	0.73	0.91	0.93	0.92	\gg	0.89	0.85	0.87	0.95	0.96	0.96	$>$
+ SOTF	0.76	0.66	0.71	0.90	0.93	0.91	$>$	0.88	0.87	0.87	0.95	0.96	0.96	\gg
+ <i>VERY-NEG</i>	0.76	0.65	0.70	0.89	0.93	0.91	\sim	0.87	0.83	0.85	0.94	0.96	0.95	\sim
+ All	0.78	0.72	0.75	0.91	0.93	0.92	\gg	0.89	0.87	0.88	0.96	0.96	0.96	\gg
1g(CountVector)	0.67	0.66	0.66	0.89	0.90	0.89		0.81	0.79	0.80	0.93	0.93	0.93	
+ Doc2Vec	0.72	0.70	0.71	0.91	0.91	0.91	\gg	0.85	0.84	0.85	0.95	0.95	0.95	\gg
+ SOTF	0.68	0.68	0.68	0.90	0.90	0.90	$>$	0.84	0.83	0.83	0.94	0.94	0.94	\gg
+ <i>VERY-NEG</i>	0.68	0.67	0.67	0.89	0.90	0.90	\sim	0.82	0.80	0.81	0.93	0.94	0.94	$>$
+ All	0.74	0.71	0.73	0.91	0.92	0.91	\gg	0.86	0.84	0.85	0.94	0.95	0.95	\gg
1g 2g(TF-IDF)	0.77	0.62	0.69	0.89	0.94	0.91		0.88	0.84	0.86	0.94	0.96	0.95	
+ Doc2Vec	0.79	0.69	0.74	0.90	0.94	0.92	\gg	0.89	0.86	0.87	0.95	0.96	0.96	\sim
+ SOTF	0.78	0.64	0.70	0.89	0.94	0.92	$>$	0.88	0.87	0.88	0.96	0.96	0.96	\gg
+ <i>VERY-NEG</i>	0.68	0.73	0.70	0.89	0.94	0.91	\sim	0.88	0.84	0.86	0.95	0.96	0.95	\sim
+ All	0.81	0.72	0.76	0.91	0.94	0.93	\gg	0.91	0.89	0.90	0.96	0.97	0.97	\gg
1g 2g(CountVector)	0.69	0.65	0.67	0.89	0.91	0.90		0.83	0.81	0.82	0.93	0.94	0.94	
+ Doc2Vec	0.75	0.70	0.73	0.91	0.93	0.92	\gg	0.86	0.85	0.86	0.95	0.95	0.95	\gg
+ SOTF	0.71	0.67	0.69	0.90	0.91	0.90	\gg	0.86	0.84	0.85	0.94	0.95	0.95	\gg
+ <i>VERY-NEG</i>	0.71	0.66	0.68	0.89	0.91	0.90	\sim	0.84	0.81	0.82	0.94	0.94	0.94	\sim
+ All	0.71	0.68	0.69	0.90	0.91	0.91	$>$	0.89	0.88	0.88	0.96	0.96	0.96	\gg

performs better with the very positive ones. Both types of features leads to statistically significant improvements when they are combined with the baselines (n-gram representations). This confirms the valuable information provided by Doc2Vec and SOTF to detect the most extreme reviews. Lexicon-based features slightly improves the baselines but not in a significant way.

Besides, in all cases the combination of all features always yield significant improvements with regard to the baselines. Finally, it is worth noting that none of the features hurts the overall performance.

6 Conclusions

In this article, we have studied different linguistic features for a particular task in Sentiment Analysis. More precisely, we examined the performance of these features within supervised learning methods (using Support Vector Machine (SVM)), to identify extreme opinions on reviews dataset of hotels. The experiments we carried out showed that n-gram models are difficult to outperform, but we found two features that consistently outperforms the baselines: neural-based embeddings and textual features. Polarity lexicons help improve the results, but their influence is moderate. In future work, we will try to compare unsupervised

method based to polarity lexicons with the supervised classification described in the current paper.

Acknowledgements. This work has received financial support from TelePares (MINECO, ref:FFI201 4-51978-C2-1-R), and the Consellería de Cultura, Educación e Ordenación Universitaria (accreditation 2016-2019, ED431G/08) and the European Regional Development Fund (ERDF).

References

1. Almatarneh, S., Gamallo, P.: Automatic construction of domain-specific sentiment lexicons for polarity classification. In: De la Prieta, F., et al. (eds.) PAAMS 2017. AISC, vol. 619, pp. 175–182. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-61578-3_17
2. Almatarneh, S., Gamallo, P.: Searching for the most negative opinions. In: Rózewski, P., Lange, C. (eds.) KESW 2017. CCIS, vol. 786, pp. 14–22. Springer, Cham (2017). <https://doi.org/10.1007/978-3-319-69548-2>
3. Almatarneh, S., Gamallo, P.: A lexicon based method to search for extreme opinions. *PloS ONE* **13**(5), e0197816 (2018)
4. Chenlo, J.M., Losada, D.E.: An empirical study of sentence features for subjectivity and polarity classification. *Inf. Sci.* **280**, 275–288 (2014)
5. Dai, A.M., Olah, C., Le, Q.V.: Document embedding with paragraph vectors. arXiv preprint [arXiv:1507.07998](https://arxiv.org/abs/1507.07998) (2015)
6. Kennedy, A., Inkpen, D.: Sentiment classification of movie reviews using contextual valence shifters. *Comput. Intell.* **22**(2), 110–125 (2006)
7. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)
8. Lee, J., Park, D.H., Han, I.: The effect of negative online consumer reviews on product attitude: an information processing view. *Electron. Commer. Res. Appl.* **7**(3), 341–352 (2008)
9. Liu, B.: *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge University Press, Cambridge (2015)
10. Mohammad, S.M., Turney, P.D.: Crowdsourcing a word-emotion association lexicon. *Comput. Intell.* **29**(3), 436–465 (2013)
11. Pang, B., Lee, L.: Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pp. 115–124. Association for Computational Linguistics (2005)
12. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: sentiment classification using machine learning techniques. In: Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing, vol. 10, pp. 79–86. Association for Computational Linguistics (2002)
13. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**(Oct), 2825–2830 (2011)
14. Potts, C.: On the negativity of negation. *Semant. Linguist. Theory* **20**, 636–659 (2010)
15. Potts, C.: Developing adjective scales from user-supplied textual metadata. In: NSF Workshop on Restructuring Adjectives in WordNet, Arlington, VA (2011)

16. Taboada, M., Brooke, J., Tofiloski, M., Voll, K., Stede, M.: Lexicon-based methods for sentiment analysis. *Comput. Linguist.* **37**(2), 267–307 (2011)
17. Tripathy, A., Agrawal, A., Rath, S.K.: Classification of sentiment reviews using n-gram machine learning approach. *Expert Syst. Appl.* **57**, 117–126 (2016)
18. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 42–49. ACM (1999)
19. Zhang, Z., Ye, Q., Zhang, Z., Li, Y.: Sentiment classification of internet restaurant reviews written in Cantonese. *Expert Syst. Appl.* **38**(6), 7674–7682 (2011)



Retinal Image Synthesis for Glaucoma Assessment Using DCGAN and VAE Models

Andres Diaz-Pinto^{1(✉)}, Adrián Colomer¹, Valery Naranjo¹, Sandra Morales¹, Yanwu Xu², and Alejandro F. Frangi³

¹ Instituto de Investigación e Innovación en Bioingeniería, I3B, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain
andiapin@upv.es

² Guangzhou Shiyuan Electronics Co., Ltd. (CVTE), Guangzhou 510670, China

³ CISTIB, University of Sheffield, Sheffield S1 3JD, UK

Abstract. The performance of a Glaucoma assessment system is highly affected by the number of labelled images used during the training stage. However, labelled images are often scarce or costly to obtain. In this paper, we address the problem of synthesising retinal fundus images by training a Variational Autoencoder and an adversarial model on 2357 retinal images. The innovation of this approach is in synthesising retinal images without using previous vessel segmentation from a separate method, which makes this system completely independent. The obtained models are image synthesizers capable of generating any amount of cropped retinal images from a simple normal distribution. Furthermore, more images were used for training than any other work in the literature. Synthetic images were qualitatively evaluated by 10 clinical experts and their consistency were estimated by measuring the proportion of pixels corresponding to the anatomical structures around the optic disc. Moreover, we calculated the mean-squared error between the average 2D-histogram of synthetic and real images, obtaining a small difference of 3×10^{-4} . Further analysis of the latent space and cup size of the images was performed by measuring the Cup/Disc ratio of synthetic images using a state-of-the-art method. The results obtained from this analysis and the qualitative and quantitative evaluation demonstrate that the synthesised images are anatomically consistent and the system is a promising step towards a model capable of generating labelled images.

Keywords: Medical imaging · Retinal image synthesis
Fundus images · DCGAN · VAE

1 Introduction

Glaucoma is an irreversible eye disease mainly characterised by optic nerve fibre loss. This loss is given by the increased intraocular pressure (IOP) and/or loss

of blood flow to the optic nerve. In a fundus image, the optic nerve head or optic disc can be visually separated into two zones, a bright and central zone called optic cup and a peripheral part called neuro-retinal rim. See Fig. 1(a).

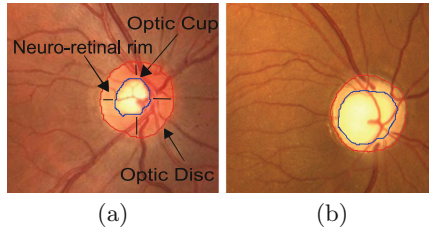


Fig. 1. Digital fundus images cropped around optic disc. (a) Main structures of a healthy optic disc and (b) glaucomatous optic disc.

While the optic disc (OD) and cup are present in all individuals, an abnormal size of the cup with respect to the optic disc is a characteristic of a glaucomatous eye, as it is shown in Fig. 1(b). A deep understanding of the anatomy of the optic disc is crucial for glaucoma understanding. For that reason, different approaches have been developed towards optic disc analysis for Glaucoma assessment using retinal images. For instance, in a state-of-the-art method developed by Chen et al. [1], they used cropped images to train and evaluate their system and obtaining an area under the ROC curve of 0.831 on a database of 650 images.

However, the amount of available images is a huge problem when trying to generalise. For that reason, retinal image synthesizer has been a focus of the scientific community. For instance, in [2] Fiorini et al. used a system that generated the retinal background and the fovea and another system to generate the optic disc by using a large dictionary of patches with no vessels that are later registered. After that, the authors developed a complementary work that is mainly focused on vessel generation [3]. Although their method allows the generation of high-quality and large resolution images, the process of concatenating the generation of the main parts of the images is a considerable complex computational algorithm that relies on how well the images are registered.

Another approach in retinal image synthesis is the one developed by Costa et al. [4]. In their work, they used a method trained on vessel networks and their corresponding retinal fundus images. In other words, they learn a transformation between the vessel trees and the retinal fundus. The main limitation of their method is the dependency of an independent algorithm to segment the vessels.

In another paper, Costa et al. present a method which improves their previous work. Instead of learning a transformation between the vessel trees and the corresponding retinal image, the authors used the original vessel trees to train an autoencoder. Then, the synthetic vessel trees are used as input to the retinal image synthesizer [5].

Although the latter system proposed by Costa et al. is a substantial improvement in their previous work, both methods are dependent on how well the independent method extracts the vessels. The quality of the segmented vessel tree will affect the synthetic vessel trees and then, the final retinal image.

In this paper, we are mainly focused on developing image synthesizers of retinal fundus images. In contrast to previous works, this novel approach does not need the vessel masks and used more images in the training stage. We trained two well-known image generators: The Variational Autoencoder (VAE) [6] and the Deep Convolutional Generative Adversarial Networks (DCGAN) [7] using 2357 images cropped around the optic disc. After that, we used these models to generate synthetic retinal samples to finally evaluate them. Ten clinical experts checked the quality and global consistency of the generated images. Moreover, we compared the structural properties of synthetic and real images by measuring the proportions of the area occupied by the vessel network and optic disc. The consistency in colour terms between the synthetic and real images is also measured by extracting the 2D-histogram (or chromaticity diagram) and computing the mean-squared error.

2 Materials and Methods

2.1 Materials

A total of 2357 images from five public glaucoma-labelled databases: HRF [8] (45 images), Drishti-GS1 [9] (101 images), ORIGA-light [10] (650 images) RIM-ONE [11] (455 images) and sjchoi86-HRF [12] database (401 images) and a private database, ACRIMA (705 images), were used to train the generative models used in this work. All images were manually cropped around the optic disc by an expert, with the exception of RIM-ONE images that are already cropped.

For all the experiments carried out in this work, the open source Deep Learning library Keras [13] and NVIDIA Titan Xp GPU were used.

2.2 Variational Autoencoder

The Variational Autoencoder is composed by two neural networks: the approximate inference network (or encoder), that maps a training example to a latent (hidden) space, and the decoder network that maps from the latent space to a synthetic sample. In this work, we used the architecture proposed in [6], in which the prior over the latent space is a centred isotropic multivariate Gaussian, and the encoder and decoder are fully-connected neural networks with a single hidden layer.

During training or learning phase, the encoder obtains the latent variables z from the input data and the decoder draws those variables to generate a sample. After that, during the generation phase, VAE draws samples from the latent space that run through the decoder to finally obtain a synthetic sample. The VAE architecture can be seen from the Fig. 2(a).

2.3 Generative Adversarial Network

Generative Adversarial Networks, or GAN, are deep neural net architectures comprised of two nets. One is called the generator and the other (the adversary) is called the discriminator.

A class of CNN called Deep Convolutional Generative Adversarial Networks (DCGAN) that are based on the adversarial strategy was used for this work. This architecture was a major improvement on the first GAN, generating better quality images and more stability during the training stage. Following the guidelines to construct the generator and discriminator, described in the paper written by Radford et al. [7], we implemented and trained them on cropped retinal images using the original discriminator and generator cost functions.

In the same way, as in the VAE approach, synthetic image generation using the DCGAN mainly consists of two phases: a learning phase and generation phase. For the training phase, the generator draws samples from an N-dimension normal distribution that run through the generator to obtain a synthetic sample and the discriminator attempts to distinguish between images drawn from the generator and images from the training set. A figure of a DCGAN architecture can be seen from Fig. 2(b).

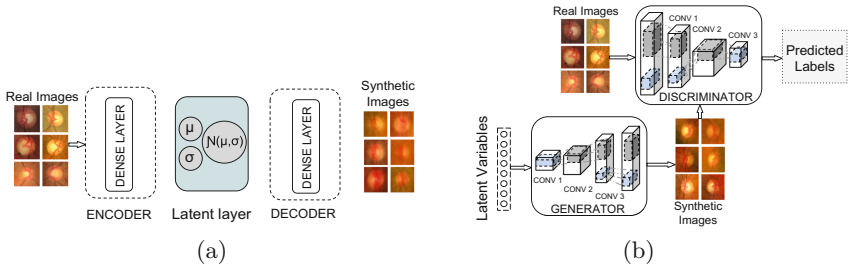


Fig. 2. Schema of the VAE and DCGAN architecture. (a) VAE architecture and (b) DCGAN architecture.

3 Experimental Evaluation

Although a great effort to develop objective metrics that correlate with perceived quality measurement has been made in recent years, it is still a challenging task. In the case of quality evaluation of synthetic images, it should be specific for each application [14]. For that reason, we created a database composed of 200 images: 100 synthetic images and 100 real images (randomly selected from the training set) to perform a qualitative and quantitative evaluation of our methods. This database was analysed by clinical experts with the aim of evaluating the anatomical consistency and plausibility of the synthetic images. The percentage of pixels that composed the vessels and the percentage of pixels that composed

the optic disc were also compared between the synthetic and real images. To obtain these percentages, optic disc masks were manually segmented by clinical experts and the vessel masks were automatically segmented using the method proposed in [15]. Moreover, an averaged chromaticity diagram per class (real and synthetic) was computed with the aim of evaluating the colour properties of the images. The mean-squared error between the averaged histograms and the individual chromaticity diagram of each sample was measured.

4 Results and Discussion

We trained the VAE and DCGAN architectures on cropped retinal images from six different databases without using data augmentation. In order to keep a trade-off between performance and system complexity, the images were automatically re-scaled to the following resolutions: 28×28 pix, 56×56 pix, 112×112 pix and 224×224 pix. For each image size, we tested a range of N-dimensional latent spaces from 32 to 100 latent variables. Each latent space was explored in order to check that the systems do not memorise the training database and, at the same time, it generates plausible retinal images. To do that, we used spherical interpolation to evaluate intermediate latent representation points [16].

For training the VAE model, we ran several tests and found out that the best results are obtained when using a 100-dimension latent space and image resolution of 28×28 and 56×56 pix. Running for 500 epochs and a small batch size of 64, we obtained the synthetic images presented in Fig. 3.

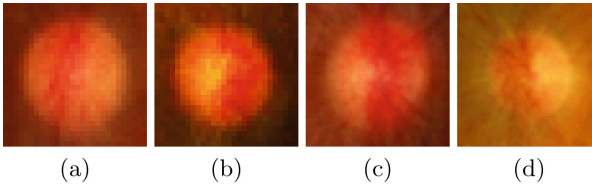


Fig. 3. Examples of images synthesised by the VAE architecture. (a–b) Images of 28×28 pix and (c–d) images of 56×56 pix.

Although the texture of the synthetic images obtained from VAE is similar to the real images, they are blurry and do not have the expected features in a fundus image. For that reason, we only trained on the resolution 28×28 and 56×56 pix.

Regarding the DCGAN architecture, we found that realistic images were obtained when using an image size of 224×224 pix, a small batch size of 32 and 35000 steps. Examples of them are shown in Fig. 4(d–f).

The main advantage of using the DCGAN architecture is that synthetic images are sharper than the ones synthesised by the VAE approach. We can see from Fig. 4(d–f) well-defined optic disc shapes, how blood vessels converge

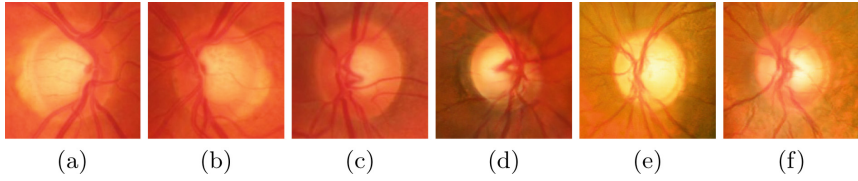


Fig. 4. Examples of real images and examples of synthetic images generated by the DCGAN architecture. (a–c) Real images and (d–f) synthetic images.

into the optic disc and the right and left eye symmetry. For this reason, we continued with the evaluation of only the images synthesised by the DCGAN architecture.

Qualitative evaluation of the database described in Sect. 3 (100 synthetic images and 100 real images) was carried out by ten experts with 3 to 10 years of experience. For each expert, we calculated the Cohen’s kappa coefficient using the ground-truth labels (Fake - Real) and the labels given by each expert. The Cohen’s kappa coefficient ranges from -1 to $+1$, where 0 represents the amount of agreement that can be expected from random chance, and 1 represents a perfect agreement between the ground-truth and the expert. The obtained results are presented in Fig. 5.

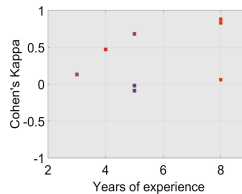


Fig. 5. Qualitative evaluation using Cohen’s Kappa coefficient and years of experience

It can be seen from Fig. 5 that although the Cohen’s Kappa coefficient is high for two experts with high expertise, most of them were fooled when evaluating synthetic images.

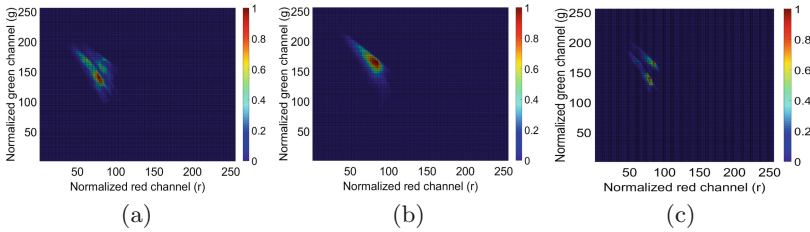
Quantitative evaluation was carried out by measuring the average proportion of pixels belonging to the vessel and optic disc structures. Table 1 shows the obtained results.

It is possible to observe from Table 1 that the mean proportions between synthetic and real images are very similar. The small difference between the mean proportion of the synthetic and real optic discs depends on the normal variation of the optic disc size among real fundus images.

To support the quantitative evaluation and to analyse the similarity between synthetic and real images, we also obtained the average 2D-histogram of real

Table 1. Mean and standard deviation of pixel proportion occupied by the vessels, optic disc and background on the evaluation database.

	Synthetic images	Real images
Vessel proportion	0.1431 ± 0.0306	0.1519 ± 0.0306
Optic disc proportion	0.1776 ± 0.0339	0.2456 ± 0.0722
Background	0.6792 ± 0.0428	0.6025 ± 0.0795

**Fig. 6.** Average 2D-histograms of the synthetic and real images. (a) Average 2D-histogram of real images, (b) average 2D-histogram of synthetic images and (c) mean-squared error between synthetic and real 2D-histogram.

and synthetic images. These 2D-histograms were constructed using the Red and Green channels normalized by the luminance (See Fig. 6).

Moreover, we calculated the mean-squared error between the average 2D-histograms and the chromaticity diagram of each of the 200 images of the database. The obtained results are presented in Table 2.

Table 2. Average and standard deviation of the mean-squared error between the average 2D-histograms and all images.

Average 2D-histogram	Real images	Synthetic images
Real	$0.0028 \pm 3.25 \times 10^{-4}$	$0.0036 \pm 5.43 \times 10^{-4}$
Synthetic	$0.0031 \pm 4.61 \times 10^{-4}$	$0.0022 \pm 5.62 \times 10^{-4}$

The obtained results of this evaluation show that the mean-squared error between synthetic and real images is smaller than the resulting standard deviation among real images (3.25×10^{-4}).

An additional experiment to further analyse the latent space and cup size of the images was performed. We automatically measured the Cup/Disc ratio (CDR) to 1500 synthetic images using the method proposed by Fu et al. [17]. Based on the CDR value, we obtained 743 glaucomatous images when setting the CDR threshold to 0.6 and 344 glaucomatous images when setting the CDR threshold to 0.7.

5 Conclusion

In this paper, two generative models based on the VAE and DCGAN architecture were trained on cropped retinal images from one private and five public databases (2357 retinal images). In contrast to previous approaches that are based on the vessel masks to train their system, the models presented here do not need the vessel masks to synthesise images. Using the DCGAN model, high plausible cropped retinal images were generated and evaluated by clinical experts. Results from this evaluation prove that this initial system is a promising step towards a model capable of generating labelled cropped images.

Acknowledgments. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. This work was supported by the Project GALAHAD [H2020-ICT-2016-2017, 732613].

References

1. Chen, X., Xu, Y., Yan, S., Wong, D.W.K., Wong, T.Y., Liu, J.: Automatic feature learning for glaucoma detection based on deep learning. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 669–677. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_80
2. Fiorini, S., Biasi, M.D., Ballerini, L., Trucco, E., Ruggeri, A.: Automatic generation of synthetic retinal fundus images. In: Smart Tools and Apps for Graphics - Eurographics Italian Chapter Conference. The Eurographics Association (2014)
3. Bonaldi, L., Menti, E., Ballerini, L., Ruggeri, A., Trucco, E.: Automatic generation of synthetic retinal fundus images: vascular network. *Proc. Comput. Sci.* **90**(Suppl. C), 54–60 (2016)
4. Costa, P., et al.: End-to-end adversarial retinal image synthesis. *IEEE Trans. Med. Imaging* **37**(3), 781–791 (2018)
5. Costa, P., et al.: Towards adversarial retinal image synthesis. [arXiv:1701.08974](https://arxiv.org/abs/1701.08974) (2017)
6. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes. [arXiv:1312.6114](https://arxiv.org/abs/1312.6114) (2013)
7. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. [arXiv: 1511.06434](https://arxiv.org/abs/1511.06434), November 2015
8. Köhler, T., Budai, A., Kraus, M.F., Odstrčilík, J., Michelson, G., Hornegger, J.: Automatic no-reference quality assessment for retinal fundus images using vessel segmentation. In: Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems, pp. 95–100 (2013)
9. Sivaswamy, J., Krishnadas, S., Joshi, G.D., Jain, M., Ujjwal, A.S.T.: Drishti-GS: retinal image dataset for optic nerve head (ONH) segmentation. In: 2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI), pp. 53–56 (2014)
10. Zhang, Z., et al.: ORIGA-light: an online retinal fundus image database for glaucoma analysis and research. In: 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology, pp. 3065–3068, August 2010
11. Medina-Mesa, E., et al.: Estimating the amount of hemoglobin in the neuroretinal rim using color images and OCT. *Curr. Eye Res.* **41**(6), 798–805 (2015)

12. sjchoi86: sjchoi86-HRF Database (2017). https://github.com/sjchoi86/retina_dataset/tree/master/dataset. Accessed 02 July 2017
13. Chollet, F., et al.: Keras (2015). <https://github.com/fchollet/keras>. Accessed 21 May 2017
14. Theis, L., van den Oord, A., Bethge, M.: A note on the evaluation of generative models. In: International Conference on Learning Representations, April 2016
15. Morales, S., Naranjo, V., Navea, A., Alcañiz, M.: Computer-aided diagnosis software for hypertensive risk determination through fundus image processing. *IEEE J. Biomed. Health Inform.* **18**(6), 1757–1763 (2014)
16. White, T.: Sampling generative networks. [arXiv:1609.04468](https://arxiv.org/abs/1609.04468) (2016)
17. Fu, H., Cheng, J., Xu, Y., Wong, D.W.K., Liu, J., Cao, X.: Joint optic disc and cup segmentation based on multi-label deep network and polar transformation. *IEEE Trans. Med. Imaging* (2018)



Understanding Learner's Drop-Out in MOOCs

Alya Itani, Laurent Brisson^(✉), and Serge Garlatti

IMT Atlantique, Lab-STICC, UBL, 29238 Brest, France
{alya.itani, laurent.brisson, serge.garlatti}@imt-atlantique.fr
<http://www.imt-atlantique.fr/fr/personne/>

Abstract. This paper focuses on anticipating the drop-out among MOOC learners and helping in the identification of the reasons behind this drop-out. The main reasons are those related to course design and learners behavior, according to the requirements of the MOOC provider OpenClassrooms. Two critical business needs are identified in this context. First, the accurate detection of at-risk droppers, which allows sending automated motivational feedback to prevent learners drop-out. Second, the investigation of possible drop-out reasons, which allows making the necessary personalized interventions. To meet these needs, we present a supervised machine learning based drop-out prediction system that uses *Predictive algorithms* (Random Forest and Gradient Boosting) for automated intervention solutions, and *Explicative algorithms* (Logistic Regression, and Decision Tree) for personalized intervention solutions. The performed experimentations cover three main axes; (1) Implementing an enhanced reliable dropout-prediction system that detects at-risk droppers at different specified instants throughout the course. (2) Introducing and testing the effect of advanced features related to the trajectories of learners' engagement with the course (backward jumps, frequent jumps, inactivity time evolution). (3) Offering a preliminary insight on how to use readable classifiers to help determine possible reasons for drop-out. The findings of the mentioned experimental axes prove the viability of reaching the expected intervention strategies.

Keywords: Learning analytics · Supervised machine learning
Massive Open Online Courses · Modeling drop-out

1 Introduction

Massive Open Online Courses (MOOCs), offer an alternative education method that changed the standards of teaching and learning forever. In the after MOOCs era, education has reformed to become attainable to the whole public at any age, price, country, time, and mean [5]. This elevated ease and unrestricted access to material led to massiveness not only in the scale of participation but also in that of incompleteness, commonly known as drop-out [14]. Consequently, a wide investigation on MOOC drop-out rates was provoked. The prevailing research on that

subject, revolved generally around anticipating drop-out and studying solutions for preventing or decreasing it among learners [10, 16]. Essentially, this is how applications of machine learning techniques for drop-out prediction started taking form. The literature encompasses several intervention strategies for drop-out prevention [3, 7, 8, 11]. Some strategies assert sending automated motivational messages or emails from the prediction system to the spotted learners at-risk. While other strategies assert sending personalized intervention messages either directly to learners or to an intermediary party, usually the teacher. In return, this intermediary, teacher, chooses the necessary intervention to make after analyzing the information offered by the prediction system [7].

OpenClassrooms mainly intend to find the reasons for drop-out among its learners and prevent this drop-out when possible using the appropriate intervention strategy. Therefore, to help OpenClassrooms in meeting their needs, we present a supervised machine learning based drop-out prediction system that uses *Predictive algorithms* (Random Forest and Gradient Boosting) for automated intervention solutions, and *Explicative algorithms* (Logistic Regression, and Decision Tree) for personalized intervention solutions to learners through an intermediary teacher. We summarize our contributions as follows: (1) Proposing a predictive system that can detect at-risk droppers at different instants of the learner's interaction with the course. (2) Introducing and testing new features associated with learners' trajectory of engagement with the course. (3) Deploying the readability of different classifiers to offer suitable intervention strategies for both teachers and learners.

This paper is structured in 6 sections. Section 2 presents related works. Section 3 describes the predictive system. Section 4 presents the experiments and their results. Section 5 offers an exhaustive analysis and discussion of the obtained results. Finally, Sect. 6 concludes the main findings.

2 Related Works

The idea of applying learning analytics on MOOCs emerged with the rise of massive raw data from recorded learners activity on various MOOC platforms [15]. At first, lights were mostly shed on exploring and evaluating MOOCs and their low completion rates; often found to be $\leq 13\%$ [1]. Subsequently, researchers' inquisition started orienting toward understanding this immense drop-out among MOOC learners and its causes. They discovered that reasons for MOOC learners drop-out can be very diverse due to its audience heterogeneity. In that context, Khalil and Ebner [8], Colman [3], and Onah et al. [11] all investigated the reasons of this marked drop-out. The most addressed reasons in the literature can be summed up as follows: (1) Lack of intention to complete (ex: material hunters, curious explorers, assessment hater, etc.) (2) Personal circumstances (ex: lack of time, family situations, etc.) (3) Bad MOOC design (ex: inefficient material, high workload, shortage in organization etc.) (4) Deficiency in digital skills. (5) Inaccurate expectations. (6) Bad prior experience.

The investigation of MOOC drop-out and its reasons opened the horizon towards using machine learning techniques to predict drop-out ahead of time and

try to prevent it. Initially, studies attempted drop-out predictions considering mono-type contextual features, like forum interactions or video restricted events [12, 17]. However, such feature restrictions can restrain the model’s predictive potential. Consequently, multi-type feature based prediction models emerged. Kloft et al. [9] proposed a machine learning algorithm that works on clickstream data and other features to identify learners’ most active time and its effect on drop-out. Still, studies were mostly restricted to one prediction algorithm. Soon after, learning analytics predictive models became more and more advanced with various tested algorithms, proper feature selection testing and evaluation [18]. In this context, Hlosta et al. [6] present an early at-risk identification upon the absence of legacy information for the case of new courses.

3 Drop-Out Predictive System

Figure 1 shows the proposed drop-out predictive system and the analysis process. This system uses the historical traces of learners in a given MOOC to construct models that accurately classify new learners into droppers and completers at some point in their progress. Hence, the prediction target in this problem is of two categorical classes (dropper, completer), and the dataset at hand is a labeled dataset. Therefore, we propose a system based on a supervised machine learning process.

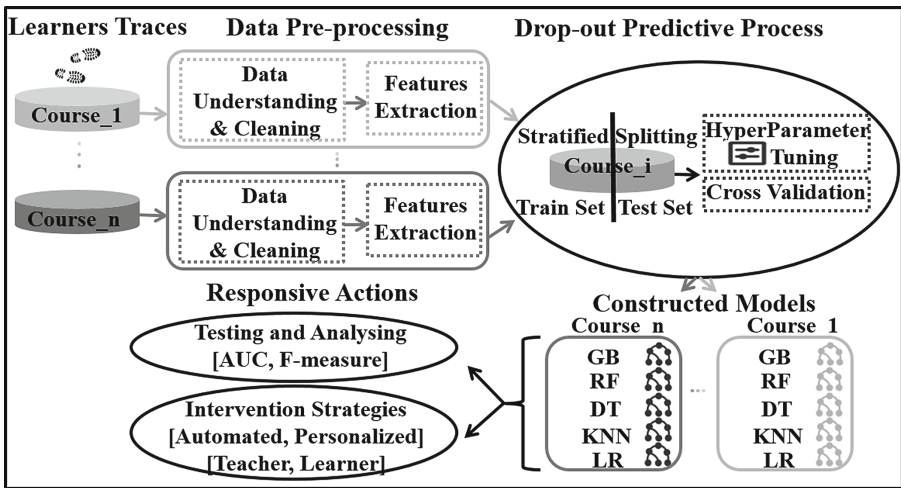


Fig. 1. Phases of the drop-out predictive system

Mainly, the process helps in attaining two goals: (1) Offering accurate predictions for the purpose of automated interventions such as motivational messages to learners (2) offering readable and explainable predictions in order to personalize interventions with learners or improve course structure. Here we describe

the analysis process that we have implemented. Once the system is configured for a MOOC platform it is easy to automate the process.

Data Preparation. In this phase, the collected historical traces undergo the necessary preparatory steps to be used by the classification process. The first step is *data exploration*, which is familiarizing with the data in hold (course structure, number of chapters and activities, important terms like: part, grade, success, etc.). Whereas, the second step is *structure validation and cleaning* of the dataset which is basically fixing any inconsistency in the data (missing entries, redundant entries, duplicates). Lastly, comes the phase of *features selection*, it involves constructing a features matrix to be used as input to the classification process. Typically, the features in this matrix should include a representation of any useful information available in the data.

Classification Process and Models Construction. In this phase, the classification process takes as input the constructed features matrix. We construct efficient predictive models that are optimized on different level of their construction. First, we perform a *Stratified Splitting* of the data. This implies partitioning the final features dataset into a training set (60%) and a testing set (40%) while preserving the initial dataset balance of droppers and completers. The testing test is used to compare the performance obtained by the 4 algorithms tested under different experimental conditions. Second, *Grid Search Hyper-parameter Tuning* is performed by searching exhaustively for the best scoring parameters for each model through a manually specified subset of the hyper-parameter space [19]. Third, to avoid overfitting *K-fold Cross Validation* is applied, by randomly partitioning the data into K subsamples. This action is then repeated K times with measuring the performance on each time and then averaging it at the end.

Responsive Actions. After the construction of the five classification models, the models are tested and evaluated on testing set. Explicative models are analyzed and the obtained information is sent to teachers for investigation.

4 Case Study and Experimentation

In this case-study, we conducted an experiment on a data set from the Open-Classrooms MOOC platform. The interest of the platform's analysts was to predict the drop-out of the platform's premium members by being able to explain the reasons to Mooc's designers. However, one constraint was not to use the demographic and social data of the platform users. We propose advanced features related to learners trajectories of engagement with the course to test their effect on the predictive efficiency and readability of the results.

4.1 The OpenClassrooms Case Study

This dataset includes activity traces of 20,142 premium learners within “*Create your Website with HTML*” (HTML5) and “*Understanding the Web*” (Web) courses from October 2015 till October 2016. OpenClassrooms courses have no sessions or weeks, once a course is posted online it is available for students to start following at any time. Also, there is no maximum duration limit for finishing a course (a learner can take months to finish a course). Each course is divided into chapters and each chapter into parts. At the end of each chapter, there are graded multiple choice or peer assessed exercises. Upon completing all exercises, learners are given a final course grade.

Table 1. Statistics of premium learners population in “*Create your Website with HTML*” (HTML5) and “*Understanding the Web*” (Web) courses

Measures	HTML5	Web
Number of learners	12,114	7,379
Number of active learners	11,520	7,160
Completers (%)	4,333 (37.6%)	5,085 (71%)
Droppers (%)	7,187 (62.4%)	2,075 (29%)

Data Description. The dataset contains subscription related events (following and un-following a course), course related events (visualization events, completions events, grades), and exercise session events. Table 1, describes the distribution of the different types of learners in our data set. We can notice that dropper rates are low compared to the rates generally observed in MOOCs: HTML5 course has a balance of 40% droppers and 60% completers, whereas Web course has 20% droppers and 80% completers. This is because we are only interested here in the premiums members of the platform: the motivation increases when a payment is involved [4].

Features Selection. MOOC designers know that learners rarely navigate the course in its planned linear manner. They rather go back and forth creating back and forward jumps. Therefore, we introduce two types of indicators for features selection: descriptive indicators and behavioral indicators. Descriptive indicators describe learner-course interactions. Whereas, behavioral indicators describe learners trajectories of engagement with the course versus the recommended trajectory of the course. The expected worth of behavioral indicators comes from their ability in revealing the flaws or strengths of the MOOC design and content. Indeed, bad MOOC design or inappropriate MOOC content are considered inevitable reasons of dropping (see Sect. 2). Tables 2 and 3 offer a detailed view on both indicator types.

Table 2. Descriptive indicators and corresponding features

Completed parts of the course	
Definition	An estimate of completed MOOC parts for each learner
Features	Binary features with values 1: Completed Part 0: Uncompleted part A Part can be either a chapter or an exercise
Purpose	Permits studying the effect of parts completion on learner’s drop-out
Exercise scores	
Definition	Incorporates the learners’ scores on each completed exercise
Features	Numeric values of grades, 0 denotes an uncompleted exercise
Purpose	Allows studying the effect of grades on MOOC completion
Time passed on exercises	
Definition	An estimate of the time passed on each completed MOOC exercise
Features	Numeric values of time in seconds, 0 denotes an uncompleted exercise
Purpose	Helps in studying the effect of exercise-invested time on completion

The final features matrix includes 34 mixed type features (numerical and categorical) alongside one categorical binary target variable, where 0 denotes completion and 1 denotes dropping.

4.2 Experimentation

We consider four main aspects upon the evaluation of the proposed drop-out predictive system:

1. The efficiency of prediction at different instants of the course. In other words, the system is tested for classifying learners into either completers or droppers at different points in the course progression, we test on 25% and 50% of activities. If a user skip some sections of the course, corresponding features are marked as uncompleted (see Table 2). All features after 25/50% of the course are left out so the dataset changes slightly between experiments. If a user skip the course before 25/50% of the activities are presented, he is considered as a dropper but is not removed from the dataset.
2. The effect of dynamic behavioral indicators on the predictive system’s efficiency and readability. The system’s performance is tested with behavioral indicators vs. without behavioral indicators.
3. Variety of supervised classification algorithms with hyper-parameter tuning¹. We test two different types. Explicative ones that are simple readable algorithms including Decision Tree (DT) and Logistic Regression (LR). Aggregated ones that have generally better prediction rates, but are more complex to read, including Gradient Boosting (GB) and Random Forest (RF).

¹ You can access the selected parameters for each model after hyper-parameter tuning by consulting the following link: <http://www.laurent-brisson.fr/publication/2018-understanding-learner-dropout-mooc/>.

Table 3. Behavioral indicators and corresponding features

Number of back jumps in a course	
Definition	Number of back jumps performed throughout the course for each learner
Features	Numeric value, number of performed back jumps
Purpose	Allows studying the effect of back jumps on course completion
Most frequent jumps in a course	
Definition	The N most frequent jumps performed by learners in the MOOC
Features	N binary features with values 0: Jump not made, 1: Jump made
Purpose	Study the effect of performing the most frequent jumps on completion
Inactivity time evolution	
Definition	The learner’s evolution of inactivity time between two parts of the course
Features	Numeric value representing a logarithmic scale of time
Purpose	Study the effect of increase or decrease of inactivity on completion

4. Multiple course topics: “*Create your Website with HTML*” (HTML5) and “*Understanding the Web*” (Web).

Tables 4 and 5 compare the performances obtained during the experiments using the F-measure which is a balance between precision and recall [13].

Table 4. “*Create your Website with HTML*” (HTML5) course (35 activities). Performance of 5 classifiers on the F-measure metric.

Percentage of activities completed:		25% (9 activities)		50% (17 activities)	
Behavioral Indicators:		with	without	with	without
Algorithms:	RF	0.76	0.77	0.84	0.85
	GB	0.79	0.77	0.85	0.85
	DT	0.75	0.78	0.85	0.85
	LR	0.74	0.74	0.85	0.85

A first analysis of these results leads us to two findings that we discuss in the next section:

- The low impact of behavioral indicators on the prediction performance (Sect. 5.2).
- The disparate impact, depending on the studied MOOC, of the number of activities completed on the performance of predictions (Sect. 5.3).

Other related research that also address post-hoc learning methods (i.e. earning takes place on the same course) exists in the literature with close marked

Table 5. “Understanding the Web” (Web) course (23 activities). Performance of 5 classifiers on the F-measure metric.

Percentage of activities completed:		25% (6 activities)		50% (12 activities)	
Behavioral Indicators:		with	without	with	without
Algorithms:	RF	0.26	0.26	0.91	0.91
	GB	0.25	0.25	0.91	0.91
	DT	0.25	0.25	0.91	0.91
	LR	0.46	0.46	0.90	0.91

attainments on early drop-out detection. For example, Whitehill et al. [18] obtained a 90.20% AUC in the detection of drop-out averaged over 8 weeks on HawardX MOOCs. In the case of our predictions at 50% of the activities we obtain an AUC around 85% which corresponds to the same order of magnitude even if the use of two different data sets and different variables (they use “*click-streams features* that contains all interaction events between every student and the MOOC courseware”) makes the comparison difficult. Additionally, neural network based methods can overcome our outcome in performance measures, but are out of our research scope and objective. In this paper, our goal does not stop at detecting drop-out we rather seek insights and actionable outcomes to help MOOC providers and stakeholders make the right decisions and understand early drop-out (with 25% of activities carried out).

5 Results Discussion

In this section we analyze the results taking into consideration: (1) the readability of models, (2) the effect of behavioral indicators and (3) the effect of the number of completed activities at the time of prediction.

5.1 Models Readability

One of the objectives of this experiment is to be able to help decision-makers, in this case the MOOC designers, to improve the structure and content of their course. Here we will compare two types of algorithms, predictive algorithms (Random Forest, Gradient Boosting) and explanatory algorithms (Decision Tree, Logistic Regression). By comparing the results obtained in Tables 4 and 5 we can realize that, in our context, purely predictive algorithms do not really have better results than explanatory algorithms. It would be interesting to know whether this is due to the nature of the studied MOOCs or due to a platform effect but this is beyond the scope of this paper.

Predictive Algorithms. Although they are not readable at all, these models can still be used to determine which features have the most influence on dropout prediction. In the case of our experiment, the most discriminating variables were

the scores obtained at the end of chapter exercises. However, these methods do not indicate the direction of this influence (positive or negative) on drop-out, which is very damaging to understand the context of each influencing feature.

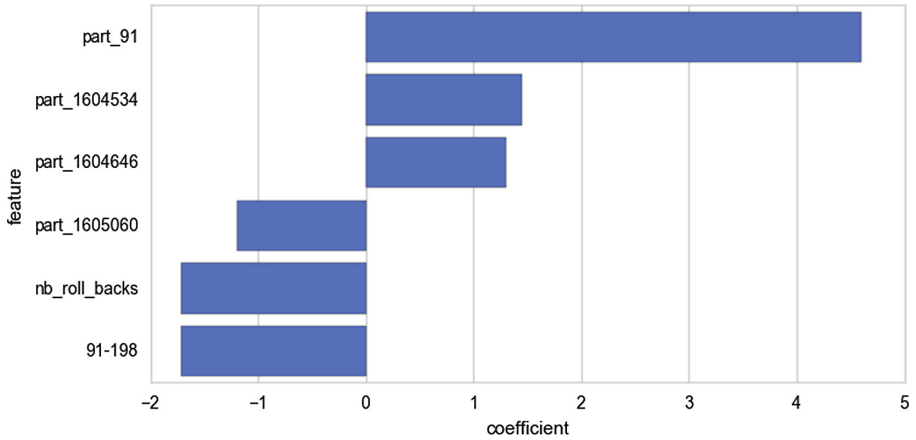


Fig. 2. Logistic regression coefficients for predicting drop-out in “*Create your Website with HTML*” (HTML5) course with 25% of the activities. Display of the 6 most discriminating features.

Explicative Algorithms. Logistic regression is capable of demonstrating the influence of features on drop-out along with the direction of each feature’s influence. Figure 2 shows the model coefficients for the “*Create your Website with HTML*” (HTML5) course after 25% of the activities have been completed. We can observe the negative impact of completing some activities (for example *part_91*) and the positive impact of some jumps (for example *91-198*) on dropout. To understand this phenomenon it is necessary to know that the activity *part_91* corresponds to the last quiz of the first chapter, and that *part_198* is the following activity which consists in making a bibliographic research. Thus, here Logistic Regression shows that the chance of success increases when learners carry out these two activities in their expected linear manner (one after the other).

Decision trees allow us to understand and measure the impact of each feature on the prediction. In Fig. 3 the dropping class nodes are in dark gray and the completion class nodes are in light gray. Each node details the feature condition, total number of samples, dropper samples, completer samples, and %ratio. Furthermore, decision rules can be derived from each tree and can hold important information on the manner droppers behave. For example a rule for HTML5 droppers at 25% of course activities: “If a learner does not jump from *part_91* to *part_198* (which is the recommended progress) \implies he is at risk of dropping with %D = 81.06% (out of 3364 learners)”.

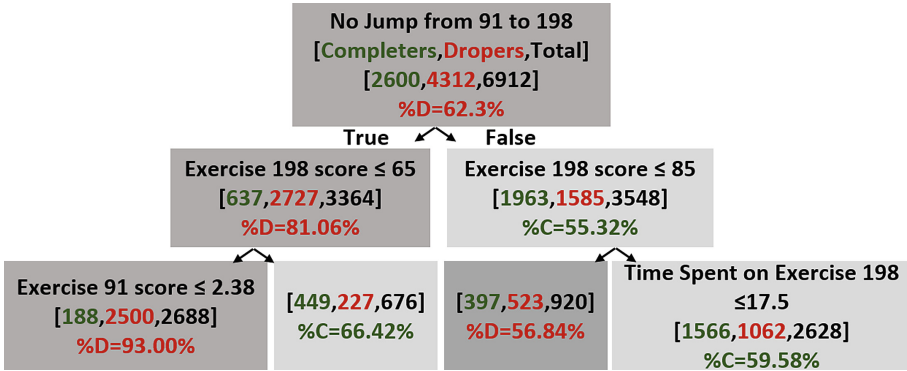


Fig. 3. Decision Tree (pruned after the 2nd branch) for predicting drop-out in “Create your Website with HTML” (HTML5) course with 25% of the activities.

That kind of information can be very helpful in discovering problems related to course material, design, or level of difficulty. However, with inspecting the entire tree, dozens of rules can be derived from each model which is a weakness of decision trees. Here, after pruning, it is easy to observe interesting rules but the trees generated can be very large and become difficult to interpret.

5.2 Effect of Behavioral Indicators

Tables 4 and 5 show the impact of behavioral indicators on the classification results: when these indicators increase the predictive performance of classifiers boxes are grayed out. We can notice that the performance increase is very low for the MOOC “Create your Website with HTML” (HTML5) while it is negligible for the MOOC “Understanding the Web” (Web). However, the results here are not disappointing because they open the doors wide to discussion with the designers of the MOOC. As we saw in Sect. 5.1, behavioral indicators allowed us to understand a phenomenon with the HTML5 course: the risk of drop-out increases with the completion of some activities while it decreases with the completion of a sequence of two activities. This is interpreted by the fact that this course attracts an audience looking for resources (a video explaining a concept, a definition or an exercise to practice) for which there is no commitment in successfully completing the MOOC. We distinguish here 3 types of interesting situations from a business point of view:

- The most discriminating feature is a descriptive indicator: the teacher must ask himself about the relevance of an activity. If it helps in detecting student involvement, this is a good thing. If it identifies a difficulty (related to a tool or concept), an accompanying measure should be put in place.
- The most discriminating feature is a behavioral indicator: this is the ideal case to suggest new routes to students, or to remind students of the importance of following the recommended progression as they move away from it.

- The prediction is bad and so no feature is really relevant: the proposed activities do not anticipate successes and failures. Either there is nothing to do, activities are designed to build student self-confidence, or there is a corrective action to consider to implement constructive alignment [2] if any of the following activities has a high failure rate.

5.3 Effect of Prediction at Different Course Instants (25%, 50%)

Tables 4 and 5 also show the impact of the number of activities considered on the results: we will focus here on predictions at two instants after having carried out 25% then 50% of the activities. Not surprisingly here, the increase in the number of activities considered to make the prediction improves the system's performance. However, what interests us here is the increase in this performance according to the course. For the course “*Create your Website with HTML*” (HTML5), the F-measure increases on average by 8.5 between the two instants (25/50%), while for the course “*Understanding the Web*” (Web) the F-measure increases on average by 61.5, i.e. 7 times more!

This difference can be explained by the presence in the Web course of an activity which takes place between 25% and 50% of their progression. In this example, the quiz in the first chapter was taken into account in the 25% of activities carried out, while the last exercise in the chapter was not included. We illustrate here very well the third situation presented in Sect. 5.2 where the activities carried out are aimed at putting the student in confidence. The fact of making predictions at different stages of completion of activities thus makes it possible to observe a learning dynamic.

6 Conclusion

In this paper we present a supervised machine learning based drop-out prediction system that uses aggregated and explicative type classifiers. The aggregated classifiers can help in accurately detecting at-risk droppers, which allows sending automated motivational feedback to learners. Whereas, explicative classifiers allows the personalized intervention through a teacher. We state the findings according to the three main tested axes: (1) **Readability of explicative models**: Decision Trees and Logistic Regression permit the detailed inspection of the classification process and the effect of features on this classification. They could be hard to interpret by non-experts, but they can be used to send teachers valuable information to analyze and accordingly make personalized interventions. (2) **Dynamic Behavioral Indicators**: Including these indicators enhances slightly the predictive performance of the system, but it noticeably contributes to the readability of the prediction, however their effect depends highly on the studied course and material included. (3) **Prediction at different instants of the course**: the further the instant is, the more material included, the better is the performance of the system. Predicting at different instants can expose activities that are critical for classification. However, a critical activity for classification is

not necessarily a critical pedagogical activity, and the current proposed system rather sheds the light on interesting aspects that can aid teachers in uncovering problems in course design and material.

References

1. Belanger, Y., Thornton, J.: Bioelectricity: a quantitative approach duke university's first MOOC. Technical report (2013)
2. Biggs, J.B.: Teaching for Quality Learning at University: What the Student Does. McGraw-Hill Education (UK), London (2011)
3. Colman, D.: MOOC interrupted: top 10 reasons our readers didn't finish a massive open online course. Open Culture (2013)
4. Devlin, K.: MOOCs and the myths of dropout rates and certification. Huff Post College (2013). Accessed 2 March 2013
5. Emanuel, E.J.: Online education: MOOCs taken by educated few. *Nature* **503**(7476), 342–342 (2013)
6. Hlosta, M., Zdrahal, Z., Zendulka, J.: Ouroboros: early identification of at-risk students without models based on legacy data. In: Proceedings of the Seventh International Learning Analytics & Knowledge Conference, pp. 6–15. ACM (2017)
7. Jayaprakash, S.M., Moody, E.W., Lauría, E.J., Regan, J.R., Baron, J.D.: Early alert of academically at-risk students: an open source analytics initiative. *J. Learn. Anal.* **1**(1), 6–47 (2014)
8. Khalil, H., Ebner, M.: MOOCs completion rates and possible methods to improve retention - a literature review. In: Viteli, J., Leikomaa, M. (eds.) Proceedings of EdMedia + Innovate Learning 2014, Tampere, Finland, pp. 1305–1313. Association for the Advancement of Computing in Education (AACE), June 2014
9. Kloft, M., Stiehler, F., Zheng, Z., Pinkwart, N.: Predicting MOOC dropout over weeks using machine learning methods. In: Proceedings of the EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs, pp. 60–65 (2014)
10. Lackner, E., Ebner, M., Khalil, M.: MOOCs as granular systems: design patterns to foster participant activity (2015). Accessed 10 September 2015
11. Onah, D.F., Sinclair, J., Boyatt, R.: Dropout rates of massive open online courses: behavioural patterns. In: EDULEARN 2014 Proceedings, pp. 5825–5834 (2014)
12. Ramesh, A., Goldwasser, D., Huang, B., Daumé III, H., Getoor, L.: Learning latent engagement patterns of students in online courses. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, pp. 1272–1278. AAAI Press (2014)
13. Rijsbergen, C.J.V.: Information Retrieval, 2nd edn. Butterworth-Heinemann, Newton (1979)
14. Rivard, R.: Measuring the MOOC dropout rate. *Inside High. Ed.* **8**, 2013 (2013)
15. Tabaa, Y., Medouri, A.: LASyM: a learning analytics system for MOOCs. *Int. J. Adv. Comput. Sci. Appl. (IJACSA)* **4**(5) (2013)
16. Wen, M., Yang, D., Rose, C.: Sentiment analysis in MOOC discussion forums: what does it tell us? In: Educational Data Mining 2014 (2014)
17. Wen, M., Yang, D., Rosé, C.P.: Linguistic reflections of student engagement in massive open online courses. In: ICWSM (2014)
18. Whitehill, J., Mohan, K., Seaton, D., Rosen, Y., Tingley, D.: Delving deeper into MOOC student dropout prediction. arXiv preprint [arXiv:1702.06404](https://arxiv.org/abs/1702.06404) (2017)
19. Witten, I.H., Frank, E., Hall, M.A., Pal, C.J.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, Burlington (2016)



Categorical Big Data Processing

Jaime Salvador-Meneses^{1(✉)}, Zoila Ruiz-Chavez¹, and Jose Garcia-Rodriguez²

¹ Universidad Central del Ecuador, Ciudadela Universitaria, Quito, Ecuador
{jsalvador,zruiz}@uce.edu.ec

² Universidad de Alicante, Ap. 99., 03080 Alicante, Spain
jgarcia@dtic.ua.es

Abstract. Prior to the application of a machine learning algorithm, the information has to be stored in memory which may consumes big memory amounts. If we reduce the amount of memory used to represent datasets, we can reduce the number of operations required to process it. All the libraries used to represent the information make a traditional representation (vector, matrix for example), which force to iterate over the whole dataset to obtain a result. In this paper we present a technique to process categorical data that was previously encoded in blocks of arbitrary size, the method process the data block by block which can reduces the number of iterations over the original dataset, and at the same time, the performance is similar to the traditional processing of the data. This method also requires the data to be stored in memory but in an encoded way that optimize the memory size consumed for the representation as well as the operations required to process it. The results of the experiments carried out show a slightly lower time processing than the obtained with traditional implementations, which allows us to obtain a good performance.

Keywords: Big data · Compression · Processing · Categorical data
BLAS

1 Introduction

The dimension (number of attributes or features) in many datasets es large, and many algorithms do not work well with datasets that have a high dimension. Currently, it is a challenge to process data with a high dimensionality such as censuses conducted in different countries [8], another important characteristic of this type of datasets is that the data are encoded as categorical values.

In the last 20 years, Latin America has tended to take greater advantage of census information [7], this information is mostly categorical information (variables that take a reduced set of values). The representation of this information in digital media can be optimized given the categorical nature ans the reduced number of possible values that they can take.

A census consists of a set of observations (also called records) each of which contains a group of attributes. An observation contains the answers to a questionnaire given by all members of a household [1]. We can extend this definition to surveys that are carried out on a regular basis.

This paper proposes a mechanism for processing categorical information using bit-level operations. Prior to processing, it is necessary to encode (compress) the information into a set of fixed blocks. The mechanism proposes compressing the information into packets of a certain number of bits (16, 32, 64 bits), in each packet a certain number of values are stored. As an example, we propose a block size of 32 bits.

Bitwise operations (AND, OR, NOT, etc.) are an important part of modern programming languages because they allow you to replace arithmetic operations (addition, subtraction, etc.) with more efficient operations [12].

This document is organized as follows: Sect. 2 summarizes the standard that defines the algebraic operations that can be performed on data sets as well as some libraries that implement it, Sect. 3 presents an alternative for information processing based on the BLAS Level 1 standard and a mechanism to encode the data, Sect. 4.2 presents several results obtained using the proposed method and, finally, Sect. 5 presents some conclusions and future work.

2 BLAS Standard

In this section we describe the BLAS standard as well some libraries that implements it.

Basic Linear Algebra Subprograms (BLAS) is a specification that defines low-level routines for performing operations related to linear algebra. BLAS define 3 levels but for the purpose of this paper, we only consider the level 1 specification: operations between vectors and scalars.

2.1 BLAS Level 1

Level 1 specifies operations between scalars and vectors and operations between vectors. Table 1 lists some functions described in BLAS Level 1.

Table 1. BLAS level 1 functions

Function	Description	Mathematical formula
Swap vectos	Exchange of two vectors	$y \leftrightarrow x$
Stretch/scale	Vector scaling	$x \leftarrow \alpha x$
Assiggment/copy	Copy a vector	$y \leftarrow x$
Multiply and add	Sum of two vectors	$y \leftarrow \alpha x + y$
Multiply and subtract	Subtraction of two vectors	$y \leftarrow \alpha x - y$
Dot product	Dot product between two vectors	$\alpha \leftarrow x^T y$
L^2 norm	Vector 2-norm (Euclidean norm in \mathbb{R}^2)	$\alpha \leftarrow \ x\ _2$

We will use L^2 norm to show the bock processing method.

2.2 BLAS Implementations

This section describes some of the libraries that implement the BLAS Level 1 specification. All libraries described below make use of vector and matrix operations and therefore require an optimal representation of this type of algebraic structure.

The information is represented as arrangements of native data-types like *float* or *double*, so the size needed to represent a vector is $size * 4$ because the size needed to represent these native types is 4-bytes.

ViennaCL provides a high-level abstraction using C++ data representation on GPU [10]. To work with GPUs, it maintains two approaches, the first based on CUDA¹, while the second based on OpenCL².

If the system running ViennaCL does not support GPU acceleration, it makes use of the multi-processing associated with the core processor (CPU).

ViennaCL can be invoked from other libraries such as Armadillo and uBLAS [9]. The information is represented in a scheme similar to STL³, for the case of vectors the data type is *vector* $\langle T, alignment \rangle$, where T can be a primitive type like *char*, *short*, *int*, *long*, *float*, *double*.

uBLAS is a C++ library that provides support for dense, packed and sparse matrices⁴. uBLAS is part of the BOOST project and corresponds to a CPU implementation of the BLAS standard (all levels) [13].

A vector is represented in a similar schema as STL, for the case of vectors the data type is *vector* $\langle T \rangle$, where T can be a primitive type like ViennaCL.

Armadillo is an open source linear algebra library written in C++. It provides object-like implementations for vectors, arrays, and cubes (tensors) [11]. There are libraries that use Armadillo for its implementation, such as MLPACK⁵ which is written using Armadillo's matrix support [4].

Armadillo represents arrays of elements using a column or row oriented format using the classes *Col* and *Row* respectively. The two types of data need as a parameter the type of data to be represented within the vector, these types of data can be: *uchar*, unsigned int (u32), int (s32), unsigned long long (u64), long long (s64), *float*, *double*⁶.

¹ <https://developer.nvidia.com/cuda-zone>.

² <http://www.khronos.org/opencl/>.

³ <https://isocpp.org/std/the-standard>.

⁴ http://www.boost.org/doc/libs/1_65_1/libs/numeric/ublas/doc/index.html.

⁵ <http://mlpack.org/index.html>.

⁶ u64 and s64 supported only in 64 bit systems.

LAPACK Linear Algebra PACKage (LAPACK) is a library of routines written in Fortran which implements the BLAS specification. Using additional modules the library can be used from other programming languages.

LAPACK use native vector implementation supported in the programming language ans is the industry standard for linear algebra software [2]. Additionally, it supports operations with complex numbers.

Vectors are represented as arrays of *float* or *double* elements, so the size in bytes needed to represent a vector is *size* * 4.

2.3 Current Compression Algorithms

As shown in the previous sections, the libraries described above work with uncompressed data. In most cases the information is represented as one-dimensional vectors containing 32-bit elements (float, int). When working with categorical data, there are some options for working with this type of information.

As shown in a previous paper, some options for compressing/representing one-dimensional (vector) data sets are: **Run-length encoding** consecutive data streams are encoded using a (*key, value*) pair [6], **Offset-list encoding** unlike the previous method, correlated pairs of data are encoded [6], **GZIP** this type of method overloads the CPU when decompressing the data [3] and **Bit-level compression** a data set is packaged in 32-bit blocks [5].

3 Processing Approach

In this section we propose a new mechanism for processing categorical data, the processing method proposed corresponds to a variation of the processing of data compressed using **bit level compression** method described in Sect. 2.3. The processing method developed corresponds to a process of bit-level elements with bit-shifting operations.

3.1 Block Storage

Figure 1 shows the storage distribution within a 32-bits block of a categorical data which values can be represented with 3 bits (0 to 7).

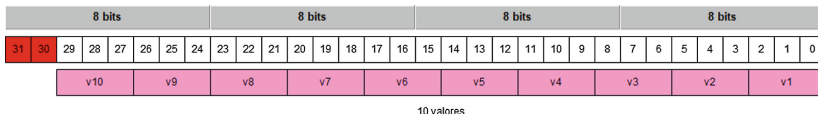


Fig. 1. Block storage

As we can see, in each 32-bits block we can store 10 values of the original data.

We can represent vector using the schema showed in Fig. 1, if we want to extend this representation to some more complex structures like matrices, we can use a row or col oriented representation of a matrix and encode the data in a so-called *matrix-block-storage*. In this case, each row or column has to be encoded with the scheme proposed.

3.2 Algorithms

The algorithm consists of representing within 4-bytes n-categorical values of a variable. This means that to access the value of a particular observation, a double indexing is necessary:

1. Access the index of the block (4-bytes) containing the searched element.
2. Index within 32 bits to access the value.

In this section we implement the L^2 norm algorithm. As you can see, all the functions described in the BLAS Level 1 can be implemented in the same way.

Algorithm 1 represents the algorithm for calculating the L^2 norm of a compressed vector. It should be noted that for each iteration on the compressed vector (*vector*), n-elements packet into 32 bits of data are accessed, so we need to process n-elements in each iteration.

The input values correspond to vector (input vector stored as compressed vector of *dataSize* bits), size (vector size) and dataSize (bit size used for representation).

```

Data: vector, size, dataSize
1  elementsPerBlock ← 32/dataSize;
2  mask ← sequence of dataSize-bits with value = 1
3  sum ← 0;
4  for index ← 0 to size - 1 do
5    | value ← vector[index];
6    | for i ← 0 to elementsPerBlock - 1 do
7    | | v ← value >> (i * dataSize) & mask;
8    | | sum ← sum + v * v;
9    | end
10 end
11 norm2 ← √sum

```

Algorithm 1: L^2 norm

The algorithm iterates over each element of the compressed vector and then iterates inside each block. The number of elements contained in each block corresponds to: $elementsPerBlock \leftarrow 32/dataSize$.

4 Experiments

This section presents the result of the processing of random generated vectors. The memory consumption of the representation of the data in the main memory of a computer was not analyzed, instead the processing speed was measured. All libraries were tested on Ubuntu 16.04 LTE using GCC 5.0.4 as compiler with the default settings.

4.1 Test Platform

To test the processing of the compressed data, several random generated vectors were used. The test data contains randomly generated values in the range [0..19]. All the libraries mentioned in the Sect. 2.2 were used with a float element representation.

The platform on which the tests were performed corresponds to:

- Processor: Intel(R) Core(TM) i5-5200 CPU, 2.20 GHz
- RAM Memory: 16 GB
- Operating System: Ubuntu 16.04LTE 64 bits with GCC 5.04 compiler

Some libraries provide particular implementations of the vector data type, in this cases we use the implementation provided by the library:

- LAPACK: `float *`
- uBLAS: `ublas::vector<float>`
- Armadillo: `arma::vec`
- ViennaCL: `std::vector<float>`

Section 4.2 shows some results obtained of running L^2 norm algorithm in the test vectors.

4.2 Results: L^2 Norm

This section shows the results obtained running the L^2 algorithm using the traditional libraries described in Sect. 2.2 and using the compressed format described in Sect. 3.1.

Figure 2 and Table 2 show the time taken to calculate the L^2 norm for random generated vectors of different sizes.

As we can see, the block compressed approach performs similar to *Armadillo* and has a better performance than the others libraries. The benefits of using the compressed format is the reduction of the amount memory necessary to store the dataset. We can expect the other functions described in the BLAS specification behave similarly.

The time needed to process vectors with size 10^3 and 10^4 is approximately equal to 0, so we remove them from the above table.

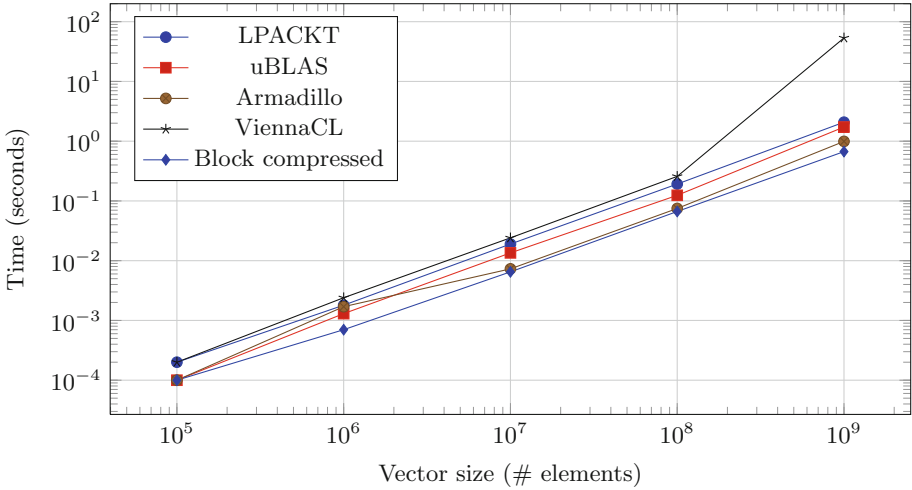


Fig. 2. L^2 norm - processing time

Table 2. L^2 norm - processing time

Vector size (# elements)	Time (seconds)				
	LAPACK	uBLAS	Armadillo	ViennaCL	Bit by bit
10 ⁵	0.0002	0.0001	0.0001	0.0002	0.0001
10 ⁶	0.0018	0.0013	0.0017	0.0024	0.0007
10 ⁷	0.0190	0.0135	0.0073	0.0241	0.0065
10 ⁸	0.1918	0.1242	0.0748	0.2576	0.0666
10 ⁹	2.0787	1.7273	0.9916	53.4602	0.6696

5 Conclusions

In this document we reviewed some common libraries which implement the algebraic operations that constitute the core for the implementation of machine learning algorithms. In general, it can be concluded that the categorical information processing proposed slightly reduces the processing time of the information compared to similar implementations under the BLAS Level 1 standard. The encoded data representation allows to reduce the amount of memory needed to represent the information as well as the number of iterations over the data.

Using the proposed algorithm of storage and processing, the calculation of the L^2 norm showed a slight reduction in the processing time (see Table 2). In the case of operations involving two or more vectors, it is recommended to encode both vectors using the same scheme (encode equal number of values in each block for all the vectors).

As future work we can propose (1) implement all the functions described in the BLAS standard, (2) extend representation and processing to more complex

structures like matrices (3) implement the representation and processing using parallel programming on CPU or GPU.

References

1. Bruni, R.: Discrete models for data imputation. *Discrete Appl. Math.* **144**(1–2), 59–69 (2004)
2. Cao, C., Dongarra, J., Du, P., Gates, M., Luszczek, P., Tomov, S.: cMAGMA: high performance dense linear algebra with OpenCL. In: *Proceedings of the International Workshop on OpenCL 2013 & 2014*, pp. 1:1–1:9 (2014)
3. Chen, Z., Gehrke, J., Korn, F.: Query optimization in compressed database systems. *ACM SIGMOD Rec.* **30**(2), 271–282 (2001)
4. Curtin, R.R., et al.: MLPACK: A Scalable C++ Machine Learning Library, pp. 1–5 (2012)
5. De Grande, P.: El formato redatam. *ESTUDIOS DEMOGRÁFICOS Y URBANOS* **31**, 811–832 (2016)
6. Elgohary, A., Boehm, M., Haas, P.J., Reiss, F.R., Reinwald, B.: Scaling machine learning via compressed linear algebra. *ACM SIGMOD Rec.* **46**(1), 42–49 (2017)
7. Ferres, J.C.: XII. Medición de la pobreza a través de los censos de población y vivienda, pp. 327–335 (2010)
8. Rai, P., Singh, S.: A survey of clustering techniques. *Int. J. Comput. Appl.* **7**(12), 1–5 (2010)
9. Rupp, K., Tillet, P., Rudolf, F., Weinbub, J.: ViennaCL—linear algebra library for multi-and many-core architectures. *SIAM J. Sci. Comput.* **38**, S412–S439 (2016)
10. Rupp, K., Rudolf, F., Weinbub, J.: ViennaCL—a high level linear algebra library for GPUs and multi-core CPUs. In: *International Workshop on GPUs and Scientific Applications*, pp. 51–56 (2010)
11. Sanderson, C., Curtin, R.: Armadillo: a template-based C++ library for linear algebra. *J. Open Sour. Softw.* **1**, 26 (2016)
12. Seshadri, V., et al.: Fast bulk bitwise and and or in DRAM. *IEEE Comput. Archit. Lett.* **14**(2), 127–131 (2015)
13. Tillet, P., Rupp, K., Selberherr, S.: An automatic OpenCL compute kernel generator for basic linear algebra operations. *Simul. Ser.* **44**(6), 4:1–4:2 (2012). <https://dl.acm.org/citation.cfm?id=2338820>



Spatial-Temporal K Nearest Neighbors Model on MapReduce for Traffic Flow Prediction

Anton Agafonov^(✉) and Alexander Yumaganov

Samara National Research University, 34, Moskovskoye shosse,
Samara 443086, Russia
ant.agafonov@gmail.com, yumagan@gmail.com

Abstract. Researches in the area of short-term traffic flow forecasting are important for traffic flow management in intelligent transport systems. In this paper, a distributed model for short-term traffic flow prediction based on the k nearest neighbors method is presented. This model takes into account spatial and temporal traffic flow distribution. We define a feature vector for a targeted road segment using traffic flow on segments in a compact area at different time intervals. To reduce the dimensionality of the feature vector, we use principal component analysis procedure. The proposed model is based on MapReduce technology and implemented using an Apache Spark framework. An experimental study data is obtained from the transportation network of Samara, Russia.

Keywords: Short-term traffic flow prediction · K nearest neighbors
MapReduce · Apache Spark

1 Introduction

The problems with traffic flow management are commonly experienced by every major city with a large transportation network. These problems may cause the delay of public transport, increased fuel consumption, environmental pollution and so on. So, traffic flow management has a big impact on urban economics. To efficiently solve traffic flow management problems it is important to provide accurate and timely traffic information to intelligent transport systems. As a result, there are a lot of science works devoted to the traffic flow forecasting problem.

Over the past decades, a number of work were dedicated to the traffic flow prediction problem. An overview of the methods of short-term traffic forecasting presented in the [12]. The recently developed models and techniques for short-term traffic flow forecasting and review of their advantages and disadvantages are described in [5]. Proposed methodologies can be classified into the following categories: time series models including seasonal autoregressive-moving average model (SARIMA) [6]; Kalman filter based models [3, 13]; artificial neural networks [4, 16]; k -nearest neighbors (kNN) method [15]; support vector regression

(SVR) [10, 14]; hybrid methods that combine the parametric and non-parametric methods [1, 11].

All described methods have both advantages and disadvantages when working under different conditions using different data sets, so it is hard to conclude that one method significantly superior others in any situation. In addition, most of the existing methods work in stand-alone mode and therefore have limitations on the used computational resources.

In this paper, a short-term traffic flow forecasting model based on the k nearest neighbors algorithm is presented. The results of experimental studies presented in [7, 15] showed the superiority of k NN based traffic flow forecasting models over another well-known forecasting models. However, if the sample data size is too large, k NN may not be suitable for real-time prediction due to the computational costs. We implemented the proposed model on the basis of big data processing framework MapReduce. The similar approach was used in [15]. The experimental studies on the real data indicate that the proposed model has good prediction precision and it is quite effective for real-time forecasting.

The paper is organized as follows. The problem formulation is presented in Sect. 2. Section 3 presents the proposed model and Sect. 4 describes its distributed implementation using MapReduce concept. Section 5 discusses the experimental results in terms of the accuracy, efficiency, and scalability of the proposed approach. Concluding remarks and possible directions for further research are described in Sect. 6.

2 Problem Formulation

A road network is considered as a directed graph $G = (V, E)$, with nodes $V, N_V = |V|$ representing the road intersections and edges $E, N_E = |E|$ denoting road segments.

Let V_t^j denotes an observed traffic flow characteristic on an edge $j \in E$ at time interval t . As a traffic flow characteristic can be used travel time, average speed, density or flow. In this work as a predicted traffic flow characteristic for the experimental study, we use the average traffic speed.

The short-term traffic flow forecasting problem can be formulated as follows: given a graph $G(V, E)$ and sequence $\{V_t^j\}, j \in E, t = 1, 2, \dots, T$ of observed traffic flow data, predict the traffic flow characteristic $\hat{V}_{t+\Delta}^j, j \in E$ at time interval $(t + \Delta)$ for a predefined prediction horizon Δ .

3 Proposed Model

In this paper, we propose a short-term traffic flow forecasting model based on non-parametric regression k -nearest neighbors algorithm. To apply the k NN method to the traffic flow prediction problem, it is necessary to solve the following tasks:

1. Define a feature vector to describe traffic flow.
2. Define a suitable distance metric to determine the proximity between a feature vector describing current traffic flow characteristics and feature vectors describing historical traffic flow observations.
3. Define a prediction function to forecast a traffic flow characteristic by selected nearest neighbors.

These challenges are described in the next subsections.

3.1 Feature Vector

The choice of a feature vector in the kNN method depends on the particular application of the method in practice. To solve the traffic flow prediction problem, it is reasonable to use a feature vector that takes into account spatial and temporal correlations of the traffic flow characteristics. In the paper [15] as a feature vector authors used traffic flow of targeted road segment j , downstream road segment $j - 1$ and upstream road segment $j + 1$ for T time intervals:

$$(V_{t-T}^j, \dots, V_{t-1}^j, V_t^j, V_{t-T}^{j-1}, \dots, V_{t-1}^{j-1}, V_t^{j-1}, V_{t-T}^{j+1}, \dots, V_{t-1}^{j+1}, V_t^{j+1}) \quad (1)$$

However, such feature vector does not consider traffic flow on adjacent segments. In addition, in some cases, the upstream-downstream road segments cannot be uniquely determined. Therefore, to describe traffic flow, it is proposed to use a feature vector that taking into account the traffic flow characteristics in the spatially-compact cluster of the transport network graph.

In this paper, we define the feature vector as follows:

1. The transportation network graph is partitioned into multiple spatially compact clusters $\{G_i\}$ corresponding to each targeted road segment. In each cluster i the feature vector is defined as follows:

$$\{V_t^j\}^i, j \in G_i, t = t_{cur} - T, \dots, t_{cur} \quad (2)$$

2. For the defined feature vector $\{V\}^i$ in the cluster i dimensionality reduction is performed using principal component analysis procedure. Result of this procedure define as a new feature vector $\{X_n\}^i, n = 1, \dots, N$.
3. Proposed feature vector for each road segment $j \in E$ is combined the initial feature vector of the targeted road segment j and the feature vector of the cluster i such that $j \in G_i$:

$$S_j = (\{V_t^j\}, \{X_n\}^i), \quad j \in G_i; \quad t = t_{cur} - T, \dots, t_{cur}; \quad n = 1, \dots, N. \quad (3)$$

In the next subsection, we describe how we choose a graph cluster for the targeted road segment (graph edge).

3.2 Graph Clustering

Let each edge $i \in E$ corresponding to the road segment e_i with two terminal points $x_{start}^i = (x_{start}^0, x_{start}^1)^i$ and $x_{end}^i = (x_{end}^0, x_{end}^1)^i$.

Denote the distance $r(i, j)$ between the edges $i \in E$ and $j \in E$ as the shortest path length from the origin vertex of the edge i, x_{start}^i to the end vertex of the edge j, x_{end}^j in the unweighted graph (i.e., the number of edges in the shortest path). The distance between each edges pair is calculated by a distance matrix.

Then each edge of the graph $i \in E$ is associated with the partition G_i^{dist} according to the following rule:

$$G_i^{dist} = \{j \in E : r(i, j) \leq R\}, \tag{4}$$

where R denotes the maximum distance that determines the size of the partition.

In Sect. 5 we provide an experimental study for the different values of the maximum distance R .

3.3 Proximity Measure

To define the proximity between the feature vectors, it is necessary to determine a suitable distance metric. Different distance functions between feature vectors are available in the literature, including Euclidean, Mahalanobis, Hamming distance.

In this paper, we use a weighted Euclidean distance, modified to use the feature vector describing transportation network clusters. The distance is considered separately for parts of the feature vector describing traffic flows on the current segment $\{V\}$ and in the corresponding cluster $\{X\}$.

$$d(S, \bar{S}^i) = \sqrt{\sum_{t=1}^T \beta^{T-t+1} (V_t - \bar{V}_t^i)^2} + \alpha \sqrt{\sum_{n=1}^N (X_n - \bar{X}_n^i)^2}. \tag{5}$$

where $0 < \alpha \leq 1$, T denotes the total number of time intervals in the feature vector, N denotes the total number of elements in the feature vector describing the graph cluster, S is the feature vector describing current traffic flow, \bar{S}^i is the feature vector describing i th historical traffic flow, V_t, \bar{V}_t^i are the feature vectors values representing respectively current and historical traffic flows on the selected road segment at time interval t , X_n, \bar{X}_n^i are the n th feature vectors values representing respectively current and historical traffic flows in the graph cluster.

3.4 Prediction Function

The traditional approach for estimating the value in kNN regression is to choose the average or the weighted average of the values of its k nearest neighbors [8]. In this paper, we use a prediction function by the weighted average that has the following form:

$$\hat{X}_{T+1} = \sum_{k=1}^K \frac{d_k^{-1}}{\sum_{k=1}^K d_k^{-1}} X_{T+1}^k \quad (6)$$

where \hat{X}_{T+1} is the predicted traffic flow value at the next time interval $T + 1$, X_{T+1}^k is the traffic flow value of the k th nearest neighbor at the time interval $T+1$, K is the total number of the neighbors, d_k denotes the distance between the feature vector describing the current traffic data and the k th nearest neighbors.

4 MapReduce Implementation

The proposed model of traffic flow prediction uses a large amount of current and historical traffic flow data. To improve the efficiency of the proposed model, we implement it on the basis of the MapReduce model [2] for distributed computing using Apache Spark engine [9].

MapReduce provides parallel processing of the big amount of data in computing clusters. MapReduce model usually consists of three main steps: Map, Shuffle and Reduce.

The first step is a preparation of input data for the Map phase. At first, the historical and test data are divided into partitions. The optimal number of such partitions depends on the amount of processed data and the number of computing nodes. Then, ordered pairs of historical and test data partitions are formed using the Cartesian product. Next, in the Map phase, a map function is applied to each pair of partitions. This function returns an intermediate set of key/value pairs - the test element/local list of k nearest neighbors. At the Shuffle phase, the key-value pairs are grouped and transferred to the reduce functions. At the final Reduce step for each test data element, the set of local k nearest neighbors lists is converted to the resulting (global) list of k nearest neighbors. The resulting lists of k nearest neighbors are subsequently used to find the predicted value traffic flow.

The results of evaluating the efficiency of the proposed model based on the MapReduce concept are presented in Sect. 5.

5 Experiments

Experimental studies of the developed model were carried out for the part of the transportation network of the Samara city, Russia. The graph contains 10818 edges (road segments). As the traffic data for the experimental studies, the average traffic speed was used. The data set contains records for 49 days, from November 13, 2015.

We compare the proposed model with the different maximum distance R using in the graph clustering algorithms, the spatial-temporal weighted TDUD-KNN model described in [15], and the seasonal time-series SARIMA model. During testing, these models are performed on each day (test set) and the remaining

days considered as a historical data set (training set). Then the average performance across the full data set is calculated.

To compare the performance of the proposed model, we use two standard metrics: mean absolute error (MAE) and mean absolute percentage error (MAPE) that can be formulated as:

$$\text{MAPE} = \frac{1}{n} \sum_{t=1}^n \frac{|V_t - \hat{V}_t|}{V_t}, \quad \text{MAE} = \frac{1}{n} \sum_{t=1}^n |V_t - \hat{V}_t|, \quad (7)$$

where V_t is the actual value of traffic flow at time interval t , \hat{V}_t is the predicted value for the same time interval t , n is the total number of observations.

We conduct the experiments on an Apache Spark cluster. Comparison results of the models are shown in Table 1. The maximum distance $R = 0$ means that only adjacent road segments to the targeted one are used.

Table 1. Algorithms comparison

	$R = 0$	$R = 1$	$R = 2$	TDUD-KNN	SARIMA
MAPE	0.1396	0.1381	0.141	0.1443	0.1459
MAE	2.93	2.89	2.91	3.02	3.01

From these results, we can observe that the proposed model obtains better prediction performance than the TDUD-KNN and the SARIMA models. The slightly better results were shown with the distance $R = 1$.

Next, we illustrate the prediction performance of the proposed model (with distance $R = 1$), TDUD-KNN model and SARIMA model by the MAPE (Fig. 1(a)) and the MAE (Fig. 1(b)) criteria for two weeks.

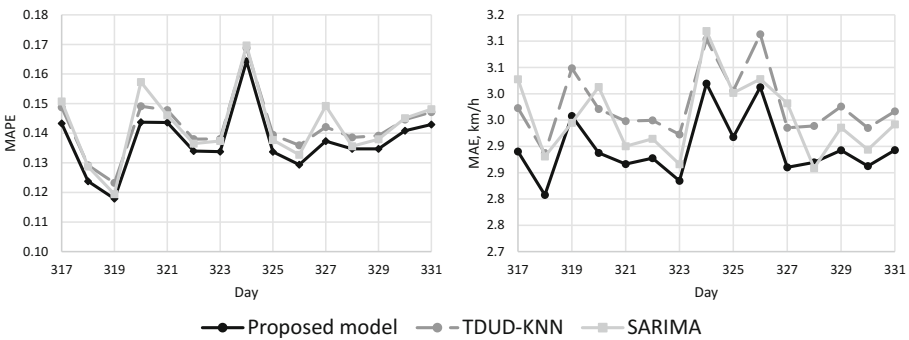


Fig. 1. MAE and MAPE of the models

Analyzing the obtained results, we can conclude that the proposed model shows better accuracy for each day.

To evaluate the scalability and efficiency of the developed model with MapReduce implementation on an Apache Spark framework, a number of experiments were performed. We validate the speedup and scaleup [15] of the parallel algorithm.

The speedup metric shows how much the parallel implementation of the algorithm is faster than the corresponding standalone implementation with fixed data set size. The scaleup metric shows how the performance changes with a proportional increase in the data set size and the number of computation nodes. These metrics are calculated as follows:

$$Speedup = \frac{T_s}{T_p}, \quad Scaleup = \frac{T_s}{T_{pp}}, \quad (8)$$

where T_s is the execution time taken to solve a given problem on a single computing node; T_p is the execution time taken to solve a given problem on p computational nodes, T_{pp} is the execution time taken to solve a problem with p -times larger data set on p computational nodes.

Table 2. Speedup and scaleup

Cores	1	2	3	4	5	6
Speedup	1	1.97	2.55	3.11	3.69	4.54
Scaleup	1	0.97	0.92	0.8	0.77	0.72

The results in Table 2 show that the use of the MapReduce parallel computing model significantly improves the performance of the prediction model. Based on the analysis of the obtained results, we can conclude that the proposed model, using the MapReduce paradigm, has scalability close to linear, which makes it possible to effectively use this model when processing a large amount of data.

6 Conclusion

The paper presents the distributed spatial-temporal model of short-term traffic forecasting based on the method of non-parametric regression k nearest neighbors. In the model, spatial and temporal characteristics of the transport flow in a compact cluster of the transport network are taken into account for the feature space description.

For distributed Big Data processing, we use MapReduce processing model implemented in the open source cluster-computing framework Apache Spark. Experimental analysis on real-world traffic data sets allows us to conclude that the proposed model has a good prediction accuracy and reasonable execution time, sufficient for real-time prediction.

For future work, it would be interesting to compare the proposed model with other algorithms, including neural networks, support vector regression, and to test these algorithms on different public open traffic data sets.

Acknowledgments. This work was supported by Project no. RFMEFI57518X0177 by the Ministry of Education and Science of the Russian Federation.

References

1. Agafonov, A., Myasnikov, V.: Traffic flow forecasting algorithm based on combination of adaptive elementary predictors. *Commun. Comput. Inf. Sci.* **542**, 163–174 (2015). https://doi.org/10.1007/978-3-319-26123-2_16
2. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. *Commun. ACM* **51**(1), 107–113 (2008). <https://doi.org/10.1145/1327452.1327492>
3. Guo, J., Williams, B.: Real-time short-term traffic speed level forecasting and uncertainty quantification using layered Kalman filters. *Transp. Res. Rec.* **2175**, 28–37 (2010). <https://doi.org/10.3141/2175-04>
4. Lana, I., Del Ser, J., Velez, M., Oregi, I.: Joint feature selection and parameter tuning for short-term traffic flow forecasting based on heuristically optimized multi-layer neural networks. *Adv. Intell. Syst. Comput.* **514**, 91–100 (2017). https://doi.org/10.1007/978-981-10-3728-3_10
5. Lana, I., Del Ser, J., Velez, M., Vlahogianni, E.: Road traffic forecasting: recent advances and new challenges. *IEEE Intell. Transp. Syst. Mag.* **10**(2), 93–109 (2018). <https://doi.org/10.1109/ITS.2018.2806634>
6. Shekhar, S., Williams, B.: Adaptive seasonal time series models for forecasting short-term traffic flow. *Transp. Res. Rec.* **2024**, 116–125 (2007). <https://doi.org/10.3141/2024-14>
7. Smith, B., Demetsky, M.: Traffic flow forecasting: comparison of modeling approaches. *J. Transp. Eng.* **123**(4), 261–266 (1997). [https://doi.org/10.1061/\(ASCE\)0733-947X\(1997\)123:4\(261\)](https://doi.org/10.1061/(ASCE)0733-947X(1997)123:4(261))
8. Smith, B., Williams, B., Keith Oswald, R.: Comparison of parametric and nonparametric models for traffic flow forecasting. *Transp. Res. Part C: Emerg. Technol.* **10**(4), 303–321 (2002). [https://doi.org/10.1016/S0968-090X\(02\)00009-8](https://doi.org/10.1016/S0968-090X(02)00009-8)
9. Spark, A.: Apache spark web site (2018). <https://spark.apache.org/>
10. Su, H., Zhang, L., Yu, S.: Short-term traffic flow prediction based on incremental support vector regression. vol. 1, pp. 640–645 (2007). <https://doi.org/10.1109/ICNC.2007.661>
11. Sun, S., Zhang, C.: The selective random subspace predictor for traffic flow forecasting. *IEEE Trans. Intell. Transp. Syst.* **8**(2), 367–373 (2007). <https://doi.org/10.1109/TITS.2006.888603>
12. Vlahogianni, E., Golias, J., Karlaftis, M.: Short-term traffic forecasting: overview of objectives and methods. *Transp. Rev.* **24**(5), 533–557 (2004). <https://doi.org/10.1080/0144164042000195072>
13. Wang, Y., Papageorgiou, M.: Real-time freeway traffic state estimation based on extended Kalman filter: a general approach. *Transp. Res. Part B: Methodol.* **39**(2), 141–167 (2005). <https://doi.org/10.1016/j.trb.2004.03.003>
14. Wu, C.H., Ho, J.M., Lee, D.: Travel-time prediction with support vector regression. *IEEE Trans. Intell. Transp. Syst.* **5**(4), 276–281 (2004). <https://doi.org/10.1109/TITS.2004.837813>
15. Xia, D., Wang, B., Li, H., Li, Y., Zhang, Z.: A distributed spatial-temporal weighted model on mapreduce for short-term traffic flow forecasting. *Neurocomputing* **179**, 246–26 (2016). <https://doi.org/10.1016/j.neucom.2015.12.013>
16. Yin, H., Wong, S., Xu, J., Wong, C.: Urban traffic flow prediction using a fuzzy-neural approach. *Transp. Res. Part C: Emerg. Technol.* **10**(2), 85–98 (2002). [https://doi.org/10.1016/S0968-090X\(01\)00004-3](https://doi.org/10.1016/S0968-090X(01)00004-3)



Exploring the Perceived Usefulness and Attitude Towards Using Tesys e-Learning Platform

Paul-Stefan Popescu^{1(✉)}, Costel Ionascu², and Marian Cristian Mihaescu¹

¹ Faculty of Automation, Computers and Electronics, University of Craiova,
Str. Decebal nr. 107, Craiova, Romania
{spopescu,mihaescu}@software.ucv.ro

² Faculty of Economy and Business Administration, University of Craiova,
Str. A.I. Cuza nr. 13, Craiova, Romania
icostelm@yahoo.com

Abstract. In this paper we present a study that aims at exploring two components of the Technology Acceptance Model. As an evaluation environment, we used Tesys e-Learning platform, R programming language for data processing, and Google Forms for data collection. The study intended to explore if there are significant differences between genders regarding the perceived usefulness and attitude towards evaluation using online educational environment. The questionnaire exploring the above-mentioned components was applied to a group of full-time students that have never used e-Learning platforms. The results showed that the questionnaire was reliable and there is no cross-gender difference regarding students' perceived usefulness of e-Learning platforms and their attitude towards using them. The overall feedback received from each of the explored items is positive with a grade above four on a five Likert scale.

Keywords: Interface evaluation · e-Learning · Data analysis · HCI

1 Introduction

Evaluation is one of the main activities that a student has to undergo in a specific discipline. Depending on the educational context, there can be several types of evaluations and some of them can be modeled using online educational environments. The most flexible type of evaluation is multiple choice quizzes with one or more valid answers as this type can be easily implemented in the e-Learning platforms and it offers almost the same experience as the full-time education systems.

There are several advantages taking test using online educational environments but there are also some drawbacks. The most important advantage is the elimination of the human error because once the system is set up, there cannot be errors during the correction and grading of the papers. Taking tests using online educational environments don't require the presence of the students in

the classroom and there is no specific time to start the test as the system can have an implemented countdown timer that may started by the student. Another advantage is that you do not waste time redoing the test setup every year or semester because once the system is set up with a consistent pool of questions it can be used over the years. The randomization used in most of the programming languages ensures a very objective questions selection and during the tests, depending on the number of questions, there is a low chance that two students can have the same question at the same time. Drawbacks are mostly technical because an unreliable Internet connection or low power can interrupt the tests.

The problem approached in this paper is the exploration of the significant indicators that can make the difference between groups of students and also how they perceive Tesys [1] e-Learning environment. Regarding the group division, considering the context of the study, only a cross gender analysis could have been performed because they had the same age and they attended the same faculty, specialization and year of study.

Taking into consideration all the benefits and drawbacks of testing using e-Learning platform, we need to understand how students perceive this action since their perception [2] can directly influence their motivation to learn. In this paper we aim at exploring the Perceived Usefulness (PU) and Attitude Towards (AT) using Tesys e-Learning platform. PU and AT are two parts of the Technology Acceptance Model (TAM), [3] which is a great theoretical tool that helps us understand how students accept the e-Learning system.

Regarding TAM, there are seven components that should be explored every time we need to evaluate an e-Learning system and see how well it is accepted: perceived ease of use, perceived usefulness, attitude, behavioral intention, e-Learning self-efficacy, subjective norm, and system accessibility. In our case, we chose PU and AT due to the user's experience with the e-Learning platform; in order to evaluate the whole model, the students needed to have experienced it fully and to have used all the platform capabilities. Another reason for exploring only these two components is that the subjects of the study were full-time students, not distance learning students. There is a difference between the form of education that students attend because full-time students are not supposed to have used previously an e-Learning platform and their first contact was during a test at one of the disciplines studied during the semester.

For our experiment, we used Tesys e-Learning platform, an e-Learning platform designed and developed at the University of Craiova. It was specially designed to fulfill the needs of the faculties within our University that provide distance learning programmes and it is continuously developed. We also strive to make it more user-friendly and helpful for the students and professors from any faculty with or without technical profile or background.

In this paper, we try to analyze if there are any differences between male and female [4] students and also their perception on taking tests using e-Learning platforms and their attitude towards it. For this study, we used Wilcoxon [5] statistic for group division and then we tested the reliability of the survey using Cronbach's alpha [6].

2 Related Works

The work described in this paper follows the analysis of three previous papers. The first paper [7] evaluated the Tesys e-Learning platform interface from a high-level view. The study revealed some small problems and helped us understand better the ease with which the platform can be used by students and professors. This paper was the starting point for the next two papers, extending the number of responders and also addressing more standardized problems. The next paper [8] explored one of the TAM components, i.e. perceived ease of use, and a slightly larger number of responders participated. The students participating in the study were taking distance learning programmes and they were very familiar with the platform. The study revealed several limitations and we considered that the professor's interface needed to be evaluated as well. The last paper [9] analyzed the perceived ease of use of the professor's interface and the responders were professors teaching two distance learning programmes on Tesys. The results presented that there are no factors that influence directly the professors' perception of the e-Learning platform. We received a good feedback from the evaluation of the professor's interface but no avenues for improving mainly because both students and professors participating in the previous studies were already familiar with the e-Learning platform, therefore they could find the functionalities and controls easily.

A recent research published by Pribeanu et al. [10] presented a similar approach applied to Facebook users and on a larger group of responders. The study explored only the PU component and the responders group was separated by two factors: gender and country. The findings showed that in both countries, students perceived Facebook as being more useful for collaboration, Romanian students found Facebook more useful than the Lithuanian students and the female students had a higher perception of Facebook usefulness. There are also other papers that explore by different methods Facebook and the reasons of using it [11], but choosing TAM or TAM components provides a standardized approach and comparable results.

Gender difference exploration in online systems is a much older problem, which was referred to in many papers, such as 'Gender differences in Facebook addiction' [12]. The authors conducted the study at Technical University of Civil engineering from Bucharest and the paper aimed at analyzing the relationship between Facebook addiction and the negative consequences on the students' work, and also at analyzing the measurement invariance across gender. Their results showed that females spend more time on Facebook, which could lead to the conclusion women are more prone to score higher at behavioral addiction involving social interaction [13].

Regarding TAM [14], there are many papers that present its usefulness on different domains, such as mobile commerce [15], online banking [16], online shopping [17] or even a more general approach [18]. The authors of [14] state that over the time, TAM has evolved and became a key model in understanding predictors of human behavior towards potential acceptance or rejection of

technology; this assumption is based on 85 scientific publications, selected and classified according to their aim and content.

Recent papers [19] explore the influence of gender and age on e-Learning platforms and enforce the idea that the success of e-Learning depends to a considerable extent on students' acceptance and use of technology. They also state that it has become imperative for practitioners and policy makers to understand the factors affecting the user's acceptance of e-Learning environments in order to provide a better learning experience. The authors found that age influenced the effect on the perceived ease of use, perceived usefulness, self-efficacy and behavioral intention, while gender influenced the perceived ease of use, social norm and behavioral intention to use e-Learning systems. For our research, we could not take age as an influencing factor because all the students were enrolled in the same year of study and the age gap between them was small.

3 Method

3.1 Context of the Study

We applied this study to the third year students of the Faculty of Automation, Computers and Electronics so we can say that they had a technical background, but they had no previous contact with e-Learning platforms. The students had to take a test consisting of 10 questions regarding software engineering discipline and then, they completed a survey described below. We need to emphasize some details of the test as they represent their experience with the e-Learning platform and they may influence the survey results. There were 50 questions assigned to the testing unit and each student received 10 questions randomly assigned by the system. They were multiple choice questions and two minutes were assigned for each question.

The survey consisted of 6 questions adapted to our context from the TAM model defined by S. Y. Park [3], one group separation question and one free completion form designed to receive feedback were added in order to improve the feedback and to extend the analysis capabilities. Table 1 presents the indicators

Table 1. Explored items.

Item	Question
PU1	Doing tests through on-line educational environments would improve my learning performance?
PU2	Doing tests through online educational environments would increase my academic productivity?
PU3	Doing tests through an online educational environment would be easier than classical tests?
AT1	Doing tests through e-Learning platforms is a good idea?
AT2	Doing tests through e-Learning platforms is a wise idea?
AT3	Do I feel positive about doing tests using e-Learning platforms?

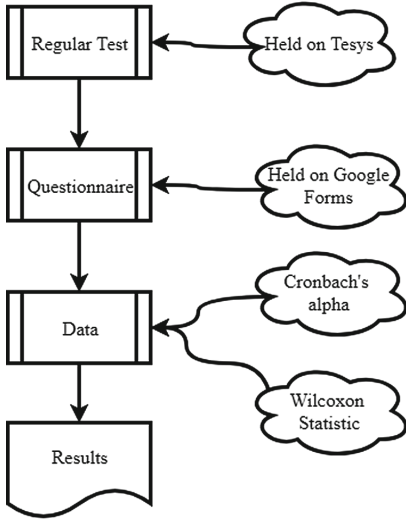


Fig. 1. Study pipeline.

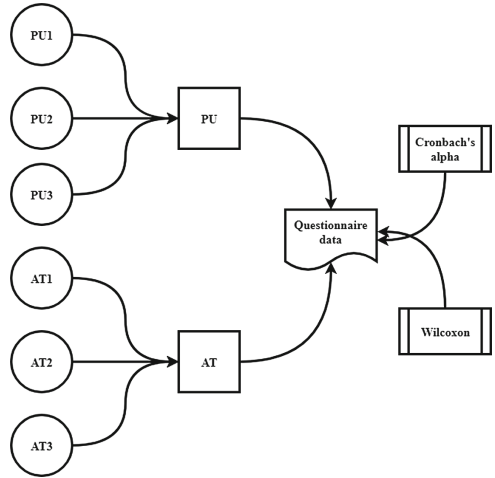


Fig. 2. Study design.

used to explore PU and AT regarding Tesys e-learning platform; we used three questions for each TAM explored component (PU and AT). The items were specially designed and modified to capture the difference between doing regular tests for full time education programmes and tests through e-Learning platforms. We used a one to five Likert scale for each question; for each answer, one was assigned as the worth answer or a strong disagree and five to a positive answer or a strong agreement with the statement.

The pipeline of this study is presented in Fig. 1. We started with a test held on Tesys e-Learning platform, designed to evaluate the knowledge level of the students. Then, the students were asked to complete a survey, which was set up on Google forms. Once the study ended, the data was preprocessed to a more familiar format and then we used R programming language to apply Cronbach’s alpha and Wilcoxon statistic on this data in order to obtain the results presented in this paper.

Figure 2 presents the study design; on the left side, we have the explored items and how they compose the explored components (PU and AT) of the TAM. Both components are explored using a survey completed by the students. We applied two statistical methods to the collected data: Cronbach’s alpha and Wilcoxon statistic, which are used for group exploration and items reliability.

The number of male students answering (37), which represents 63.8% of the study audience, is significantly higher than the number of female students (36.2%). We analyzed the proportions and also the number of responders before proceeding further and we concluded that the number of female responders (21 of a total of 58) is big enough to calculate statistical differences between groups.

All the collected data can be checked at <https://goo.gl/ZVTwh4>.

4 Study Results

The study presented in this paper analyses the differences between genders regarding the PU and AT using Wilcoxon statistic, and the reliability of survey collected data using Cronbach alpha. Firstly, we used Wilcoxon statistics on the survey collected data after we have separated it based on the responders' gender. After computing the p-value given by Wilcoxon, we also calculated the average score received by male and female responders for each question.

Table 2. Results based on Wilcoxon statistic.

Item	P-value	Average for male	Average for female
PU1	0.55	4.13	4.28
PU2	0.59	4	4.19
PU3	0.69	4.59	4.47
AT1	0.93	4.56	4.61
AT2	0.30	4.18	4.47
AT3	0.27	4.40	4.66

Table 2 presents p-value and the average score of the answers from male and female responders. The p-value must be less than 0.05 in order to conclude a strong statistical difference between genders and its range of values is from zero to one, where one means a clear, non-statistical difference. In our case, p-value indicates that users' gender does not reveal a statistical difference in terms of perceived usefulness and attitude towards using an e-Learning platform for testing purposes. It is noteworthy to mention that the fact that these students are very familiar with information technology may have a strong impact on our study. They are not familiar with e-Learning systems and they didn't have tests using e-Learning systems, but depending on their knowledge they may tend to perceive technology friendlier than others.

By examining the grade gathered for each analyzed item, we can see that there is a small overall difference between male and female subjects, as females gave a slightly higher grades for each question. Therefore, even if every question got a grade higher than four, female responders tend to perceive better the testing through e-Learning platforms and also their attitude towards using them is better. Overall, a grade higher than four on a scale from one to five give us a good feedback regarding the explored items.

After applying Cronbach's alpha on the six explored items we gathered the values presented in Table 3. On the first column we have the identifier for a specific item, and on the next ones their results. The second row presents the raw alpha's value, with a range value from 0 to 1. A value of 0.7 or higher for alpha represents a reliable survey and the lowest acceptable value is 0.6. In our case, raw alpha is between 0.74 and 0.83, which certifies the reliability of each question. The third column represents the standardized correlation based alpha while the second column which was covariance based alpha [20]. In both cases,

Table 3. Results based on Cronbach's alpha

Item	Raw alpha	Std. alpha	Average score
PU1	0.75	0.76	4.2
PU2	0.74	0.76	4.1
PU3	0.82	0.82	4.6
AT1	0.75	0.75	4.6
AT2	0.73	0.74	4.3
AT3	0.75	0.74	4.5

the values are above the reliability limit and we can conclude that the survey gathered results are reliable enough.

The average score represents the average for each question and give us an insight into the explored items. Our explored indicators regarding PU and AT got average values between 4.1 and 4.6. The question with an average of 4.1 explored if the students' perception of the usage of e-Learning would improve the academic productivity. The average was determined by thirteen students who graded less than four. The question with the highest average score referred to the difference between doing classical tests and e-Learning tests.

5 Conclusion and Future Work

This paper presents a study conducted on two components of TAM model, applied on Tesys e-Learning platform. The results revealed that there is no statistical difference between male and female in terms of how they perceive the usefulness of using e-Learning platforms or their attitude towards using e-Learning platforms. Based on the explored items, we can conclude that they think it is useful and have a good attitude towards using the e-Learning platforms. We also carried out reliability tests on the explored items in order to get an insight into the values received for each explored item and the confidence factor that scored high enough for each of them.

The study needs to be further extended into two main directions: one is to explore other TAM components or the whole model, and the other one is to conduct this study on a different group of students, attending different faculties. Enforcing the study with groups of students enrolled at other faculties can give us an insight into the bias that may be given by their context.

Acknowledgements. This work was supported by QFORIT Programme, University of Craiova, 2018.

References

1. Burdescu, D.D., Mihaescu, M.C.: TESYS: eLearning application built on a web platform. In: ICEB, pp. 315–318 (2006)
2. Song, L., Singleton, E.S., Hill, J.R., Koh, M.H.: Improving online learning: student perceptions of useful and challenging characteristics. *Internet High. Educ.* **7**(1), 59–70 (2004)

3. Park, S.Y.: An analysis of the technology acceptance model in understanding university students' behavioral intention to use e-learning. *J. Educ. Technol. Soc.* **12**(3), 15 (2009)
4. Ong, C.S., Lai, J.Y.: Gender differences in perceptions and relationships among dominants of e-learning acceptance. *Comput. Hum. Behav.* **22**(5), 816–829 (2006)
5. Bethea, R.M.: *Statistical Methods for Engineers and Scientists*. Routledge (2018)
6. Bonett, D.G., Wright, T.A.: Cronbach's alpha reliability: interval estimation, hypothesis testing, and sample size planning. *J. Organ. Behav.* **36**(1), 3–15 (2015)
7. Popescu, P.S., Mihaescu, M.C., Mocanu, M., Ionascu, C.: Evaluation of the Tesys e-Learning platform's interface. In: *RoCHI-International Conference on Human-Computer Interaction*, pp. 86–90 (2016)
8. Mihaescu, M.C., Popescu, P.S., Ionascu, C.M.: Questionnaire analysis for improvement of student's interaction in Tesys e-Learning platform. *Roman. J. Hum.-Comput. Interact.* **10**(1), 61–74 (2017)
9. Popescu, P.S., Mihaescu, M.C., Mocanu, M., Ionascu, C.: Evaluation of the Tesys e-Learning platform's interface. In: *RoCHI-International Conference on Human-Computer Interaction*, pp. 15–20 (2017)
10. Pribeanu, C., Balog, A., Lamanauskas, V., Slekiene, V.: Perceived usefulness of Facebook for university students: a gender analysis across two countries. In: *RoCHI- International Conference on Human-Computer Interaction*, pp. 81–86 (2017)
11. Iordache, D.D., Pribeanu, C.: Exploring the motives of using Facebook - a multi-dimensional approach. *Rev. Romana Interactiune Om-Calc.* **9**(1), 19–34 (2016)
12. Iordache, D.D., Manea, V.: Gender differences in Facebook addiction. *Rev. Romana Interactiune Om-Calc.* **9**(4), 334–346 (2016)
13. Andreassen, C.S., Torsheim, T., Brunborg, G.S., Pallesen, S.: Development of a Facebook addiction scale. *Psychol. Reports* **110**, 501–517 (2012)
14. Marangunić, N., Granić, A.: Technology acceptance model: a literature review from 1986 to 2013. *Univ. Access Inf. Soc.* **14**(1), 81–95 (2015)
15. Wu, J.H., Wang, S.C.: What drives mobile commerce?: An empirical evaluation of the revised technology acceptance model. *Inf. Manag.* **42**(5), 719–729 (2005)
16. Pikkarainen, T., Pikkarainen, K., Karjaluoto, H., Pahlila, S.: Consumer acceptance of online banking: an extension of the technology acceptance model. *Internet Res.* **14**(3), 224–235 (2004)
17. Ashraf, A.R., Thongpapanl, N., Auh, S.: The application of the technology acceptance model under different cultural contexts: the case of online shopping adoption. *J. Int. Mark.* **22**(3), 68–93 (2014)
18. Van der Heijden, H.: User acceptance of hedonic information systems. *MIS Q.* 695–704 (2004)
19. Tarhini, A., Hone, K., Liu, X.: Measuring the moderating effect of gender and age on e-learning acceptance in England: a structural equation modeling approach for an extended technology acceptance model. *J. Educ. Comput. Res.* **51**(2), 163–184 (2014)
20. Falk, C.F., Savalei, V.: The relationship between unstandardized and standardized alpha, true reliability, and the underlying measurement model. *J. Pers. Assess.* **93**(5), 445–453 (2011)



An ELM Based Regression Model for ECG Artifact Minimization from Single Channel EEG

Chinmayee Dora^(✉) and Pradyut Kumar Biswal

Department of Electronics and Telecommunication Engineering, International Institute of Information Technology, Bhubaneswar, India
chinmayee.dora@gmail.com, pradyut@iiit-bh.ac.in

Abstract. Electroencephalogram (EEG) is the most widely used non-invasive technique to record the electrical activity of brain for analysis or diagnostic procedures. The sensitive electrodes of EEG are susceptible to high amplitude electrocardiogram (ECG) signals, which superimpose on the recorded EEG. Minimizing this artifact effectively from a single channel EEG without a reference ECG channel is a challenge. In this paper, extreme learning machine (ELM) algorithm as a regression model is implemented for ECG artifact removal from single channel EEG. The S-transform (ST) of the EEG signals are used as the feature set of for ELM training and testing. ST combines the progressive resolution and absolutely referenced phase information for the given time series uniquely. By training the ELM with pairs of contaminated and clean EEG signals both in magnitude and phase, is able to minimize the ECG artifact from contaminated EEG signal effectively in the testing phase. The average Root mean square error (*RMSE*) and the correlation coefficient (*CC*) for actual EEG signal to the estimated EEG signal from the ELM based regression model obtained are 0.32 and 0.96 respectively.

Keywords: EEG · ECG · Artifact · ELM · Regression

1 Introduction

A human brain is a natural processor for all the control activity of the human body. Electroencephalogram (EEG) is an acclaimed non-invasive tool to record and analyze the brain activities. The EEG signals are very low amplitude bio-electric signals in the range of microvolts. Hence the sensitive electrodes used for recording EEG signals are susceptible to physiological artifacts like electro-oculogram (EOG), electromyogram (EMG) and electrocardiogram (ECG) signals.

The spike like activity with a quasi-periodic pattern of ECG signal contaminate the EEG signals near the auricular lobe [4]. While analyzing the EEG, if a concurrent ECG channel is absent then the presence of ECG artifact in the EEG could mislead the analysis/diagnostic problem. Several approaches are proposed

using different techniques in the last decade [3, 7, 11, 16, 20]. Most of the algorithm considers the higher energy of QRS waves in EEG or its quasi periodicity to detect ECG in EEG, which requires thresholdings in the algorithm. Hamaneh et. al. [7] proposed algorithm using independent component analysis (ICA) and wavelets, which is multichannel approach for ECG artifact correction, whereas Devuyt et. al. [3] proposed ICA based technique to using single channel EEG with concurrent ECG channel. Jafarimand et. al. proposed a neural network based adaptive filter to remove ECG artifact from EEG which requires a reference ECG channel. Patel et.al. proposed an ensembled empirical mode decomposition (EEMD) based algorithm for ECG artifact suppression which also requires an additional ECG channel. Till date, Extreme machine learning (ELM) in the context of EEG is used for several classification problems for brain computer interface (BCI)[18, 19], motor-imagery task [5] classification, sleep stage identification [1, 12], epileptic seizure detection [2, 13, 14]. The essence of ELM is its fast learning, random initialization of hidden node parameters and a good generalization performance. ELM based regression model for speech enhancement are proposed by Odelowo et. al. [15] and Hussain et. al. [10] for retrieving voice signals in presence of different background noises.

Here in this paper, the approach is to remove the ECG artifact from the contaminated EEG using ELM algorithm, which upon training with reference contaminated and clean EEG signals can eliminate the ECG artifact effectively from the contaminated EEG signal without any reference ECG channel. The S-Transform (ST) of the EEG signal is provided as the feature set to the ELM input node. To our best of knowledge and belief, artifact removal from EEG signals using ELM regression model is not proposed in the literature.

The paper is organized as follows. Section 2 briefly introduce the concepts of ST and ELM. Section 3 describes the methodology. The data set used for evaluation and obtained results are discussed in the Sect. 4. Finally, a conclusion is drawn in the Sect. 5.

2 Basic Preliminaries

2.1 Stockwell Transform

S-transform is suggested by R. G. Stockwell in 1996 [17], to produce a time- frequency representation of signal in time domain. ST provides the time-frequency representation of the EEG time series. This transformation is special case of multi- resolution Fourier transform defined with parameters τ , the time of spectral localization and f , Fourier frequency. The continuous S-transformation of any time series $x(t)$ at time $t = \tau$ and frequency f is defined by,

$$S(\tau, f) = \int_{-\infty}^{\infty} x(t) \frac{|f|}{\sqrt{2\pi}} e^{-\frac{(\tau-t)^2 f^2}{2}} e^{-i2\pi ft} dt \quad (1)$$

When the time series $x(t)$ is windowed with a Gaussian window function $g(t)$, the spectrum of the signal results as,

$$X(f) = \int_{-\infty}^{\infty} x(t)g(t)e^{-i2\pi ft}dt \tag{2}$$

Where the general Gaussian function $g(t)$ is a function of time translation τ and window width σ ,

$$g(t) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{t^2}{2\sigma^2}} \tag{3}$$

For simplification and making the transformation independent of the third parameter σ , a restriction constraint is added. This is achieved with σ to be proportional to time period or inverse of frequency i.e. $\sigma = \frac{1}{|f|}$.

The discrete version of ST signal $x[kT]$ corresponding to $x(t)$ with $k = 0, 1, \dots, N - 1$ and time sampling interval T is given by:

$$S \left[jT, \frac{n}{NT} \right] = \sum_{m=0}^{N-1} X \left[\frac{m+n}{NT} \right] e^{-\frac{2\pi^2 m^2}{n^2}} e^{\frac{i2\pi mj}{N}} \tag{4}$$

where, $X \left[\frac{m+n}{NT} \right]$ is the Fourier transform of $x[kT]$ and $j, m, n = 0, 1, \dots, N - 1$.

The discrete ST of the EEG is the complex $ST[t, f]$ matrix of dimension $N \times M$, with rows and columns representing time and frequencies respectively. The matrix is used as M numbers of feature set for each distinct sample of the time series.

2.2 ELM

The mathematical model of a single Hidden Layer Feedforward Neural Network (SLFN) for N number of arbitrary individual inputs $(y_i, t_i) \in R^n \times R^m$, with hidden nodes P is given by,

$$\sum_{i=1}^P \beta_i a_i(y_j) = \sum_{i=1}^P \beta_i A_f(w_i, b_i, y_j) = o_j, \quad j = 1, 2, \dots, N \tag{5}$$

where a_i is the output function $A_f(w_i, b_i, y_j)$ of the i^{th} hidden node. Approximation of SLFN for the N samples with zero error suggests that $\sum_{j=1}^P \|o_j - t_j\| = 0$, for such (w_i, b_i) and β_i that

$$\sum_{i=1}^P \beta_i A_f(w_i, b_i, y_j) = t_j, \quad j = 1, \dots, N \tag{6}$$

Writing the above equations in matrix form,

$$H\beta = T \tag{7}$$

where,

$$H = \begin{bmatrix} h(y_1) \\ \vdots \\ h(y_N) \end{bmatrix} = \begin{bmatrix} A_f(w_1, b_1, x_1) & \dots & A_f(w_P, b_P, y_P) \\ \vdots & \dots & \vdots \\ A_f(w_1, b_1, x_N) & \dots & A_f(w_P, b_P, y_N) \end{bmatrix} \tag{8}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_P^T \end{bmatrix}_{P \times m} \quad \text{and} \quad T = \begin{bmatrix} t_1^T \\ \vdots \\ t_L^T \end{bmatrix}_{N \times m} \tag{9}$$

H is the hidden layer output matrix of SLFN, Huang et. al. [8,9] proposed ELM algorithm which instead of adaptively tuning the node parameters (w_i, b_i) of (SLFN), randomly assigns them. This random assignment fixes the nonlinearities of the network without iterations. The parameters (w_i, b_i) remain constant after random generation. Through the training of SLFN, least square solution of $\hat{\beta}$ of the linear system $H\beta = T$:

$$\|H\hat{\beta} - T\| = \min_{\beta} \|H\beta - T\| \tag{10}$$

As the number of hidden nodes are much less than the number of training samples (i.e. $P \ll N$), H is a non-square matrix. Hence solution of the smallest norm least square problem is,

$$\hat{\beta} = H^\dagger T \tag{11}$$

where, H^\dagger is the Moore-Penrose generalised inverse of matrix H .

ELM Algorithm: For a given training set $\mathbb{N} = (y_i, t_i) | y_i \in R^n, t_i \in R^m, i = 1, \dots, N$, hidden node output function $A_f(w_i, b_i, y)$, hidden node number P ,

1. Generate hidden node parameters randomly, i.e. $(w_i, b_i), i=1,2,\dots,P$.
2. Calculate output matrix H for the hidden layer.
3. Calculate β , the output weight vector : $\hat{\beta} = H^\dagger T$

3 Proposed Technique

This section describes the use of ELM as a regression model to perform ECG artifact removal from the contaminated EEG. Figure 1 illustrates the system architecture of the proposed ELM-based ECG artifact removal approach. The main concept is to use an ELM model to remove ECG noise to obtain the clean EEG. The overall system includes training and testing stages.

During the training stage, a set of ECG contaminated EEG (EEG_{cont}) - clean EEG (EEG_{act}) pairs are prepared. The EEG_{cont} - EEG_{act} signal with N samples pairs are first converted into the time-frequency domain using the ST.

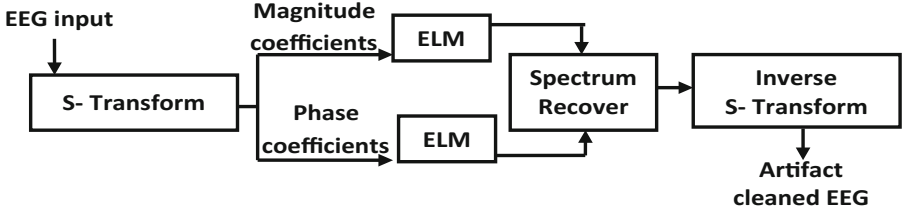


Fig. 1. Block diagram of the proposed methodology

The simultaneous localization of the real and imaginary spectra i.e. magnitude and phase coefficients along with the frequency dependent resolution is provided by ST. From the obtained ST transformed signal (a $N \times M$ matrix or $S_{N \times M}$ matrix), the magnitude coefficients and phase coefficients are separated. Two ELM routines with 20 hidden nodes are used to train the magnitude and phase coefficients separately. The aim of the ELM as a regression model is to predict the clean EEG from the contaminated EEG. The coefficients of $S_{N \times M}$ matrix of contaminated and clean EEG signal are kept at the input and output of the ELM model respectively. H is the hidden layer output matrix, calculated using Radial basis function (RBF) as the output function. $\hat{\beta}$ is estimated using Eq. (11). In the testing phase, the magnitude and phase coefficient separated S matrix of the contaminated EEG signal is given as the input to the two ELM layer respectively. The output of both the ELM is combined to obtain the estimated ST (S_{est}), the regressed contaminated EEG signal. The inverse ST of the S_{est} gives the ECG artifact cleaned EEG.

4 Result Analysis

4.1 Dataset Used for Algorithm Evaluation:

EEG recordings from Cyclic Alternating Pattern (CAP) sleep database of Physionet [6] is used to generate simulated contaminated EEG. The recordings are registered at the Sleep Disorders Center, Ospedale Maggiore of Parma, Italy and available publicly. The simulated EEG recordings are generated by adding corresponding ECGs with SNR 10. A segment of simulated EEG used for the algorithm evaluation is shown in the Fig. 2.

4.2 Discussion

To evaluate the cleaned EEG signal obtained after proposed ELM based regression, three parameters viz. root mean square error ($RMSE$), correlation coefficient (CC) and signal to artifact ratio (SAR) are used. Mathematically $RMSE$ is defined as,

$$RMSE = \sqrt{\frac{1}{N} (EEG_{act} - EEG_{est})^2} \quad (12)$$

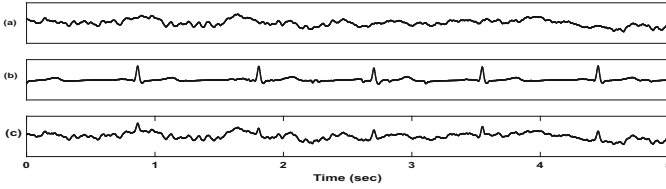


Fig. 2. *Simulated EEG Data:*(a) Uncontaminated EEG, (b) Contaminating ECG, (c) Contaminated EEG

where, EEG_{act} is the actual clean EEG and EEG_{est} is the estimated clean EEG signal through proposed ELM regression. N is the length of the signal.

The correlation factor CC is defined as the correlation coefficient between EEG_{act} and EEG_{est} using proposed technique.

Signal to artifact ratio (SAR) is defined by Eq. 13, where std is the standard deviation.

$$SAR = 10 \log \left(\frac{std(EEG_{cont})}{std(EEG_{cont} - EEG_{est})} \right) \tag{13}$$

The Table 1 shows the performance of the algorithm in terms of $RMSE$ and CC for various EEG signals contaminated with ECG. The mean $RMSE$ and CC obtained are 0.32 and 0.96 respectively. Table 2 shows an improved SAR performance of the proposed algorithm over algorithm proposed by Patel et. al. [16].

Table 1. Performance analysis for different dataset with SNR 10 dB

EEG record	RMSE	CC
EEG1	0.27	0.95
EEG2	0.28	0.96
EEG3	0.32	0.96
EEG4	0.32	0.96
EEG5	0.32	0.95
EEG6	0.32	0.96
EEG7	0.32	0.96
EEG8	0.38	0.96
Mean±Stdv	0.32±0.03	0.96±0.003

Table 2. Comparative SAR performance

EEG record	SAR	
	Proposed	EEMD [16]
EEG1	11.02	10.32
EEG2	13.32	11.20
EEG3	9.75	8.60
EEG4	14.09	9.64
EEG5	13.61	11.02
EEG6	10.57	9.76
EEG7	13.39	9.95
EEG8	9.87	10.94
Mean	11.95	10.18

The graphical illustration of the ECG artifact cleaned EEG signals from the testing phase with corresponding actual EEG signals and contaminated EEG signals are given in the Figs. 3 and 4. As the graphical illustration shows, the spiked

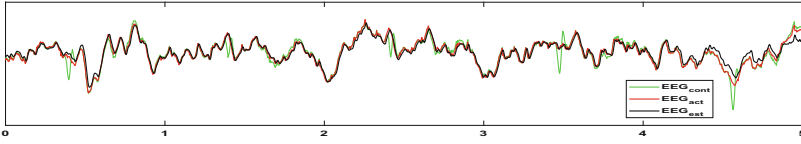


Fig. 3. Estimated EEG with $RMSE = 0.28$, $CC = 0.96$

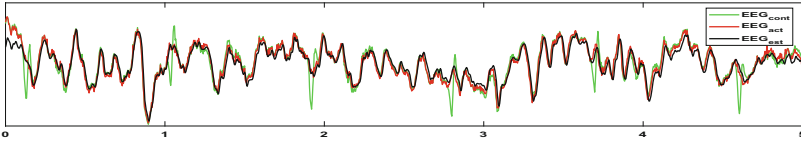


Fig. 4. Estimated EEG with $RMSE = 0.26$, $CC = 0.94$

activity of ECG artifact present in the EEG_{cont} are very effectively minimized. The EEG_{est} is approaching the EEG_{act} closely.

5 Conclusion

In this paper, ELM regression model is used for ECG artifact minimization from contaminated EEG without requiring reference ECG channel. ELM randomly assigns for the input weights and hidden layer biases for the set of hidden nodes. The output layer weights are determined by using the least square method. The whole process is without iteration and has an improved learning and neural network generalization ability than other SLFN approaches. The ST of the EEG time series in two sets i.e. magnitude and phase coefficients provides the ELM with input for training and testing. ST with its good frequency resolution with magnitude and phase localizing ability provides a good feature set for ELM. The results obtained from the testing phase of the proposed ELM based regression model for estimating the clean EEG signal with a low $RMSE$, high CC , and improved SAR . As a scope of future work, the performance of the defined problem can be further improved using variants of ELM.

References

1. Cho, D., Lee, B.: Optimized automatic sleep stage classification using the normalized mutual information feature selection (NMIFS) method. In: 2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 3094–3097. IEEE (2017)
2. Cui, G., Xia, L., Tu, M., Liang, J.: Automatic classification of epileptic electroencephalogram based on multiscale entropy and extreme learning machine. *J. Med. Imag. Health Inform.* **7**(5), 949–955 (2017)
3. Devuyt, S., Dutoit, T., Stenuit, P., Kerkhofs, M., Stanus, E.: Cancelling ECG artifacts in EEG using a modified independent component analysis approach. *EURASIP J. Adv. Signal Process.* **2008**, 1–14 (2008)

4. Dirlich, G., Vogl, L., Plaschke, M., Strian, F.: Cardiac field effects on the EEG. *Electroencephalogr. Clin. Neurophysiol.* **102**(4), 307–315 (1997)
5. Duan, L., Bao, M., Miao, J., Xu, Y., Chen, J.: Classification based on multilayer extreme learning machine for motor imagery task from EEG signals. *Proc. Comput. Sci.* **88**, 176–184 (2016)
6. Goldberger, A.L., et al.: PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation* **101**(23), e215–e220 (2000). <http://circ.ahajournals.org/content/101/23/e215.fullPMID:1085218>; <https://doi.org/10.1161/01.CIR.101.23.e215>. (June 13), *circulation Electronic Pages*:
7. Hamaneh, M.B., Chitravas, N., Kaiboriboon, K., Lhatoo, S.D., Loparo, K.A.: Automated removal of EKG artifact from EEG data using independent component analysis and continuous wavelet transformation. *IEEE Trans. Biomed. Eng.* **61**(6), 1634–1641 (2014)
8. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: a new learning scheme of feedforward neural networks. In: *Proceedings 2004 IEEE International Joint Conference on Neural Networks, 2004*, vol. 2, pp. 985–990. IEEE (2004)
9. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: theory and applications. *Neurocomputing* **70**(1–3), 489–501 (2006)
10. Hussain, T., Siniscalchi, S.M., Lee, C.C., Wang, S.S., Tsao, Y., Liao, W.H.: Experimental study on extreme learning machine applications for speech enhancement. *IEEE Access* **5**, 25542–25554 (2017)
11. Jafarifarmand, A., Badamchizadeh, M.A.: Artifact removal in EEG signal using a new neural network enhanced adaptive filter. *Neurocomputing* **103**, 222–231 (2013)
12. Liang, Y., Leung, C., Miao, C., Wu, Q., McKeown, M.J.: Automatic sleep arousal detection based on C-ELM. In: *2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, vol. 2, pp. 376–382. IEEE (2015)
13. Lin, Q., Ye, S., Wu, C., Gu, W., Wang, J., Zhang, H.L., Xue, Y.: A novel framework based on biclustering for automatic epileptic seizure detection. *Int. J. Mach. Learn. Cybern.* pp. 1–13 (2017)
14. Liu, Q., Zhao, X., Hou, Z., Liu, H.: Epileptic seizure detection based on the kernel extreme learning machine. *Technol. Health Care* **25**(S1), 399–409 (2017)
15. Odelowo, B.O., Anderson, D.V.: Speech enhancement using extreme learning machines. In: *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 200–204. IEEE (2017)
16. Patel, R., Gireesan, K., Sengottuvel, S., Janawadkar, M., Radhakrishnan, T.: Common methodology for cardiac and ocular artifact suppression from EEG recordings by combining ensemble empirical mode decomposition with regression approach. *J. Med. Biol. Eng.* **37**(2), 201–208 (2017)
17. Stockwell, R.G., Mansinha, L., Lowe, R.: Localization of the complex spectrum: the S-transform. *IEEE Trans. Sig. Process.* **44**(4), 998–1001 (1996)
18. Tan, P., Sa, W., Yu, L.: Applying extreme learning machine to classification of EEG BCI. In: *2016 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pp. 228–232. IEEE (2016)
19. Tan, P., Tan, G.Z., Cai, Z.X., Sa, W.P., Zou, Y.Q.: Using ELM-based weighted probabilistic model in the classification of synchronous EEG BCI. *Med. Biol. Eng. Comput.* **55**(1), 33–43 (2017)
20. Waser, M., Garn, H.: Removing cardiac interference from the Electroencephalogram using a modified Pan-Tompkins algorithm and linear regression. In: *35th Annual International Conference of IEEE EMBS*, pp. 2028–2031. July 2013



Suggesting Cooking Recipes Through Simulation and Bayesian Optimization

Eduardo C. Garrido-Merchán^(✉) and Alejandro Albarca-Molina^(✉)

Universidad Autónoma de Madrid, Francisco Tomás y Valiente 11, Madrid, Spain
eduardo.garrido@uam.es, alejandro.albarca@estudiante.uam.es

Abstract. Cooking typically involves a plethora of decisions about ingredients and tools that need to be chosen in order to write a good cooking recipe. Cooking can be modelled in an optimization framework, as it involves a search space of ingredients, kitchen tools, cooking times or temperatures. If we model as an objective function the quality of the recipe, several problems arise. No analytical expression can model all the recipes, so no gradients are available. The objective function is subjective, in other words, it contains noise. Moreover, evaluations are expensive both in time and human resources. Bayesian Optimization (BO) emerges as an ideal methodology to tackle problems with these characteristics. In this paper, we propose a methodology to suggest recipe recommendations based on a Machine Learning (ML) model that fits real and simulated data and BO. We provide empirical evidence with two experiments that support the adequacy of the methodology.

Keywords: Bayesian optimization · Cooking · Simulation

1 Introduction

Creating good quality cooking recipes is an ancient art that has been developed by a wide variety of cultures. Nowadays, technology and modelling bring us new tools to experiment and test cooking recipes. Some solutions that have been studied are synthesizing cooking recipes using video [4] or recommend cooking recipes based on preferences of a user [13]. But a few solutions tackle the problem of generating an optimal cooking recipe based on an input space of ingredients, tools and other criteria.

Lots of variables and decisions are involved in the process. From which ingredients to use and in which quantity to what tools to use, cooking times or intensity of the home appliances. We may use an optimization framework to model this problem. Let all these parameters lie in a space represented by Θ , where $\theta \in \Theta$ is a single configuration of that parameter space. We can model this space by establishing the parameters that are going to be used and the ranges that they fit in. For example, for a pasta dish, we may consider the boiling time minutes, [5, 15], and the quantity of used pasta grams [100, 300]. $\theta = [8, 150] \in \Theta$ would be a valid recipe that considers 8 boiling minutes and 150 pasta grams.

All these parameters generate a cooking recipe that has to be evaluated. This evaluation is subjective. Let the quality of a recipe be represented by a scalar variable $\lambda \in [0, 10]$, with 0 being low quality and 10 being high quality. If two or more individuals are given a recipe, their evaluation may differ. More formally, let $\lambda = \mathbf{e}(\theta)$ be a vector of evaluators, where each $\lambda_i = f_i(\theta)$ represents the evaluation of a single evaluator and θ is the configuration of a recipe. Then, every λ_i may be different w.r.t another one contained in λ . We have a latent function f contaminated by noise, $y(\theta) = f(\theta) + \epsilon : \epsilon \sim \mathcal{N}(\mu, \sigma)$. Other characteristics of the cooking process are that the evaluation is very expensive both in time, human and other resources. In this setting, we cannot depend on an optimization framework that needs lots of iterations to deliver a good solution to the problem, like metaheuristics. Moreover, in this problem, we do not have access to gradients as there is no analytical expression that models the cooking process. Hence, we can not use classical optimization techniques that find an optimal solution using gradients.

Bayesian Optimization (BO), described in a friendly way in this tutorial [1], emerges as a useful solution for problems with the described characteristics. BO relies on modelling the response surface by a probabilistic model, typically a Gaussian Processes (GP) [10], generating a surrogate model that is cheap to evaluate and gives predictions for all the input space that consider uncertainty. From this GP, BO builds an acquisition function that represents the utility of evaluating each point in order to find the optimum to the problem.

In a real case scenario, where evaluations are performed while the optimization process is being done, BO is an adequate solution. But by only using BO, we need to cook as many times as iterations we need to perform, and for generating prototypes of recipes, that may be expensive in infrastructure and human resources. In this paper, we proposed a methodology to overcome this issue by not cooking for every iteration of the BO process, we will only need to cook before the process starts in order to build a Dataset.

We propose a methodology for generating prototype cooking recipes where we have simulated the cooking process by a fitted ML model, m , that has approximated, y_{approx} , the cooking evaluation, y , by having learned it from data \mathcal{D} . This data \mathcal{D} is the union of real data \mathcal{D}_{real} and synthetic data \mathcal{D}_{sim} generated by simulation. We have used these approximations to the evaluation and the data as we have the restriction of not having the human and infrastructure resources for evaluating each cooking recipe nor cooking thousands of meals. On the other hand, we have used BO since our objective is to corroborate the hypothesis that using BO, [7], for this kind of problem is a good solution and that it also serves to optimize the cooking process if it is approximated by a fitted ML model, m .

The rest of the paper is divided as follows: Sect. 2 describes related work in this topic. Section 3 is an overview of BO basics. Section 4 presents the methodology that we have proposed to tackle this problem. Then, an experiments section provides empirical evidence that supports the adequacy of the proposed methodology. Finally, we sum up with Conclusions and Further Work around this topic.

2 Related Work

There is little work in this area, but some systems have been proposed to give a solution to this problem. The first one is IBM Chef Watson: [9], which relies on the IBM Watson system that won the Jeopardy Contest. This system process natural language texts and uses predictive modelling to provide cooking recipes. It has the disadvantages that it relies on this private platform and it needs lots of data to fit the preferences of the user. Google Vizier, [7], is a free platform for BO proposed by Google. They have used this tool to provide a recipe for chocolate cookies by having them tested by employees each day in their offices. Our approach differs in that we do not need to wait for several weeks to suggest a good quality cooking recipe. With our approach, we can provide a good quality cooking recipe in a single day if we have enough human resources to evaluate a set of meals.

3 Black-Box Bayesian Optimization

BO has been used in a plethora of scenarios: From Hyperparameter Tuning of ML Algorithms [2, 12] to renewable energies [3] and a wide variety of applications [11]. BO process follows an iterative scheme where it uses a probabilistic model as a surrogate model, typically a Gaussian Process (GP) which is a prior over functions. This model serves as a prior for the objective function that we are optimizing, that is, we assume that the function can be sampled from a GP.

The GP [10] is a non-parametric model defined by a mean vector μ and a covariance function $K(\mathbf{x}, \mathbf{x}')$ that defines a covariance matrix \mathbf{K} , $f(\mathbf{x}) \approx GP(\mu, \mathbf{K})$. By using BO we are assuming that the objective function $o(\mathbf{x})$ that we want to optimize can be sampled from the GP, which is going to be fitted by the evaluations $\mathcal{D}\{\mathbf{X}, \mathbf{y}\}$ that we will be performing in the iterative scheme. The predictive distribution of the GP defines a mean μ_i and a standard deviation σ_i for every point \mathbf{x} where the objective function can be evaluated $y_i = o(\mathbf{x}) \approx N(\mu_i, \sigma_i)$, in other words, we have an uncertain prediction $N(\mu_i, \sigma_i)$ of every point of the space \mathbf{x} where the objective function can be evaluated $y_i = o(\mathbf{x})$. The equations for the predictive distribution of the GP are:

$$\begin{aligned} \mu &= \mathbf{k}_*^T (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{y}, \\ \sigma^2 &= k(\boldsymbol{\theta}_t, \boldsymbol{\theta}_t) - \mathbf{k}_*^T (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{k}_*. \end{aligned} \quad (1)$$

where $\mathbf{y} = (y_1, \dots, y_{t-1})^t$ is a vector with the objective values observed so far; σ_n^2 is the variance of the additive Gaussian noise; \mathbf{k}_* is a vector with the prior covariances between $f(\boldsymbol{\theta}_t)$ and each y_i ; \mathbf{K} is a matrix with the prior covariances among each y_i ; and $k(\boldsymbol{\theta}_t, \boldsymbol{\theta}_t)$ is the prior variance at the candidate location $\boldsymbol{\theta}_t$. The covariance function $k(\cdot, \cdot)$ is pre-specified; for further details about GPs and example of covariance functions we refer the reader to [10].

In every iteration, the BO algorithm fits the GP with the point recommended from BO in the previous step, \mathbf{x}^* . This recommendation is obtained by the construction of an acquisition function, $\alpha(\mathcal{X})$, which is built from the predictive

distribution $p(\mathcal{X})$ in every point of the input space $p(\mathbf{x})$ of the GP $\alpha(p(\mathbf{x}))$. The suggestion for the next evaluation is given by the maximum of this function $\mathbf{x}^* = \arg \max(\alpha(\mathcal{X}))$, where $\mathcal{X} \in \mathbb{R}^d$ is the bounded d -dimensional input space. There exist different criteria for this acquisition function $\alpha(\cdot)$ that basically tries to obtain a balanced tradeoff between exploitation of promising surfaces of the input space and exploration of unknown surfaces. Some examples of acquisition functions are the Expected Improvement (EI) [1] or the Predictive Entropy Search (PES) [8]. The key facts from BO are that the process of optimizing this acquisition function is cheap, it is typically done by extracting the best point from a grid and then starting there a local search algorithm such as the L-BFGS algorithm.

Once \mathbf{x}^* is computed, it is added to the previous datasets of observations $\mathcal{D}\{\mathbf{X}, \mathbf{y}\}$, the GP is conditioned over that point and the cycle starts again, building another acquisition function and extracting the maximum. This process is repeated until a convergence criterion is satisfied, typically when the budget of evaluations is satisfied. The final suggestion given by BO can be extracted by the best observation made so far. More details of BO can be found in this tutorial [1].

4 Proposed Methodology: Simulation and Bayesian Optimization

In a real case scenario, a possible methodology is to cook a recipe with a certain configuration θ , evaluate it by a jury of experts $y = f(\theta)$ and give that evaluation to BO, which will provide another suggestion to be cooked. As this is a very tedious, but possible scenario for restaurants, process and because our objective is to provide a prototype of an optimal recipe θ^* , we will approximate the evaluation function $f(\theta)$ given by experts by the prediction of an ML algorithm $m(\theta) \approx f(\theta)$, which will be fitted by the union of real \mathcal{D}_{real} and simulated data \mathcal{D}_{sim} .

In order to obtain prototypes of cooking recipes $\theta \in \Theta$, we need to follow the following steps. First of all, we need to obtain to define the bounded input space of parameters that are going to be involved in the cooking recipe $\Theta \in \mathbb{R}^d$. Example of parameters $\theta \in \Theta$ could be the temperature of the oven or the meat grams used. We need to specify the lower l_i and upper u_i limits of each variable. In the BO process, GPs assume real variables but in the kitchen, we can find integer variables, intensity of the glass-ceramic or categorical variables, an ingredient brand. In order to overcome this issue, we use the approach given by Garrido [6].

To build a dataset of real data \mathcal{D}_{real} , we must use a uniform grid over the input space Θ in order to cook meals from diverse configurations. For the evaluation function $y = f(\mathbf{x}) \in [0, 10]$, we have chosen a jury of N experts to provide an evaluation y_i for every recipe. The final evaluation $y = f(\mathbf{x})$ of every recipe is given by the mean of these experts $y = \frac{\sum_{i=1}^N y_i}{N}$.

In this process, the chefs will acquire expert knowledge Φ about the different variables ϕ involved in every different recipe and how they link to the quality y . We propose to encode this expert knowledge Φ into a set of conditional probability density functions Ξ to augment the dataset \mathcal{D}_{real} by sampling the quality y from them. For example, we know that if the temperature of the oven is very high, the quality of the recipe will be low with high security, we can model this expert knowledge by a Gamma Distribution $y \sim \Gamma(\cdot|\theta_1)$. Other distributions such as Gaussian have also been used, to model for example that a particular cooking time frame generated good quality meals $y \sim \mathcal{N}(\cdot|\theta_2)$. We fit the parameters of these conditional distributions Ξ by real data or by expert knowledge. By sampling points from these distributions Ξ , we provide the latent noise of the objective function $y(\theta) = f(\theta) + \epsilon : \epsilon \sim \mathcal{N}(\mu, \sigma)$. We sample points from this set of distributions Ξ to build the dataset fitted by the ML model $\mathcal{D} = \mathcal{D}_{sim} \cup \mathcal{D}_{real}$.

We now fit an ML model m and its hyperparameters ρ to this data, choosing the best ML model $m = \operatorname{argmin} v(\mathbf{M})$. In order to do so, we propose to use 10-fold cross validation $v(m)$ over a pipeline of ML models \mathbf{M} . We can use BO, random or grid search to find the best model. The approximation to the jury of experts evaluation of the taste of the recipes will be given by the prediction of the chosen ML algorithm $m(\theta) \approx f(\theta)$. Finally, we use BO having as objective function the prediction of the fitted ML model $m(\theta)$ of the recipes θ , avoiding the use of statistical tests as BO is an efficient search mechanism in this scenario. The suggested recipe θ^* will be given by the best observation found by BO in a certain number of iterations. By following this methodology, we obtain a way to optimize a recipe and suggest a prototype without human intervention nor the need to cook in the optimization process.

5 Experiments

We have performed two experiments to show the results of our proposed methodology. More experiments can be done following these steps, but these two combine real, integer and categorical variables, Table 1, resulting in a complete coverage of the problem. The first experiment involves the cooking process of a Hot Dog and the second one of a Cesar Salad. For both problems, 45 different meals have been cooked and tested by experts. Then, we have codified the gained expert knowledge into probability distributions, where we have generated more instances.

We have fixed an SVR model for the generated data. In order to find the best values of the hyperparameters of this model that generate the lowest medium squared error of a 10 fold cross-validation over the generated dataset, we have used a grid search, testing different values for C and γ . For both experiments, the optimum hyperparameters have been $C = 1$ and $\gamma = 0.01$. We obtain errors of 2.56 and 2.6 points, which are very reasonable, since the probability distributions add noise to the evaluation function to model the subjectivity of this function.

Having fixed the prediction model, we know are going to test our hypothesis that states that using BO we will improve the quality of the recipe given by

Table 1. Input space of the performed experiments.

Cesar Salad		Hot Dog	
Variable name	Bounds	Variable name	Bounds
Sausage cooking time (s.)	[0, 200]	Ceramic intensity (bread)	[1, 9]
Bread cooking time (s.)	[0, 200]	Bread cooking time (s.)	[5, 50]
Cooking place	[pan, microwave]	Ceramic intensity (chicken)	[1, 9]
Bbq sauce teaspoons	[0, 9]	Chicken cooking time (s.)	[1, 200]
Mayonnaise teaspoons	[0, 9]	Cesar brand	[X, Y]
Mustard teaspoons	[0, 9]	Lettuce brand	[X, Y, Z, T]

an expert. We predict the quality of the expert criterion using the model for both experiments. We also compare the BO optimization with a Random Search (RS). For BO, we have used a Matérn Kernel for the GP. We optimize the hyperparameters of the GP by maximizing their log-marginal likelihood. We use 10 GP samples to build an averaged PES acquisition function. In Fig. 1, we plot the quality of the recipe in every iteration of the process. We show how BO outperforms the expert and random search criterion easily, which is an empirical proof of the good behaviour of BO in these scenarios.

Figure 2 shows us histograms of the suggestions given by BO for all the 100 replications of the experiment. The suggested recipe given by the optimization methodology is formed by the union of the most suggested values of all the

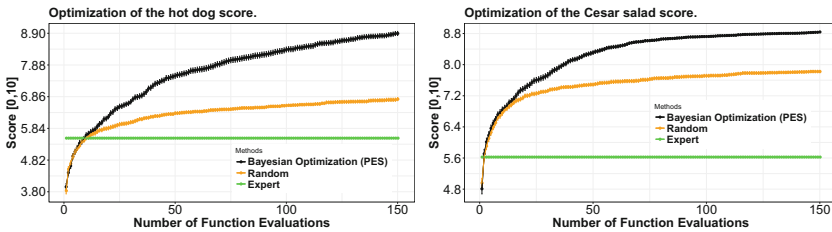


Fig. 1. Average results of the quality of the two real experiments by BO, RS and Expert Criterion approaches.

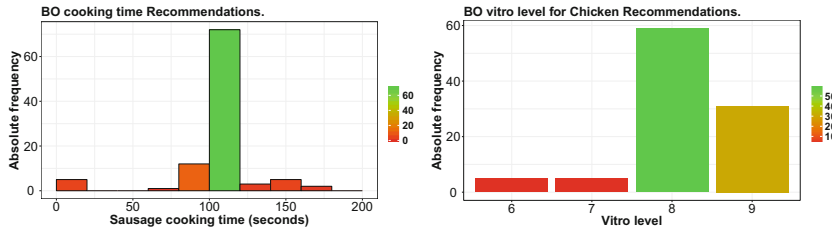


Fig. 2. Two histograms of proposed values for the variable of the two real experiments by the 100 replications of BO performed for the two experiments.

variables, shown in Table 2. The suggested values have been tested a posteriori and the experts agree on the quality of the meals given by BO.

Table 2. Recipes generated by the best observed result of BO. As the result is the most voted amongst 100 replications, bins have been created for integer and real-valued variables. Most voted bins are shown in the table.

Cesar Salad		Hot Dog	
Variable name	Value	Variable name	Value
Sausage cooking time (s.)	[45, 55]	Ceramic intensity (bread)	9
Bread cooking time (s.)	[100, 115]	Bread cooking time (s.)	[12, 15]
Cooking place	pan	Ceramic intensity (chicken)	8
Bbq sauce teaspoons	0	Chicken cooking time (s.)	[70, 100]
Mayonnaise teaspoons	0	Cesar brand	X
Mustard teaspoons	0	Lettuce brand	Y

6 Conclusions and Further Work

We have proposed a methodology for generating cooking recipes from an input space of variables such as ingredients, tools and other variables. As we are restricted to our human and time resources we have simulated the process of cooking by an ML model, an SVR, that fits real and synthetic data generated by expert knowledge. The choices of the hyperparameters for BO, the ML model and the probability distributions mentioned in this methodology can be varied depending on the problem at hand. We can even use a metaheuristic such as a genetic algorithm or a particle swarm optimization to optimize instead of BO, but in a real case scenario, we will need BO due to the fact that the evaluations are too expensive both in time and human resources.

We have provided two real experiments that show the benefits of using BO to optimize the input space of cooking variables and compared it to a RS and an Expert Criterion. We replicate these experiments and provide the most voted recommendation by the BO approach. In both scenarios, BO outperforms the results obtained by the other approaches. Other objectives could have been modelled such as, for example, the presentation of the cooking recipe. Restrictions can also appear in the cooking process such as the time needed to elaborate a recipe or the total price of the recipe. These characteristics could be modelled by a Multi-Objective Constrained scenario that can be solved by Multi-Objective Constrained BO [5]. We are also planning to implement an interface that, given an input space and choices such as the optimization algorithm, the ML model and the expert knowledge rules it suggests a recipe written in natural language.

Acknowledgments. The authors acknowledge the use of the facilities of Centro de Computación Científica (CCC) at Universidad Autónoma de Madrid. The authors also acknowledge financial support from the Spanish Plan Nacional I+D+i, Grants TIN2016-76406-P, TEC2016-81900-REDT (MINECO/FEDER EU), and from Comunidad de Madrid, Grant S2013/ICE-2845.

References

1. Brochu, E., Cora, V. M., De Freitas, N.: A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv preprint [arXiv:1012.2599](https://arxiv.org/abs/1012.2599) (2010)
2. Córdoba, I., Garrido-Merchán, E.C., Hernández-Lobato, D., Bielza, C., Larrañaga, P.: Bayesian optimization of the PC algorithm for learning Gaussian Bayesian networks. arXiv preprint [arXiv:1806.11015](https://arxiv.org/abs/1806.11015) (2018)
3. Cornejo-Bueno, L., Garrido-Merchán, E.C., Hernández-Lobato, D., Salcedo-Sanz, S.: Bayesian optimization of a hybrid system for robust ocean wave features prediction. *Neurocomputing* **275**, 818–828 (2018)
4. Doman, K., Kuai, C.Y., Takahashi, T., Ide, I., Murase, H.: Video cooking: towards the synthesis of multimedia cooking recipes. In: Lee, K.-T., Tsai, W.-H., Liao, H.-Y.M., Chen, T., Hsieh, J.-W., Tseng, C.-C. (eds.) *MMM 2011*. LNCS, vol. 6524, pp. 135–145. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-17829-0_13
5. Garrido-Merchán, E.C., Hernández-Lobato, D.: Predictive entropy search for multi-objective Bayesian optimization with constraints. arXiv preprint [arXiv:1609.01051](https://arxiv.org/abs/1609.01051) (2016)
6. Garrido-Merchán, E.C., Hernández-Lobato, D.: Dealing with categorical and integer-valued variables in Bayesian optimization with Gaussian processes. arXiv preprint [arXiv:1805.03463](https://arxiv.org/abs/1805.03463) (2018)
7. Golovin, D., Solnik, B., Moitra, S., Kochanski, G., Karro, J., Sculley, D.: Google vizier: a service for black-box optimization. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1487–1495. ACM (2017)
8. Hernández-Lobato, J.M., Hoffman, M.W., Ghahramani, Z.: Predictive entropy search for efficient global optimization of black-box functions. In: *Advances in Neural Information Processing Systems*, pp. 918–926 (2014)
9. Lohr, S.: And now, from IBM, it's chef Watson. *New York Times* (2013)
10. Rasmussen, C.E.: Gaussian processes in machine learning. In: Bousquet, O., von Luxburg, U., Rätsch, G. (eds.) *ML -2003*. LNCS (LNAI), vol. 3176, pp. 63–71. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28650-9_4
11. Shahriari, B., Swersky, K., Wang, Z., Adams, R.P., De Freitas, N.: Taking the human out of the loop: a review of Bayesian optimization. *Proc. IEEE* **104**(1), 148–175 (2016)
12. Snoek, J., Larochelle, H., Adams, R.P.: Practical Bayesian optimization of machine learning algorithms. In: *Advances in Neural Information Processing Systems*, pp. 2951–2959 (2012)
13. Ueda, M., Takahata, M., Nakajima, S.: User's food preference extraction for personalized cooking recipe recommendation. In: *Workshop of ISWC*, pp. 98–105 (2011)



Assessment and Adaption of Pattern Discovery Approaches for Time Series Under the Requirement of Time Warping

Fabian Kai-Dietrich Noering¹(✉), Konstantin Jonas¹,
and Frank Klawonn²

¹ Volkswagen Group Research, 38436 Wolfsburg, Germany
{fabian.kai-dietrich.noering,
konstantin.jonas}@volkswagen.de

² Ostfalia University of Applied Sciences,
Salzdahlumer Str. 46/48, 38302 Wolfenbüttel, Germany
f.klawonn@ostfalia.de

Abstract. In the automotive industry, the cars themselves as well as the production lines, produce a high amount of data day by day. To get information out of these data there is a need for high performance data mining tools. One of these tools is the pattern discovery. This paper addresses the assessment of different approaches to discover frequent pattern in time series. Our special requirement is the detection of time warped pattern with variable length. The comparison includes approaches based on dynamic time warping (DTW), discretization as well as Keogh's Matrix Profile. Every approach is exemplarily implemented in MATLAB and (if necessary) adapted to face our use cases. The focus of the assessment will be the quality of the results, the runtime and the effort of parametrization. For evaluation, time series test datasets are generated with predefined patterns based on random walks. The output patterns, identified by the different pattern discovery algorithms, are compared with the initial patterns and evaluated with respect to the Jaccard index. This leads to a quality score for every algorithm and every parametrization and the possibility to compare different algorithms as well as approaches.

Keywords: Pattern discovery · Motif discovery · Pattern enumeration
Grammar induction · Pattern evaluation · Time series · Time warping
Unsupervised

1 Motivation

During the operation of complex mechatronic systems various time-resolved data is accumulated. The analysis of such time series data provides information on usage, condition or even misbehavior of the systems or components. Focusing on fault diagnosis, usually only short error logs are analyzed with regard to a possible abnormal signal behavior. However, to obtain long-term information on usage, wear-off, behavior or system's condition an analysis of the entire data record is required. In this context, change detection methods are needed to identify slow or abrupt changes in the

signals or signal groups behavior. In the context of wear-off for systems having recurring stress patterns, e.g. manufacturing robots, it is advantageous to compare only such patterns. In our use case, we cannot expect experts to specify suitable patterns and therefore we need unsupervised methods for automatic definition of such recurring stress patterns without input of expert knowledge.

2 Introduction

Although there are a lot of different approaches and algorithms for the problem of (unsupervised) pattern discovery in time series data, the number of useful approaches limits with the requirements of our special use cases. We need reliable methods to detect recurring patterns, which are usually time warped and noisy, in datasets with far more than one million data points at a reasonable runtime. Beyond that we also would like to detect patterns in multidimensional time series. However, in this paper we limit our analysis for simplicity to a one dimensional case. Following the conditions described above we will compare three different approaches. There may be the need of adaption to fulfill the requirements, e.g. because some approaches aim to discover motifs¹, while our goal is the pattern² discovery.

At first there are algorithms based on the principle of *dynamic time warping* (DTW). DTW is a distance or similarity measurement for time series with different length with the possibility to stretch or compress one of the time series to fit the other one [1]. Beyond that, the DTW can be adjusted and extended to the problem of discovering patterns. The authors of [2] proposed a DTW-based algorithm called *CrossMatch* to identify similar subsequences in different sequences without the need of sequences. A similar technique is described in [3] where the approach is extended by a hierarchical clustering algorithm to find patterns.

The second approach is based on the symbolic representation of the time series, which is interesting in terms of reduction of complexity and therefor also for the reduction of the runtime. Furthermore these approaches can be more robust against noise than those based on the DTW. To get a symbolic representation, there are many methods for time series discretization. The *Symbolic Aggregate Approximation* (SAX) [4] algorithm and other symbolic time series methods use intervals with equal probability. Besides, there is the possibility to use intervals of equal size. The relevance of symbolic approaches is evidenced by the variety of existing pattern discovery algorithms from various research fields like bioinformatics or text mining. For example in bioinformatics the *Smith Waterman* algorithm is used for sequence alignment, hence the pattern discovery in DNA sequences [5]. As DTW, Smith Waterman is based on dynamical programming and is able to find time warped patterns. The *Sequitur* algorithm, proposed in [6], extracts the hierarchical structure of a symbolic sequence by replacing recurring phrases with grammatical rules while forming a dictionary of these rules, which is formally known as grammar induction. Sequitur identifies only identical

¹ Definition "motif": A motif is a pair of similar subsequence in a time series.

² Definition "pattern": A pattern is a group of at least two similar subsequences in a time series.

patterns and ignores possible similarities in rules/patterns. Furthermore it is not guaranteed to find all the patterns in the sequence, because of its replacing procedure. Pattern enumeration is often referenced to cope with last mentioned problem. To perform time warped pattern discovery with those algorithms there have been proposed *numerosity reduction* techniques, as e.g. employed in [7]. Especially for long sequences the number of enumerated patterns can increase exponentially. That is why there is also research in constraint programming techniques like in [8] or [9].

The last approach we want to focus on is the principle of *Matrix Profile* introduced by Keogh et al. [10]. Basically, it is a brute force approach to find motifs or patterns of a fixed length by calculation the distances between every possible query sequences and every other sequence of the chosen length. Nevertheless it is said to be fast because of the *MASS* algorithm (Mueen’s ultra-fast Algorithm for Similarity Search) also proposed in [10]. As it uses the z-normalized Euclidian distance, i.e. a fixed length, it is not suitable to find time warped patterns. A possibility to solve this issue is to replace the Euclidian distance with the DTW. However, this makes it even more computational expensive and doesn’t solve the problem to be only able to compare sequences of equal length. Furthermore, because of the z-normalization, it ignores the original values of the compared sequences. That is why we implemented additionally a version of the standard Euclidean distance.

In this paper we focus on the comparison of the aforementioned approaches including necessary adaptations. Our target is an evaluation concerning runtime, parameterization effort and pattern quality. Therefor we apply the Jaccard Index to calculate a quality score called overlap, which gives us the opportunity to evaluate and compare different algorithms as well as approaches.

The following section gives detailed information about the applied algorithms and our exemplary implementations to fulfill our requirements. Section 4 explains our approach for the comparison of pattern discovery algorithms. Section 5 evaluates our results and gives statements to the assessment of the different approaches. The final section gives a conclusion and an outlook to future research.

3 Approaches for Pattern Discovery

3.1 DTW-Based

The tested implementation is based on the approaches from [2] and [3]. We are using the scoring function of the CrossMatch algorithm, which is necessary to compute the matrix equivalent to the DTW algorithm. In the case of CrossMatch the matrix contains values of similarity in contrast to the DTW which computes a distance matrix. The scoring function for every cell (i, j) of the similarity matrix v of CrossMatch using the time series x and y is shown in Eq. (1).

$$v(i, j) = \max \begin{cases} \varepsilon b_d - ||x_i - y_j|| + v(i - 1, j - 1) \\ \varepsilon b_v - ||x_i - y_j|| + v(i, j - 1) \\ \varepsilon b_h - ||x_i - y_j|| + v(i - 1, j) \\ 0 \end{cases} \tag{1}$$

A major difference between the scoring functions of the classic DTW and CrossMatch is the usage of weighting factors ϵb_d , ϵb_v and ϵb_h . It enables us to penalize time warping, i.e. horizontal or vertical steps in the similarity matrix. While DTW is only able to calculate a distance between two sequences, CrossMatch enables the detection of motifs as well. The end of a motif is marked by a local maximum in the similarity matrix v . Furthermore the CrossMatch algorithm calculates a position matrix, which contains the starting points of the motifs. For further explanations of DTW see [11] and CrossMatch see [2].

For the tests that we are going to perform, we limit the weighting factors to $\epsilon b_d = 1$, as a reward for similarity, and $-1 \leq \epsilon b_v = \epsilon b_h \leq 0$, as a possible penalization for time warping. Furthermore we need to make an adaption to the CrossMatch scoring function, see Eq. (2). In order to search for motifs in one and the same time series, we have to exclude the trivial matches. Therefore we implement an offset region around the diagonal of the similarity matrix $v_{adapt}(i, j)$. These matrix cells are set to zero by default. Due to symmetry only half of the matrix has to be calculated. These adaptations also apply to the position matrix.

$$v_{adapt}(i, j) = \begin{cases} v(i, j), & |i - j| > offset \\ 0, & |i - j| \leq offset \end{cases} \quad (2)$$

In order to form patterns, additional to the CrossMatch, we extract and cluster the motifs like in [3]. To extract possible pattern candidates from the matrix, we use a minimum length and a minimum similarity. Every candidate's score higher than the product of *min_length* and *min_similarity* is going to be clustered. The clustering algorithm needs another parameter specifying the sensitivity of the extracted motifs to be clustered to patterns, which is later referenced to *max_similarity_motif*. To ensure, that every subsequence in the time series similar to the pattern is discovered, a representative of every pattern is extracted and searched in the whole time series by recalculating the CrossMatch similarity matrix with the representative query sequence.

3.2 Discretization-Based

Concerning discretization-based approaches for pattern discovery we focus on an adapted Sequitur algorithm and an algorithm for pattern enumeration. Both are based on an equal probability discretization technique. However, in contrast to the technique described in [4], we don't assume a Gaussian distribution of the data values, which is necessary in terms of various signal types. We formed a simple algorithm to form classes of equal count of data points with respect to the constraints *uniqueness* and *allocation*. Uniqueness means that there must not be multiple classes with the same data value. According to the allocation constraint, every data value has to be allocated to a class. The algorithms we will describe are only dependent on one parameter, the number of discretization steps.

Because of the need of time warping we apply an optional numerosity reduction technique like in [7], which we call *symbolic reduction*. Therefore every identical consecutive symbol in the time series is reduced to a unique symbol. Hence in every

step of the time series is a change in the symbolic value. We will later discuss the benefit of this technique.

Pattern Enumeration Algorithm

To evaluate the performance of our adapted Sequitur algorithm, we also want to apply a pattern enumeration algorithm. The algorithm repeatedly scans the whole time series with different query pattern length by using a matrix representation of the symbolic time series and a sort algorithm. In Table 1 the pseudocode of the algorithm is shown.

Table 1. Pseudocode for pattern enumeration

```

1  l_pattern = 1
2  empty = false
3  while ~empty
4      l_pattern = l_pattern + 1
5      ts_matrix = form_matrix(sts, l_pattern)
6      [ts_matrix_sorted, index] = sort(ts_matrix)
7      [symbolic_repr, num, loc] = get_symbolic_repr(ts_matrix_sorted)
8      for p = 1:length(symbolic_repr)
9          if num(p) > 1
10             add_to_dictionary(symbolic_repr(p), location(p))
11         end
12     end
13     if isempty(find(num > 1))
14         empty = true
15     end
16 end

```

The core of the algorithm is in lines 5 to 7. In line 5 the time series array is transformed to a matrix of the size $(length(sts) - l_pattern) \times l_pattern$. $l_pattern$ describes the length of the patterns that are searched in the current iteration of the while loop. In the function *form_matrix* the symbolic time series *sts* gets multiplied $l_pattern$ times and shifted by a number of entries dependent on the column. This leads to matrix rows with consecutive time series values. In line 6 the matrix is sorted along the first dimension, which leads to a matrix where equal row values are located in consecutive rows. By detecting the differences of the sorted matrix within the function *get_symbolic_repr*, the contained symbolic combinations or representations can be extracted. Furthermore the number and the location of the occurrences can be identified. In the last step every detected pattern with a count greater than one, is documented in a dictionary. The dictionary contains every detected pattern and its location.

Adapted Sequitur Algorithm

The Sequitur algorithm is based on two constraints. The first constraint is named *diagram uniqueness*, which means that every diagram (a pair of two adjacent symbols) is allowed to occur only once in the symbolic time series. If a diagram appears more than once, then a rule has to be formed. Every rule has to fulfill the second constraint, the *rule utility*. It says that every rule has to occur at least twice. For detailed information about the execution of the Sequitur algorithm see [6].

Our adaption of the algorithm is shown in Table 2. The first three steps within the WHILE loop are similar to the pattern enumeration algorithm, except for a fixed matrix column size of 2. To fulfill both, the diagram uniqueness and the rule utility, the algorithm adds the diagram that occurs the most to the dictionary and replaces the diagram in the symbolic time series by a new symbol. It stops when the diagram uniqueness is fulfilled.

Table 2. Pseudocode for adapted Sequitur

```

1  empt = false
2  while ~empt
3      ts_matrix = form_matrix(sts, 2)
4      [ts_matrix_sorted, index] = sort(ts_matrix)
5      [symbolic_repr, num, loc] = get_symbolic_repr(ts_matrix_sorted)
6      if max(num) > 1
7          [v, i] = max(num);
8          [Dict, numofrule] = addtodict(Dict, symbolic_repr(i), loc(i));
9          sts = replace_bynewrule(sts, loc, numofrule);
10     else
11         empt = true;
12     end
13 end

```

3.3 Matrix Profile

The Matrix Profile introduced by Keogh et al. is an approach able to be executed on raw time series without any preprocessing. Matrix Profile is based on the calculations of distance profiles. A distance profile visualizes the distances between a query sequence and every other possible sequence in a time series (with $length(time-series) \gg length(query)$) while the query sequence is part of the time series and all the compared sequences have the same length. Thus the minimum of the distance profile, excluding the trivial match, is the best match with the query sequence. To extend the distance profile to a Matrix Profile, a distance profile for every possible query is calculated. The minima of every distance profile are visualized in the Matrix Profile. For further details see [10].

To extract motifs of a fixed length we need a maximum distance that is not allowed to be exceeded between two sequences of a motif. Every data point in the matrix profile that fulfills this constraint can be seen as the starting point of a motif. The corresponding distance profile gives information about other sequences that are similar to the motif. Again every data point that falls below the maximum distance leads to a similar sequence. The motif with its similar sequences can then be called a pattern. To discover patterns without a fixed query length we have to apply the algorithm multiple times for every possible query length [10].

Because of the naïve structure of the approach, the MASS algorithm was proposed in [10], which calculates the z-normalized Euclidian distance profile by convolution. In comparison to the naïve approach of calculating the Euclidian distance between the query and every other sequence, this technique reduces the time complexity for a distance profile from $O(n * m)$ to $O(n * \log(n))$ with m being the length of the query

and n the length of the whole time series. However, for time warped pattern discovery the Euclidian distance is not suitable. An alternative solution is to replace it with DTW, which theoretically increases the complexity to $O(n * m^2)$. For the calculation of a Matrix Profile we have to calculate approximately n distance profiles, which leads to a complexity of $O(n^2 * m^2)$. As we want to find patterns without input of a query length the Matrix Profile has to be calculated multiple times for different query lengths [10].

In a second implementation we tested Matrix Profile with the Euclidean distance but reduced complexity. It is based on the fact that the distance profiles of the query at starting point i and the one at starting point $i + 1$ have $query_length - 1$ identical Euclidean parts. While calculating the distance profiles in serial, it is possible to keep these identical parts and calculate with just two additional computation steps (per data point in the distance profile) the next distance profile.

4 How to Compare Different Approaches/Algorithms?

As we want to compare different approaches to discover patterns, we need a test standard. One possibility is to use benchmark time series, for example from financial stock markets or from seismology, which are commonly used to test different time series exploration tools. However, generalized results regarding every possible kind of pattern can be obtained only by use of synthetically created time series. Our test data is composited by predefined patterns, which can be labeled automatically. The advantage of this approach is the independency of a certain test case. Furthermore we can create an infinite number of test sequences without the effort of labeling it.

The evaluation is divided in three steps. In the first step the test data is created automatically from patterns that are generated randomly. In the second step, the pattern discovery, the predefined patterns are rediscovered using different algorithms and parameter sets. By using the index information of every predefined and located pattern, in the third step a quality score for the results is calculated.

Create Time Series Test Data

To find generalized evaluation for different use cases, the test patterns should cover every possible kind of shape. That is why we chose the random walk as the basis for every pattern. In our case we use a Gaussian random walk, which changes the distribution of the randomly chosen step size from equal to normal. For the creation of a primal pattern by a Gaussian random walk the following parameters are also chosen randomly once: Value of the first data point; number of steps; maximum step size.

After creating different primal patterns, each of them is multiplied and distorted to form a set of similar members for each of the primal pattern. The number of members within a pattern is randomly chosen. The distortion is done by adding white Gaussian noise to the members, given a fixed signal to noise ratio, and by randomly stretching or compressing the length of the members, given a maximum ratio between the length of the primal pattern and the distorted member.

To form a sequential time series based on the randomly created patterns and its members, the members are concatenated in a random order. Note that every member can only occur once in the time series. To overcome value jumps between the linked

patterns, we insert a smooth crossing. Start and end indices of every member and every pattern are saved.

Pattern Validation

Output of the pattern algorithms are indices of starting and ending points of the located pattern. To evaluate the rediscovered patterns we calculate an overlap comparing these indices I_{loc} with the indices of the predefined pattern I_{pre} by using the Jaccard Index:

$$overlap(pre, loc) = Jaccard(pre, loc) = \frac{|I_{pre} \cap I_{loc}|}{|I_{pre} \cup I_{loc}|} \quad (3)$$

Hence an overlap score of 0 describes a mismatch and a score of 1 describes a perfect match between predefined and discovered pattern. Note that the overlap also decreases from a perfect match if the detected pattern is longer than the predefined. The calculation is performed for every combination of predefined and detected pattern. Afterwards a best match for every predefined pattern can be chosen. To express a one-score quality criterion for the algorithms, we chose the mean overlap value of the best matches.

Furthermore we perform the whole routine multiple times, to get a reliable statement for the quality of the algorithms. This gives us also the possibility to make a statement concerning the variance of the quality. Besides, in every iteration of the routine, the runtime of the different algorithms is recorded.

5 Experimental Evaluation

After explaining the different approaches for the pattern discovery and the methodology of calculating the quality score overlap, we now compare the different approaches. In our evaluation we created 10 random datasets, ran every algorithm with different parametrizations, and calculated the overlap. For each algorithm we figure out the parametrization leading to the best overlap, see Fig. 1. The overlap is plotted for every approach and for every random dataset. Next to the bars the corresponding runtime of the pattern discovery algorithm is indicated.

It is evident that the DTW-based approach outperforms every other tested approach concerning overlap. Discretization-based approaches, in turn, advance greatly concerning the runtime. The algorithm for pattern enumeration provides constantly better results than the Sequitur algorithm. As expected, the Matrix Profile with the use of the Euclidean distance is not able to compete with the other approaches because of its complexity and the inability of time warping. We expect better results when using the Matrix Profile in combination with DTW instead of the Euclidean distance, though substantially increasing runtime. As the DTW-based approach outperforms the Matrix Profile in runtime and overlap, Matrix Profile is not suitable for our use cases. However, as we calculated an entire Matrix Profile for every possible query length, there is still optimization potential for reducing the runtime based on early abandoning. Nonetheless this reduces the runtime, but doesn't improve the overlap.

In general all algorithms output a higher count of patterns than the number of predefined patterns, i.e. 16. While the DTW-based and the adapted Sequitur algorithm locates a reasonable count, the pattern enumeration algorithm and the Matrix Profile produce an extremely high amount of patterns (>20.000, >50.000, resp.). This shows the need of additional postprocessing techniques.

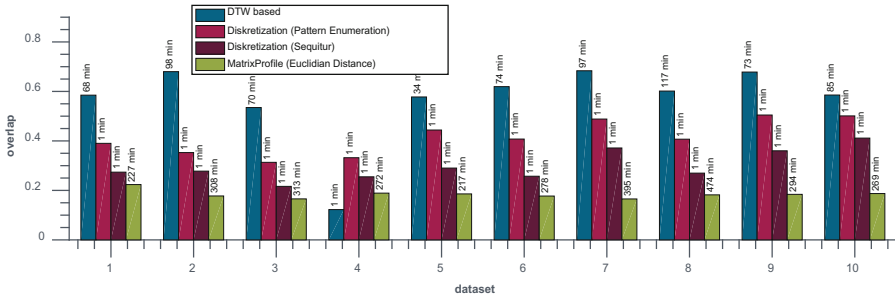


Fig. 1. Overview of different approaches concerning overlap and runtime; length of datasets approximately 50.000 data points; average count of located patterns: DTW ~800; Pattern Enumeration >20.000; adapted Sequitur ~200; Matrix Profile >50.000;

After giving a brief overview of the best case performances, we now have a closer look at the approaches including the process of parametrization. In Fig. 2 (left) the performance of Matrix Profile with variation of the maximum allowable distance within a pattern and with use of Euclidean and z-normalized Euclidean distance is shown. The Euclidean Matrix Profile is calculated with the runtime reduction technique while the z-normalized Euclidean Matrix Profile is calculated by the MASS algorithm. As shown in Fig. 2 the Euclidean Matrix Profile leads to a better overlap for most of the parametrizations, while requiring less than 10% of the runtime.

In Fig. 2 (right) the performance of the discretization-based approaches are evaluated with the variation of the number of discretization steps. Furthermore the benefit of the symbolic reduction is shown. It can be seen that the symbolic reduction has a higher impact on the results of the pattern enumeration algorithm than on the results of the Sequitur algorithm concerning overlap and runtime. However, in both cases the overlap as well as the runtime gets better with the use of the symbolic reduction, due to the time warping. Nevertheless, compared to the DTW-based approach, it is a slight disadvantage because the time warping is not controllable. i.e., after symbolic reduction, a symbol can represent two or considerably more identical consecutive symbols. A problem for both algorithms is the dependency of the discretization method, despite the dependency to only one parameter. Furthermore, we experienced a weakness in terms of multiple symbolic value toggling because of noise in value regions near the discretization borders. This effect could be mitigated by applying a frequency filter as a preprocessing step.

Figure 1 shows the best case results of the approaches concerning the overlap. For the DTW-based approach, for acceptable runtimes we chose a compromise between

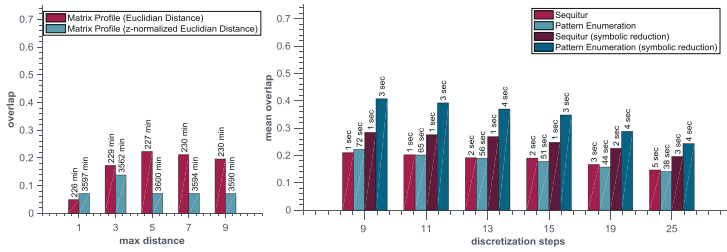


Fig. 2. Left: results (overlap and runtime) of Matrix Profile with variation of the maximum distance; Right: results (mean overlap and mean runtime of all datasets) of the discretization approach (Sequitur, Pattern Enumeration) with and without symbolic reduction and with variation of the discretization steps

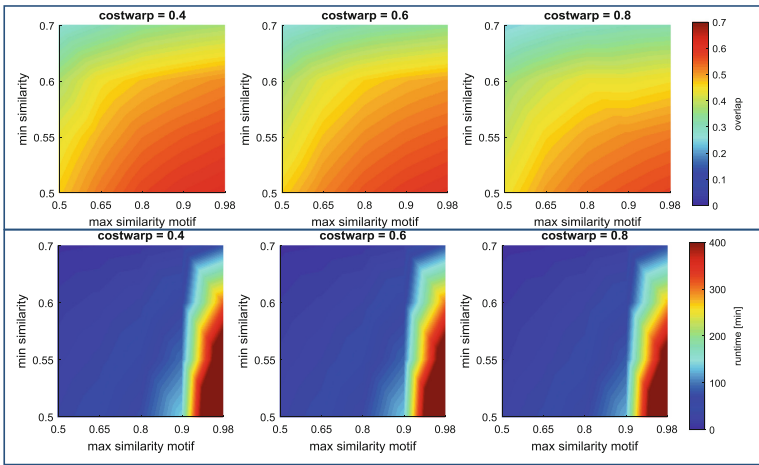
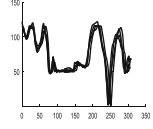
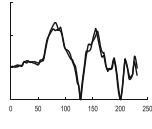
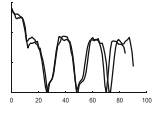
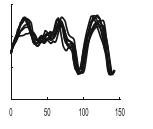


Fig. 3. Results of DTW-based algorithm (10 random datasets), for every *costwarp* parameter value a separate plot; top: mean overlap of random datasets with different parameters; bottom: mean runtime in minutes

acceptable runtime and high overlap. This tradeoff is shown in Fig. 3. With values of *max_similarity_motif* > 0.9, the runtime increases dramatically. This is caused by the clustering of motifs located by CrossMatch. In case of high values of *max_similarity_motif* there are only few motifs clustered together to form pattern. This leads to a high count of patterns, which are then again searched in the whole time series. For further research it has to be evaluated if this last step of the DTW-based approach can be reduced or replaced by a powerful clustering of the candidates. It is also noticeable that the effort of parametrization is high in comparison to the other approaches. At least three parameters have to be chosen carefully to get the expected results. As shown in Fig. 3 the highest overlap values result at values of *min_similarity* = 0.5. We experienced an increase of the need of RAM in cases of values lower than 0.5, which is not practical for our use cases.

For a final validation, for each approach, we applied the best case parameter set to a real life dataset from a typical use case in the automotive industry. Table 3 shows exemplary results for each algorithm, which were produced with the formally determined best case parameters.

Table 3. Exemplary results of a real data set

Approach	DTW	Matrix Profile	Pattern Enum.	Sequitur
Parameters	min_similarity = 0.5 costwarp = 0.4 max_sim_motif=0.8	max_distance = 3	Discr. steps = 9	Discr.steps = 9
Runtime	142 min	31 hours	18 seconds	1 second
Σ pattern	1.041	99.718	46.813	386
Exemplary pattern				

6 Conclusion and Outlook

In this paper we presented different approaches for pattern discovery in time series under the requirement of time warping. Furthermore we applied well known algorithms and adapted them for our purposes. We developed methods for a Jaccard-index-based comparison of these algorithms as well as for a creation of random time series. The comparison helped us to identify advantages and weaknesses of the different approaches. In general discretization-based approaches have a high performance concerning runtime while the DTW-based approaches perform best regarding the quality of the results. The Matrix Profile can be classified as not suitable for our use cases due to the inability of time warping. Because of our goal to mine time series with far more than one million data points our future focus will be on the discretization-based approaches. Therefore it is interesting to evaluate the influence of further preprocessing steps as well as the development of new discretization techniques with robustness regarding noise and the possibility of controllable time warping. Besides the pattern evaluation of unlabeled data and the extension of the pattern discovery algorithms to multidimensional cases are fields of interest.

7 Acknowledgments

We thank Thomas Lehmann (thomas.lehmann@ivi.fraunhofer.de, Fraunhofer IVI, Dresden, Germany) and Dr. Ralf Bartholomäus (ralf.bartholomäus@ivi.fraunhofer.de, Fraunhofer IVI, Dresden, Germany) for the cooperation concerning the adaption and implementation of the DTW-based approach (Sect. 3.1). Furthermore we thank Jan Piewek (jan.piewek@volkswagen.de, Volkswagen AG, Wolfsburg, Germany) for helpful ideas and discussions concerning the discretization-based approaches (Sect. 3.2).

References

1. Höppner, F.: Improving time series similarity measures by integrating preprocessing steps. *Data Min. Knowl. Discov.* **31**, 851–878 (2017)
2. Toyoda, M., Sakurai, Y., Ishikawa, Y.: Pattern discovery in data streams under the time warping distance. *Very Large Data Bases* **22**, 295–318 (2013)
3. Jancovic, P., Köküer, M., Zakeri, M., Russel, M.: Unsupervised discovery of acoustic patterns in bird vocalisations employing DTW and clustering. In: *European Signal Processing Conference* (2013)
4. Lin, J., Keogh, E., Lonardi, S., Chiu, B.: A symbolic representation of time series, with implications for streaming algorithms. In: *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, San Diego, CA, USA (2003)
5. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. *J. Mol. Biol.* **147**, 195–197 (1981)
6. Nevill-Manning, C.G., Witten, I.H.: Identifying hierarchical structure in sequences: a linear-time algorithm. *J. Artif. Intell. Res.* **7**, 67–82 (1997)
7. Li, Y., Lin, J., Oates, T.: Visualizing variable-length time series motifs. In: *SIAM International Conference on Data Mining*, Philadelphia, USA (2012)
8. Coquery, E., Jabbour, S., Sais, L.: A constraint programming approach for enumerating motifs in a sequence. In: *International Workshop on Declarative Pattern Mining*, Vancouver, Canada (2011)
9. Rajeb, A., Loukil, Z., Hamadou, A.B.: On the enumeration of frequent patterns in sequences. In: *The International Conference on Artificial Intelligence and Pattern Recognition*, Kuala Lumpur, Malaysia (2014)
10. Yeh, C.-C., et al.: Matrix profile I: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets. In: *16th International Conference on Data Mining*, Barcelona, Spain (2016)
11. Salvador, S., Chan, P.: FastDTW: toward accurate dynamic time warping in linear time and space. In: *KDD Workshop on Mining Temporal and Sequential Data* (2004)



Machine Learning Methods Based Preprocessing to Improve Categorical Data Classification

Zoila Ruiz-Chavez^{1(✉)}, Jaime Salvador-Meneses¹, and Jose Garcia-Rodriguez²

¹ Universidad Central del Ecuador, Ciudadela Universitaria, Quito, Ecuador
{zruiz, jsalvador}@uce.edu.ec

² Universidad de Alicante, Ap. 99., 03080 Alicante, Spain
jgarcia@dtic.ua.es

Abstract. The following study is aimed at dealing with large volumes of data whose main characteristic is to contain a high number of variables, most of which are categorical in nature. In the knowledge extraction process, Knowledge Discovery in Databases (KDD), it is very common to deal with a stage of data pre-processing and dimensionality reduction. A key part of extracting information is having high quality data. This paper proposes the use of the Pairwise and Listwise methods as part of the dimensionality reduction process, when there is a high level of missing values present in one or more variables. As part of the pre-processing, we generate n-clusters using Kohonen Self-Organizing Maps (SOM) algorithm with H2O on R. A comparison of the performance and accuracy of classification algorithms is made with the complete subdata set and the algorithms are applied to each cluster. As a case study, we analyzed the characteristics that influence the level of schooling of women of childbearing age.

Keywords: Machine learning · Classification · Clustering
Missing data

1 Introduction

In the handling of large volumes of data, we usually find a high dimensionality, but nevertheless in the task of predicting a variable, whether numerical or categorical, not all variables have the same importance within the process. Most features are often irrelevant or redundant [10]. This is why it is necessary to use a method of Feature Selection prior to the process of predicting data. In this way we considerably reduce the dimensionality of the data and improve the efficiency of the proposed models by reducing the complexity of the analysis.

However, the data are not always complete, they usually contain out of range, erroneous or missing data. There are three types of missing data: Completely Random Absence (MCAR), Random Absence (MAR) and Non-Random Absence (NMAR) [5]. This type of problem is generally dealt with at the data

pre-processing stage, either by elimination (Pairwise or Listwise) or by imputation of data to avoid a considerable decrease in the sample.

A traditional source of information is the population and housing censuses. In Latin America, censuses are conducted periodically over a ten-year period. These have been of vital importance because they are a source of information that allows decisions to be made based on [2] projections. Censuses are widely used to analyse the profile of a country's population [4].

The proposed model for the classification of categorical variables is based on the generation of clusters using SOM, each clusters may or may not contain all the categories of the variable to predict, however the elements in each cluster possess similar characteristics that were detected by the SOM algorithm. Each cluster is considered as a subset of data that will be processed by the different classification algorithms.

This document is organized as follows: Sect. 2 summarizes the work done in the area, Sect. 3 performs an analysis of the data and describes the pre-processing to obtain a complete dataset, Sect. 4 compares different classification algorithms based on the sets generated in the previous section and, finally, Sect. 5 presents some conclusions and future work.

2 Related Work

It is very common to find studies oriented to the improvement in the selection of relevant characteristics (FS), or in the performance and precision in the classification process. Dealing with large data sets directly involves dealing with reduction of complexity inherent in the size and format of the data [7].

This problem is usually treated from two approaches (ALL or PART). On the one hand, using all the data in the FS process can introduce selection bias, but dealing with only sample of the data can underestimate the relevant characteristics [1].

While there are several classification techniques for dealing with large volumes of data, with certain advantages and disadvantages inherent in the algorithm or related to the size or types of data they work with, it is difficult to choose the most appropriate one. In the work of Gorade et al. [3] a comparison of five techniques of supervised classification is made based on Trees, Neural Networks, Neighborhood and vectors, presents characteristics or qualities that classifiers generally possess. But when dealing with classification algorithms in data with high dimensionality, we also deal indirectly with characteristic selection algorithms. In Shirzad et al. [9], he examines the importance of feature selection within the process of training a classification model. Improving the accuracy of machine learning problems is one of the main objectives, the work of Pandey et al. [6], refers to the complexity involved in the dimensionality of data in the Classification process.

3 Data Analysis

In the field of data analysis the main objective is undoubtedly the extraction of knowledge KDD (Knowledge Discovery in Databases). With the growth of available data and taking into account the diverse sources that produce data at speeds previously unthinkable (social networks), it became increasingly necessary to find alternatives that fit the characteristics present in the data (4V's).

Algorithms that work properly for the management of categorical data are limited. For this reason, in this paper we take as a case study the data from censuses that are mostly categorical data, with a total of 101 variables, in addition to a wide range (age[0–120]), that is, a variable can have 120 categories.

For this study, Ecuadorian demographic information corresponding to the 2010 population and housing census is used. The information is grouped under: (1) **Living Place:** Information associated with the dwelling (28 features), (2) **Home:** Household information (26 features), (3) **Person:** Information associated with persons (95 features).

In addition to the above information, there is also information related to the political and administrative division of the country (geographic information called DPA). For this particular case, one province was selected for each region, with a population (according to the 2010 census) of 1048575 inhabitants.

Survey data usually present a very common problem, the presence of missing data or data out of range. In order to use the data and obtain suitable results, it is necessary to go through a pre-processing stage taking into account the variance and the correlation between them.

3.1 Missing Values Processing

In data from surveys or censuses it is very common to find a high percentage of missing or out of range values, either because of data collection failures or because of non-response on the part of respondents to certain questions.

There are two ways to deal with this problem, eliminating or replacing missing or out of range values, which can be statistical methods or based on Artificial Intelligence algorithms. They can be classified into two groups:

- Disposal Methods
- Imputation Methods

For this study, we based ourselves on the elimination methods in order to obtain a complete dataset that allows us to verify the accuracy of the algorithms, and although it reduces the sample size, it preserves the correlation, variance and distribution of the data. Out of range data are treated as missing values.

The algorithms were executed in WEKA, RapidMiner and H2O over R. The equipment used for the experimentation has an i5 processor, 4 GB in RAM and 256 GB SSD.

3.2 Pre-processing

One of the objectives that we should not lose sight of at this stage is the elimination of bias caused by errors in the data. We can apply different approaches, depending on the objectives set out in the study [8]. In general, pre-processing has general tasks common to any type of data and study, such as: Cleanliness, Integration, Transformation and Reduction [11].

In order for the data to be consistent, it has been detected if there are lost, inconsistent or missing values to be able to correct, impute or if necessary eliminate them. Linear Regression was applied to determine the most representative variables, by means of backwards elimination (P.Values > 0.05), in a recursive manner.

With this previous selection, the Listwise and Pairwise methods were executed to obtain a complete set. A dataset was obtained with 18 variables and 181257 observations. Table 1 contains the selected variables that contain missing values:

Table 1. Variables with missing data

Variable	Description	Total NA
p08	Have permanent disability for more than one year	5496
p23	Highest level of instruction attended or attended	1795
p24	Highest grade, course or year that passed	7232
p36	Children and daughters born alive	1292
p37	Total live children currently	34819
p38	At what age did your first son or daughter have	36333
graesc	Degrees of schooling	7232

As a result of the pre-processing stage we get a complete dataset with 181257 observations and 18 attributes.

3.3 Machine Learning Techniques

In a previous paper, some existing classification techniques in the field of automatic learning were described. For this study we used both supervised and unsupervised classification techniques. In the proposed model we use techniques for classification, selection of characteristics and clustering. Supervised classification algorithms require a complete data set for pre-prediction training. The complete data set obtained in pre-processing allows us to evaluate the accuracy of each algorithm. We use the following methods: Random Forest (RF), Multilayer perceptrons (MLP), Naive Bayes (NB) and K-Nearest Neighbor (kNN).

These methods are applied to compare their performance against randomly constructed datasets and clusters generated by SOM.

3.4 SOM Clustering

With a complete set of data we can train the model to predict, as an example we use the variable *graesc*. Due to the size of the data set there are difficulties in terms of calculation capacity, time or processing power of the algorithms. As an alternative to the proposed model, segmentation of the data set is proposed. We generate n-cluster using SOM, then different classification algorithms are run for each of the clusters and thus compare their performance.

Below we present the process used for cluster generation through self-organizing maps (SOM). SOM only operates with numeric variables, so dummy variables are created for each categorical variable.

The execution of SOM was parameterized with the following values: Rectangular topology, 5×5 Grid size, 100 iterations and a Learning rate of 0.05 and 0.01

The algorithm is executed with these parameters, then the following graphics generated by the algorithm are analyzed:

- *Trainin progress*, which shows us the decrease in the average distance of each cell from its neighbours with respect to the number of iterations.
- *Heat map*, which allows you to associate observations with cells in the grid. Each observation is assigned to the cell with the nearest representative vector.
- *Map of distances*, the cell color represents the total distance to neighboring cells.

Each observation is assigned to the representative vector as a cluster, each cell corresponds to a cluster, and each observation is assigned to the nearest vector within the grid. With this process we obtain 25 clusters that correspond to the size of the grid.

3.5 Clustering Based on the Codebook

To generate the codebook (mapping of the original data to a smaller set of values, close to the original values), the output of the SOM model was used. For this purpose, a cluster was created on the set of representative vectors, which in the case of the example corresponds to 25. After generating n-cluster with the dataset of nodes, the assignment of clusters to the original dataset is extended.

The first step is to determine the WCSS (Within Cluster Sum of Squares). Figure 1 shows the distribution of the error according to the number of clusters.

Through the codebooks a cluster size equal to 7 was taken, in this value you can see a break point in the error distribution.

In the Table 2 we can visualize the distribution of observations by each cluster:

This generates the following distribution that is feasible to visualize for each cluster, since it is information that is represented in two dimensions (Fig. 2):

The clusters generated have similar characteristics, making it a more suitable type of segmentation and easier for classification algorithms to analyze.

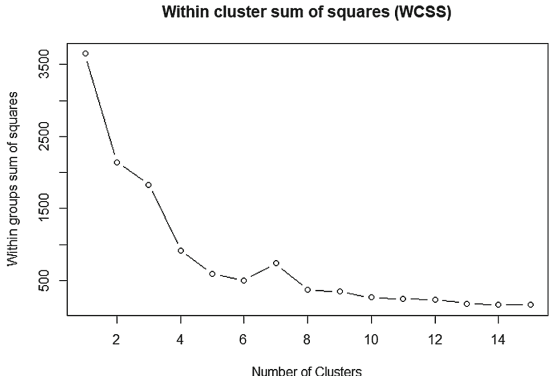


Fig. 1. WCSS

Table 2. Distribution by cluster

Cluster	Total elements	Cluster	Total elements
1	32824	5	19288
2	12966	6	46691
3	12969	7	34021
4	22498		

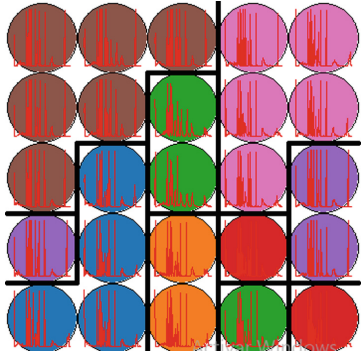


Fig. 2. Clusters

4 Application of the Classification Model

With the complete dataset with 18 variables and the clusters generated from the original dataset, experimentation is carried out to compare the results and see if it is more convenient, both in terms of performance, computational costs and precision, to deal with a random sample to predict the *graesc* variable or by segmentation of the data through the generation of clusters.

Three subsets of data were generated randomly, in each set the 25 categories of the variable to be predicted were present. In the execution of the algorithms, 80% for the training and 20% for the test were considered. The results obtained with each algorithm allow us to verify the accuracy and process time according to the characteristics of each subset.

Table 3. Results of the algorithms

Data			RF		MLP		NB		KNN	
Data	NR	NC	P	T	P	T	P	T	P	T
data1	25433	25	91.92	3:51	93.38	9:01	32.24	0:01	66.46	0:16
data2	12966	25	97.06	0:57	93.38	3:32	44.34	0:01	59.49	0:02
data3	18123	25	95.68	0:52	92.99	3:52	66.94	0:01	51.24	0:03
cluster1	32824	25	99.79	3:53	93.38	9:01	32.24	0:01	66.46	0:16
cluster2	12966	25	99.43	0:59	93.38	3:32	44.34	0:01	59.49	0:02
cluster3	12969	25	99.36	0:52	92.99	3:52	66.94	0:01	51.24	0:03
cluster4	22498	25	99.73	1:59	92.09	6:27	41.30	0:01	62.99	0:06
cluster5	19288	2	100	0:37	100	0:51	100	0:01	99.98	0:04
cluster6	46691	3	99.97	7:12	99.97	2:16	99.99	0:01	99.17	0:31
cluster7	43021	25	99.77	4:12	92.77	9:05	47.32	0:01	67.23	0:16

In the Table 3 we find a comparison of the accuracy of the classification techniques used and the execution time, with the following columns:

- DATA: the dataset with which the experiment is performed, NR: the number of available records, NC: the number of categories present for the variable *graesc*, P: Algorithm precision, T: algorithm run time (in minutes)

With the results obtained we can make a comparison of the different algorithms with different sample sizes and number of categories present in a variable, which are summarized in the Table 4.

Table 4. Comparison features

Algorithm	Precision	Time	Data size	Parameters
Random forest	✓✓✓	✓✗	✓✓✓	I = 100, S = 1
MultiLayer perceptron	✓✓	✓✗	✓✓✗	L = 0.3, M = 0.2, N = 500, E = 20
Naive Bayes	✓✗	✓✓✓	✓✗	Laplace correction = true
KNN	✓✗	✓✓	✓✓✗	K = 3

5 Conclusions and Future Work

In terms of accuracy, the best results were obtained with the RF and MLP algorithms, without significant differences between them. However, the difference in processing time is significant. It was noted that the algorithms trained with the subsets generated by SOM-Clustering are more efficient than if only randomly generated subsets are used. Without a doubt, use clustering techniques to obtain subsets of data benefits the processing with Machine Learning techniques (subsets are more homogeneous). For each cluster generated we will obtain a possible value (n -values total), generating a final value by means of multiple imputation. As future work we will apply this model with different imputation methods to test its performance.

References

1. Aldehim, G., Wang, W.: Determining appropriate approaches for using data in feature selection. *Int. J. Mach. Learn. Cybern.* **8**(3), 915–928 (2017)
2. Chackiel, J.: Métodos de estimaciones demográficas de pueblos indígenas a partir de censos de población: La Fecundidad y la Mortalidad. In: *Pueblos indígenas y afrodescendientes de América Latina y el Caribe: relevancia y pertinencia de la información sociodemográfica para políticas y programas*, p. 30 (2005)
3. Gorade, M.S.M., Deo, A., Purohit, P.: A study of some data mining classification techniques. *IRJET* **4**, 3112–3115 (2017)
4. Acuña, M.: Redatam Informa. *Redatam Inf.* **19**(19), 13–17 (2013)
5. Mojirsheibani, M., Shaw, C.: Classification with incomplete functional covariates. *Stat. Prob. Lett.* **139**, 40–46 (2018)
6. Pandey, G., Ren, Z., Wang, S., Veijalainen, J., de Rijke, M.: Linear feature extraction for ranking. *Inf. Retrieval J.* **1**, 1–26 (2018)
7. Ramírez-Gallego, S., Krawczyk, B., García, S., Woźniak, M., Herrera, F.: A survey on data preprocessing for data stream mining: current status and future directions. *Neurocomputing* **239**, 39–57 (2017)
8. Roy, A., Cruz, R.M., Sabourin, R., Cavalcanti, G.D.: A study on combining dynamic selection and data preprocessing for imbalance learning. *Neurocomputing* **286**, 179–192 (2018)
9. Shirzad, M.B., Keyvanpour, M.R.: A systematic study of feature selection methods for learning to rank algorithms. *Int. J. Inf. Retr. Res. (IJIRR)* **8**(3), 46–67 (2018)
10. Spolaôr, N., Cherman, E.A., Monard, M.C., Lee, H.D.: A comparison of multi-label feature selection methods using the problem transformation approach. *Electron. Notes Theor. Comput. Sci.* **292**, 135–151 (2013)
11. Zulkepli, F.S., Ibrahim, R., Saeed, F.: Data preprocessing techniques for research performance analysis. In: Patnaik, S., Popentiu-Vladicescu, F. (eds.) *Recent Developments in Intelligent Computing, Communication and Devices. AISC*, vol. 555, pp. 157–162. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-3779-5_20



Crossover Operator Using Knowledge Transfer for the Firefighter Problem

Krzysztof Michalak^(✉)

Department of Information Technologies, Institute of Business Informatics,
Wrocław University of Economics, Wrocław, Poland
krzysztof.michalak@ue.wroc.pl

Abstract. This paper concerns the Firefighter Problem (FFP) which is a graph-based problem in which solutions can be represented as permutations. A new crossover operator is proposed that uses a machine learning model to decide how to combine two parent solutions of the FFP into an offspring. The operator works on two parent permutations and the machine learning model provides information which parent to select the next permutation element from, when constructing a new solution. Training data is collected during a training run in which transpositions are applied to solutions found by an evolutionary algorithm for a small problem instance. The machine learning model is trained to classify pairs of graph vertices into two classes corresponding to which vertex should be placed earlier in the permutation.

In the experiments the machine learning model was trained on a set of FFP instances with 1000 vertices. Subsequently, the proposed operator was used for solving FFP instances with up to 10000 vertices. The experiments, in which the proposed operator was compared against a set of other crossover operators, shown that the proposed operator is able to effectively use knowledge gathered when solving smaller instances for solving larger instances of the same problem.

Keywords: Knowledge-based optimization · Graph problems
REDS graphs

1 Introduction

Transfer learning is an approach used in machine learning in which a model is trained on one problem instance and is subsequently reused for solving other instances of the same or similar problems [1]. This approach is popular, among others, in deep learning tasks, because of huge computational resources required to train deep learning models. For example, training deep neural models for computer vision and natural language processing tasks can take days or weeks on a modern hardware. However, it turns out that in many cases the networks learn very similar features in the first layer. Thus, models pre-trained on representative datasets can be used to speed up learning on other datasets [2]. Obviously, the

general idea of transferring knowledge from one problem instance to another is not limited to deep neural networks. In this paper a machine learning model is used to extract information from a set of optimization problem instances and to reuse this information when solving other, larger instances.

In this paper the concept of knowledge transfer (using information obtained from one problem instance for solving other problem instances) is used for solving the Firefighter Problem (FFP) [3]. The FFP is a combinatorial optimization problem which concerns the containment of fire spreading on a graph. This problem is an abstraction of several real-life problems such as prevention of epidemics, fire and floods containment and catastrophic failures prevention (such as cascading failures in power grids or waves of bankruptcies). In the FFP spreading of fire is simulated in discrete time steps on an undirected graph $G = \langle V, E \rangle$ with N_v nodes, in which vertices can be in one of the states ‘B’ - burning, ‘D’ - defended, ‘U’ - untouched. The edges of G represent neighbourhood relations between vertices and they are not subject to burning nor they can be defended. At $t = 0$ vertices of the graph G are in an initial state S_0 in which, typically, some of the vertices are in the ‘B’ state and the remaining ones are in the ‘U’ state. In each subsequent time step, N_f untouched nodes become defended by firefighters (change their state to ‘D’). Defended nodes remain in the ‘D’ state until the end of the simulation and fire cannot spread to nor through them.

The goal of optimization in the FFP is to determine which nodes to defend at each time step in order to contain the fire, with the constraint that only a limited number of nodes N_f can become defended in each time step. Solutions of the FFP can be represented as permutations of N_v elements which define in what order to protect vertices of the graph G . For a given solution $\pi \in \Pi_{N_v}$, at each time step, N_f first elements are taken from π such that the corresponding vertices are in the ‘U’ state and these vertices become defended. In order to evaluate a solution π the spreading of fire is simulated starting from the initial state S_0 and using the solution π to determine which vertices to protect at each time step. In the classical FFP formulation [3] the value of the objective function is the number of non-burning vertices at the end of the simulation, when fire stops spreading. In a generalized single-objective variant of the problem [4] costs are assigned to vertices of the graph G and the value of the objective function is calculated by summing the costs of the non-burning vertices at the end of the simulation. Likewise, in the multiobjective version of the problem m different costs are assigned to each vertex (representing, for example, the amount of various resources lost if the vertex burns). The values of m objective functions are calculated by summing the values of these m costs assigned to the non-burning vertices at the end of the simulation.

2 Proposed Approach

In this paper a machine learning model is used in a crossover operator to decide from which parent to select the next element of a new permutation.

2.1 Training the Machine Learning Model

Formally, the model used in this paper is a classifier $\Psi : V \times V \rightarrow \{0, 1\}$ which, given a pair of vertices, returns a value indicating which vertex should be placed closer to the beginning of the permutation. Such classifier requires training on a data set containing pairs of vertices and, for each pair, a 0 or 1 value indicating which vertex to place closer to the beginning of the permutation. To allow applying a trained classifier to various graphs, the vertices are represented using several graph-based attributes presented in Table 1. Thus, the classifier can be defined as $\Psi : \mathbb{R}^{2d} \rightarrow \{0, 1\}$, where d is the number of attributes describing one vertex (therefore, the classifiers receives $2d$ attributes for a pair of vertices).

Table 1. Attributes of graph vertices used in the proposed method

Name	Description
<i>deg_centr</i>	Degree centrality. The number of edges that are adjacent to the vertex divided by $N_v - 1$
<i>clos_centr</i>	Closeness centrality. An inverse of the mean length of shortest paths from the vertex v to other vertices in the graph: $(N_v - 1) \frac{1}{\sum_{w \neq v} d_{min}(v, w)}$, where $d_{min}(v, w)$ is the length of the shortest path from v to w
<i>betw_centr</i>	Betweenness centrality. A measure which indicates how often the shortest paths connecting other vertices go through the vertex v . Denote $\sigma(s, t)$ - the total number of shortest paths connecting vertices s and t , $\sigma(s, t v)$ - the number of shortest paths connecting vertices s and t going through v . Then, betweenness centrality is calculated as: $\frac{1}{(N_v - 1)(N_v - 2)} \sum_{s \neq v \neq t} \frac{\sigma(s, t v)}{\sigma(s, t)}$
<i>burned_at</i>	Time step number in a simulation at which the vertex caught on fire

The training data set is constructed by running an evolutionary algorithm solving training FFP instances. During the optimization run, an improvement operator is applied to solutions found by the evolutionary algorithm, which randomly selects two elements in the permutation to transpose. In order to limit the training data set size, in the experiments presented in this paper only one transposition of randomly selected vertices was performed for each solution found by the evolutionary algorithm. If the transposition of vertices v_i and v_j resulted in an improvement of the objective function, a sample consisting of attributes of v_i , attributes of v_j and a class value ‘1’ was added to the training data set. If no improvement was obtained, a sample consisting of attributes of v_i , attributes of v_j and a class value ‘0’ was added to the training data set.

The proposed approach can use any classifier capable of classifying objects described by real-valued attributes. In this paper a multilayer perceptron (MLP) was used with the number of input neurons N_{in} equal to the number of attributes describing two vertices $2d$ and the number of outputs $N_{out} = 2$. The number of hidden neurons was adjusted during the network selection phase of the experiments. Neurons in the hidden layer used the hyperbolic tangent activation function [5] and neurons in the output layer used the softmax activation function [6]. The latter is typically used in networks trained for classification tasks in which the number of output neurons equals the number of classes. Note, that for the network architecture in which the number of output neurons equals the number of classes, the classes in the training sample are not encoded as numbers, but as binary vectors with the element at the index corresponding to the actual class set to ‘1’. Therefore, in the discussed classification problem the classes are encoded as [1, 0] (the ‘0’ class) and [0, 1] (the ‘1’ class). Correspondingly, the output values generated by the network are not class numbers, but weights assigned by the network to the classes. To interpret the output values as classes one has to compare these values and select the class number equal to the index of the output returning the highest value.

The entire process of training the neural network is shown in Fig. 1. This figure shows how the information about improving (or non-improving) transpositions is combined with attributes of the transposed vertices to generate samples in the training data set. Next, this data set is used to train the neural network. Note, that the key idea in the proposed approach is that the model is trained once (and preferably on a small problem instance) and is subsequently used many times to solve other (preferably larger) problem instances.

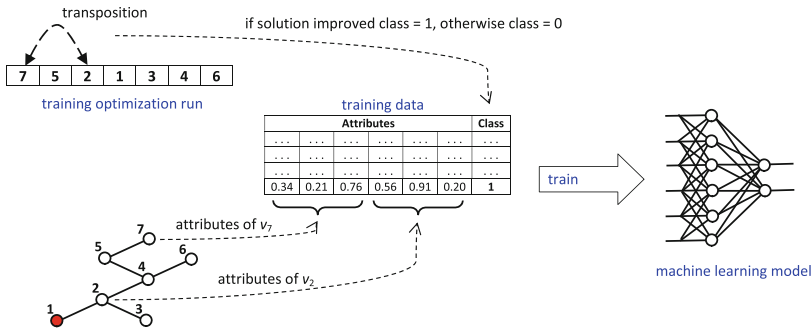


Fig. 1. Training of the machine learning model

2.2 The Crossover Operator

To construct the offspring, the crossover operator presented in this paper compares pairs of elements that appear in parent permutations. The machine learning model is used to decide from which of the parents to take the next element.

Let v_i and v_j be the next two vertices selected from each of the parents respectively. A vector containing attributes of vertex v_i and vertex v_j is formed and is classified using the trained machine learning model. If the class returned by the machine learning model is ‘0’ then vertex v_i is added to the offspring and if the returned class is ‘1’ then vertex v_j is added. The working of the proposed crossover operator is described in Algorithm 1. The function **Concat** used in this algorithm merges two vectors together, and is used to construct one attribute vector *attr* from the attributes of two vertices and to add a new vertex to the constructed offspring S . The **M.Classify** method classifies a given vector of attributes using the machine learning model M . As discussed in Sect. 2.1 in the case of a neural network two output values o_1 and o_2 are produced. If value o_1 is larger it indicates class ‘0’ and if value o_2 is larger it indicates class ‘1’. Therefore, if $o_1 > o_2$ then vertex v_i is added to the offspring and vertex v_j is added otherwise. Figure 2 shows how the neural network is used for selecting the next element from one of the parents.

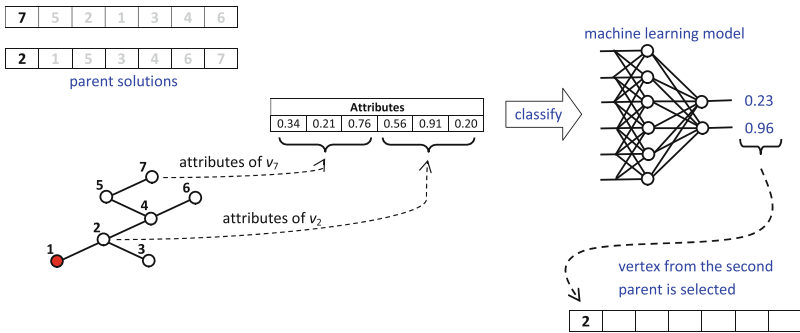


Fig. 2. The working of the proposed crossover operator

3 Experiments and Results

Experiments presented in this paper were aimed at verifying if the proposed crossover operator improves the results obtained by the evolutionary algorithm solving the Firefighter Problem. In particular, the tests were performed so as to determine if the knowledge gathered on a set of FFP instances with $N_v = 1000$ can be transferred to larger problem instances.

3.1 Test Instances

As discussed in the introduction, an instance of the Firefighter Problem consists of a graph G , the initial state S_0 , and the number of vertices that become defended at each time step N_f . In this paper REDS graphs were used, which

Algorithm 1. The working of the proposed crossover operator.

```

IN:    $P_1, P_2$  - parent solutions
       $M$  - machine learning model
OUT:   $S$  - a new offspring

 $S := \emptyset$ 
 $idx_1 := 1$ 
 $idx_2 := 1$ 

for  $i := 1, \dots, N_v$  do
  // Skip in  $P_1$  and  $P_2$  the elements already present in  $S$ 
  while  $P_1[idx_1] \in S$  do
     $idx_1 := idx_1 + 1$ 
  end while

  while  $P_2[idx_2] \in S$  do
     $idx_2 := idx_2 + 1$ 
  end while

   $v_1 := P_1[idx_1]$ 
   $v_2 := P_2[idx_2]$ 

  // Concatenate attributes of  $v_1$  and  $v_2$ 
   $attr := \text{Concat}(v_1.attr, v_2.attr)$ 

  // Classify the attributes vector
   $cl := M.\text{Classify}(attr)$ 

  // Decide, based on the class, which parent to use
  if  $cl = 0$  then
    // Class '0' returned, add the element from  $P_1$ 
     $S := \text{Concat}(S, [v_1])$ 
     $idx_1 := idx_1 + 1$ 
  else
    // Class '1' returned, add the element from  $P_2$ 
     $S := \text{Concat}(S, [v_2])$ 
     $idx_2 := idx_2 + 1$ 
  end if
end for

```

were proposed in order to obtain spatial edge distribution with denser cliques separated by relatively sparse areas resembling the distribution in a real-life social network [7]. The construction of REDS graphs is parameterized using three parameters: R - the maximum distance between connected vertices, E - the social energy, and S - the synergy parameter. Positions of vertices in a REDS graph are uniformly drawn from $[0, 1] \times [0, 1]$ and each vertex has initially an energy budget E . Pairs of vertices located not more than R from each other are

randomly selected and a construction of an edge is attempted for each of these randomly drawn pairs. An edge incurs a cost equal to its length D_{ij} on both vertices it connects and in order for an edge to be constructed the limit of the energy that each vertex can spend E cannot be exceeded. However, if vertices v_i and v_j have k_{ij} neighbours in common, the cost of the edge connecting these vertices is discounted by the factor of $\frac{1}{1+Sk_{ij}}$, where S is the synergy parameter. During edge generation a newly created edge contributes to the cost discount before the limit E is checked. Therefore, an edge can be formed between vertices that have less than D_{ij} energy left, because of the energy cost discount. Examples of REDS graphs obtained for $N_v = 1000$ and $N_v = 5000$ and R , E and S set as shown in Table 2, are presented in Fig. 3. In the figure nodes of the graphs are coloured according to the degree (white - low degree, red - high degree) and edges are coloured according to the value of the cost discount denominator $1 + Sk_{ij}$ (red - low values, blue - high values).

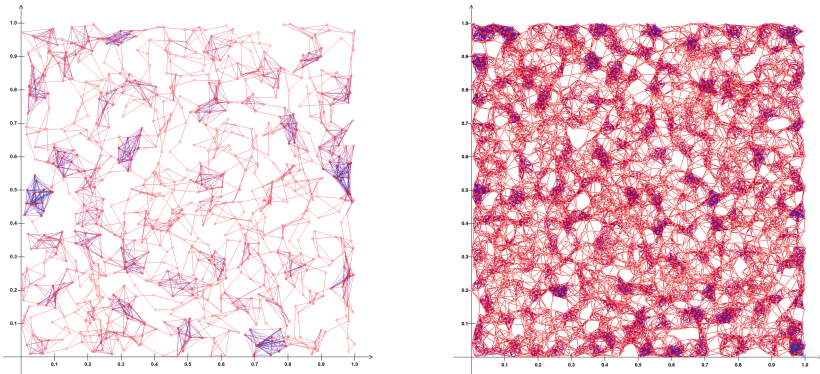


Fig. 3. Examples of REDS graphs obtained for $N_v = 1000$ and $N_v = 5000$ and R , E and S set as shown in Table 2 (colour online).

Depending on where the fire breaks out and how many vertices can be defended at each time step in a given FFP instance, it can be very easy or impossible to save most of the graph. Obviously, if fire breaks out at a vertex with a very low degree it is easy to cut it off from the rest of the graph. Also, if many vertices can be defended at each time step it is easy to stop the fire. In order to make the test instances reasonably difficult, one vertex was randomly selected in each graph as the fire starting point from vertices with degree no less than $k_{min} = \lceil \bar{k} \rceil$, where \bar{k} is the average vertex degree in graphs generated for a given number of vertices N_v . On the other hand, the parameter N_f controlling how many vertices can be defended at each time step was set to $N_f = k_{min} - 1$. These settings guarantee that fire will not be contained right after it breaks out, because the vertex where fire starts cannot be totally surrounded by defended vertices. Table 2 presents the parameters used for generating the test instances.

Table 2. Parameters of problem instances used in the experiments.

Test instance parameters						
N_v	R	E	S	\bar{k}	k_{min}	N_f
1000	0.1000	0.15	0.5000	7.35	8	7
1250	0.0890	0.15	0.4470	7.97	8	7
1500	0.0820	0.15	0.4080	8.11	9	8
1750	0.0760	0.15	0.3780	8.57	9	8
2000	0.0700	0.15	0.3500	9.16	10	9
2250	0.0670	0.15	0.3330	9.43	10	9
2500	0.0630	0.15	0.3160	10.09	11	10
5000	0.0447	0.15	0.2236	13.61	14	13
10000	0.0316	0.15	0.1581	18.35	19	18

3.2 Optimization Algorithm

In the experiments the proposed crossover operator was tested using a classical genetic algorithm with an auto-adaptation mechanism [8] used for crossover operator selection. The crossover operators used in this paper were: Cycle Crossover (CX), Linear Order Crossover (LOX), Merging Crossover (MOX), Non-Wrapping Order Crossover (NWOX), Order Based Crossover (OBX), Order Crossover (OX), Position Based Crossover (PBX), Partially Mapped Crossover (PMX), Precedence Preservative Crossover (PPX) and Uniform Partially Mapped Crossover (UPMX). These operators are commonly used in the literature for permutation-based problems, and therefore were selected as representative of the state of the art and used for comparison with the proposed operator. The experiments compared the solutions obtained using the above-mentioned set of crossover operators with and without the MLP-based operator proposed in this paper. The mutation operators were: displacement mutation, insertion mutation, inversion mutation, scramble mutation and transpose mutation. In the experiments the objective function value for a solution π was the number of non-burning vertices obtained in a simulation when vertices were defended according to the solution π . Because in the FFP evaluation of solutions consumes a large part of computational resources used for solving the problem, the number of solution evaluations was used as the stopping criterion for the algorithm. The limit of $N_{SE} = 200\ 000$ solution evaluations was used. Other parameters of the algorithm were tuned using the grid search approach. The following values were tested for the parameters: population size $N_{pop} = \{50, 100, 200, 500\}$, crossover probability $P_{cross} = \{0.2, 0.4, 0.6, 0.8, 1.0\}$ and mutation probability $P_{mut} = \{0.02, 0.04, 0.06, 0.08, 0.10\}$. To select the best settings, 30 runs of the optimization algorithm were performed on an FFP instance with $N_v = 1000$ for each triple of values of these parameters. The best set of parameters, for which the highest average of the best solution evaluations in 30 runs was achieved, was $N_{pop} = 200$, $P_{cross} = 1.0$ and $P_{mut} = 0.1$.

3.3 Neural Network Training

The training data set was prepared as discussed in Sect. 2.1 by running the evolutionary algorithm 30 times on 30 different training problem instances with $N_v = 1000$ vertices. These instances were used solely for training and were *not* reused in the tests described in Sect. 3.4 for which a separate set of 30 instances with $N_v = 1000$ was generated. A single training data set was formed from all the training samples generated in these 30 runs. The training data shown a high imbalance of the classes, with about 96% of the training samples belonging to the ‘0’ class and only about 4% of the training samples belonging to the ‘1’ class. Because of that, three different approaches to handling class imbalance were tested: none (no balancing), downsample (taking fewer examples from the majority class) and oversample (taking multiple copies of examples from the minority class). Neural networks were trained using the Scaled Conjugate Gradient (SCG) algorithm [9] with the number of iterations $N_{SCG} = 100$. Networks that attained the best values of four classification quality measures were selected: accuracy (ACC), precision (P), true positive rate (TP, a.k.a. recall) and true negative rate (TN, a.k.a. specificity).

For each of the quality measures $Q \in \{ACC, P, TP, TN\}$ the procedure for selecting the best network was as follows.

- 90 neural networks were trained with the number of hidden neurons $N_{hid} = 1, \dots, 30$ and the class imbalance setting $B \in \{\text{none, downsample, oversample}\}$.
- For each pair of values of N_{hid} and B a 10-fold cross-validation procedure [10] was performed. In such procedure the training data set is randomly split into 10 equal parts and the machine learning model is trained on 90% of data and tested on the remaining 10% (taking a different 10% part in each fold). The cross-validation procedure produced 10 neural networks and 10 values of the quality measure Q .
- The values of N_{hid} and B for which the highest average value of the quality measure \bar{Q} was obtained were taken as the best number of hidden neurons N_{hid}^* and the best class balancing approach B^* .
- From 10 cross-validation folds performed for the values N_{hid}^* and B^* the network corresponding to the highest value of the quality measure Q was selected.

Neural networks trained using different classification quality measures Q are presented in Table 3.

3.4 Testing of the Crossover Operator

After neural networks are trained, they can be used in the crossover operator according to Algorithm 1. In the experiments the performance of the proposed crossover operator using the four networks selected using different quality measures was compared to a situation when the MLP-based crossover was not used

Table 3. Parameters of the trained neural networks

Network name	Quality measure Q	Class balancing	N_{hid}^*	Average quality measure \bar{Q}	Quality measure for the best network
Net_{ACC}	Accuracy	None	9	0.81254	0.96085
Net_P	Precision	Oversample	23	0.58264	1.00000
Net_{TN}	True negative rate	None	9	0.84023	1.00000
Net_{TP}	True positive rate	None	5	0.89296	0.99983

at all (the ‘None’ setting). Table 4 summarizes the results. Presented values are means of the best solution evaluation obtained in 30 runs (higher values are better). For the MLP-based crossovers values better than for the ‘None’ setting are underlined. As can be seen in the table, neural networks selected using the best classification accuracy (ACC) and precision (P) are not very effective in selecting good elements for a new offspring in the crossover operator. On the other hand, both networks selected using the best true negative rate (TN) and the best true positive rate (TP) improve the working of the crossover operator. Except for two tests (with $N_v = 1500$ and 5000 for Net_{TN} and with $N_v = 1000$ and 5000 for Net_{TP}) they produced better results than the ones obtained when no MLP-based crossover was used.

Table 4. Mean results obtained by the crossover operators tested in the experiments in the limit of $N_{SE} = 200\ 000$ solution evaluations. For the MLP-based crossovers values better than for the ‘None’ setting are underlined.

N_v	None	Net_{ACC}	Net_P	Net_{TN}	Net_{TP}
1000	856.40	<u>856.67</u>	848.80	<u>862.40</u>	<u>863.77</u>
1250	726.47	722.30	<u>746.07</u>	<u>767.40</u>	724.37
1500	634.33	613.90	<u>638.60</u>	623.63	<u>656.23</u>
1750	553.70	<u>575.97</u>	<u>567.60</u>	<u>568.13</u>	<u>566.10</u>
2000	555.77	548.97	541.57	<u>571.83</u>	<u>562.37</u>
2250	518.17	<u>521.43</u>	<u>523.80</u>	<u>531.83</u>	<u>526.80</u>
2500	502.20	499.40	498.40	<u>505.53</u>	<u>504.27</u>
5000	670.20	659.50	666.67	656.73	668.10
10000	962.60	<u>965.23</u>	960.43	<u>968.93</u>	<u>969.30</u>

Table 5 presents mean running times of the methods tested in the experiments. For the MLP-based crossovers running times shorter than for the ‘None’ setting are underlined. The running times for the crossover using Net_{TN} are

from 4% faster to 8% slower than when informed crossover is not used (the ‘None’ setting). The crossover using Net_{TP} is from 2% faster to 4% slower than the method without MLP-based crossover. Decreasing the running times when using more expensive MLP-based operators is possible, because the cost of running simulations used for evaluating solutions in the FFP is lower for better solutions (fire is contained and the simulation stops).

Table 5. Mean running times (seconds) obtained by the crossover operators tested in the experiments. For the MLP-based crossovers running times shorter than for the ‘None’ setting are underlined.

N_v	<i>None</i>	Net_{ACC}	Net_P	Net_{TN}	Net_{TP}
1000	265.94	280.57	299.91	287.30	<u>262.08</u>
1250	445.57	463.99	491.09	457.31	455.54
1500	631.15	658.64	695.04	662.95	655.62
1750	885.24	889.84	946.91	892.56	<u>873.21</u>
2000	1098.9	1131.2	1176.2	1135.2	1145.1
2250	1403.0	1407.6	1465.7	1409.5	1417.9
2500	1613.3	1692.4	1713.4	1665.7	1655.5
5000	6357.1	<u>6167.2</u>	6440.6	<u>6101.2</u>	<u>6233.4</u>
10000	24187.2	<u>22233.5</u>	<u>23019.8</u>	24264.5	<u>23540.3</u>

4 Conclusions

In this paper a method is proposed for extracting knowledge from smaller Firefighter Problem (FFP) instances and for reusing this knowledge in an informed crossover operator used when solving larger instances of the FFP. During a training run, transpositions are applied to solutions found by an evolutionary algorithm and attributes of transposed vertices are stored along with the class representing the outcome (‘0’ = there was no improvement in the objective function after the transposition, ‘1’ = there was an improvement in the objective function after the transposition). Based on this training data neural networks are trained for a classification task in which the class corresponds to the expected improvement (or lack thereof) when vertices described by a given set of attributes are transposed. Trained neural networks are used in an informed crossover operator in which a pair of vertices taken from parent solutions is classified and the predicted class is used for deciding from which parent to take the next vertex.

In the experiments the training data set was obtained in a training run on a set of 30 FFP instances with $N_v = 1000$. Neural networks trained on this data set were selected with respect to four quality measures: accuracy (ACC), precision (P), true positive rate (TP, a.k.a. recall) and true negative

rate (TN, a.k.a. specificity). The proposed crossover operator using neural networks selected according to each of the quality measures was tested on sets of 30 FFP instances with $N_v = 1000, \dots, 10000$. When networks selected using the true negative rate (TN) and the true positive rate (TP) were used the results were improved in comparison to the tests in which the MLP-based crossover operator was not used. It can be concluded that useful knowledge can be learnt on smaller instances of the FFP and that this knowledge can subsequently be reused when solving larger instances.


Acknowledgment. This work was supported by the Polish National Science Centre under grant no. 2015/19/D/HS4/02574. Calculations have been carried out using resources provided by Wrocław Centre for Networking and Supercomputing (<http://wcss.pl>), grant No. 407.

References

1. Torrey, L., Shavlik, J.: Transfer learning. In: Olivas, E.S., et al. (eds.) *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods and Techniques*, vol. 2. Information Science Reference - Imprint of: IGI Publishing, Hershey (2009)
2. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: *Proceedings of the 27th International Conference on Neural Information Processing Systems, NIPS 2014*, vol. 2, pp. 3320–3328. MIT Press, Cambridge (2014)
3. Hartnell, B.: Firefighter! An application of domination. In: *20th Conference on Numerical Mathematics and Computing* (1995)
4. Michalak, K.: Estimation of distribution algorithms for the firefighter problem. In: Hu, B., López-Ibáñez, M. (eds.) *EvoCOP 2017*. LNCS, vol. 10197, pp. 108–123. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-55453-2_8
5. Vogl, T., Mangis, J., Rigler, A., Zink, W., Alkon, D.: Accelerating the convergence of the backpropagation method. *Biol. Cybern.* **59**, 257–263 (1988)
6. Bridle, J.S.: Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In: Soulié, F.F., Héroult, J. (eds.) *Neurocomputing*. NATO ASI Series, vol. 68, pp. 227–236. Springer, Heidelberg (1990). https://doi.org/10.1007/978-3-642-76153-9_28
7. Antonioni, A., Bullock, S., Tomassini, M.: REDS: an energy-constrained spatial social network model. In: Lipson, H., et al. (eds.) *ALIFE 2014*. MIT Press (2014)
8. Michalak, K.: The Sim-EA algorithm with operator autoadaptation for the multiobjective firefighter problem. In: Ochoa, G., Chicano, F. (eds.) *EvoCOP 2015*. LNCS, vol. 9026, pp. 184–196. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16468-7_16
9. Moller, M.F.: A scaled conjugate gradient algorithm for fast supervised learning. *Neural Netw.* **6**, 525–533 (1993)
10. Arlot, S., Celisse, A.: A survey of cross-validation procedures for model selection. *Stat. Surv.* **4**, 40–79 (2010)



Exploring Coclustering for Serendipity Improvement in Content-Based Recommendation

Andrei Martins Silva, Fernando Henrique da Silva Costa,
Alexandra Katiuska Ramos Diaz, and Sarajane Marques Peres^(✉) 

School of Arts, Sciences and Humanities, University of São Paulo, São Paulo, Brazil
{andreimartins, fhscosta0993, katy.rd, sarajane}@usp.br

Abstract. Content-based recommender systems are widely used in different domains. However, they are usually inefficient to produce serendipitous recommendations. A recommendation is serendipitous if it is both relevant and unexpected. The literature indicates that one possibility of achieving serendipity in recommendations is to design them using partial similarities between items. From such intuition, coclustering can be explored to offer serendipitous recommendations to users. In this paper, we propose a coclustering-based approach to implement content-based recommendations. Experiments carried out on the MovieLens 2K dataset show that our approach is competitive in terms of serendipity.

Keywords: Content-based recommender systems · Serendipity
Coclustering · Nonnegative Matrix Factorization · Jaccard similarity

1 Introduction

A recommender system (RS) helps users find items that are useful to them. These systems work by predicting relevant items to a user, given his previous interactions with the system, e.g.: Amazon [13] recommends interesting products to its customers based on similar products they have bought, seen or liked before; Netflix [11] suggests movies based on its customers' watching history. Content-based recommendation is a successful approach for recommending items. The idea is to build a user profile from features which represents the items the user expressed interest before, compare that profile to unseen items and recommend the most similar ones [14]. Although widely adopted, this approach tends to present the problem of lack of serendipity, in which only very similar items are recommended to the user [1]. Serendipity is desirable because it allows the users to receive relevant and unexpected recommendations, which they would not be able to find autonomously [6]. Usually, content-based recommender systems (CRS) are based on global similarity - an item is similar to another regarding all their attributes. Coclustering allows finding partial similarities between items - a pair of items may be deemed similar even though the similarity occurs only over

a subset of their attributes. Therefore, we propose the use of coclustering to find serendipitous recommendations. The underlying assumption is that by finding partial similarities, the adequate balance between relevance and unexpectedness can be achieved and, thus, serendipitous recommendations, provided.

This paper is organized as follows: Sect. 2 provides some background on serendipity in the RS literature. Section 3 briefly describe the coclustering task and a special class of algorithms designed to solve this task. In Sect. 4, our contribution, a coclustering-based approach to provide recommendations, is presented along with a second recommendation approach that is used for comparison purposes. Experimental results are presented in Sect. 5. Sections 6 and 7 review some related works and conclude the paper.

2 Serendipity in Recommender Systems

Efforts to build good RSs were initially focused on reaching recommendation lists with high levels of accuracy [15]. Systems were designed to recommend items similar to those that users have liked in the past. However, recommendation strategies built on these ideas are limited because they suffer from a problem defined as over-specialization and are no longer sufficient to meet users' preferences. In fact, a RS must have mechanisms capable of recommending new items, different from those already known by the users and that meet their interests [7]. To satisfy these new needs, researchers in the RS field have worked with serendipity as an aspect to be sought when composing lists of recommendations.

In the context of RSs, serendipity is related to the quality of recommendations. In general terms, the recommendation is serendipitous if it brings relevant and unexpected items, i.e.: the items serve user's needs and are items that the users did not expect to receive; therefore, they would not have found them if they have solved their request on their own [6]. In the literature, the definitions for serendipity employ subjective terms and abstract notions that make serendipity a complex concept to understand and measure [7]. Some of these definitions are: *the experience of a user who has received an unexpected and fortuitous recommendation; how good an RS is at suggesting serendipitous items that are relevant, novel and unexpected for a particular user* [4]; *serendipitous items are, by definition, unpopular and significantly different from the user profile* [12].

The measurement of serendipity in RSs has received more attention from the scientific community in recent years [12]. Some efforts delegate the measurement of this aspect to procedures that directly involve the perception of the user [10], others propose quantitative measures based on the distance between the results produced by the method to be evaluated and those produced by a primitive prediction method [4, 15] and, finally, some authors propose measurements based on observations regarding the history of ratings and items popularity [5]. This latter strategy is adopted herein. Serendipity can be measured combining relevance and unexpectedness measures. To establish metrics for these concepts, the authors in [5] consider: a recommender system S with users u and items i ; a recommendation list L with N items; τ_{ui} as the rating given to the item i by the

user u ; μ_u as the mean rating given by u to a subset of the items in S ; an item i as relevant to a user u if $\tau_{ui} > \mu_u$; $\#\tau_i$ as the number of ratings given to the item i ; ν_S as the average number of ratings given to items in S ; an item i as popular (or expected) in S if $\#\tau_i > \nu_S^1$, otherwise it is an unexpected item. Thus, relevance of L is defined as $\frac{\sum_{i \in L} R(i)}{N}$, where $R(i) = 1$ if $\tau_{ui} > \mu_u$ and $R(i) = 0$ otherwise. Unexpectedness of L is defined as $\frac{\sum_{i \in L} U(i)}{N}$, where $U(i) = 1$ if $\#\tau_i \leq \nu_S$ and $U(i) = 0$ otherwise. A recommendation is serendipitous if it is both relevant and unexpected. Regarding to L , serendipity is defined as $\frac{\sum_{i \in L} S(i)}{N}$, where $S(i) = 1$ if $(\tau_{ui} > \mu_u) \wedge (\#\tau_i \leq \nu_S)$ and $S(i) = 0$ otherwise.

3 Coclustering

Coclustering is a data mining task that allows the extraction of relevant and particular information from data. In coclustering, rows and columns of data matrices are simultaneously grouped, enabling the discovery of structures called coclusters. The coclustering task looks for coclusters that form a bi-partition (a partition of objects that is strongly related to a partition of attributes). Each cluster of objects is such that each object belonging to it is strongly and differently related to any other objects belonging to the same cluster with respect to all clusters of attributes and vice versa [16]. According to [9], coclustering methods have two main advantages: they are more effective in dealing with the curse of dimensionality problem and provide an insightful description of clusters of objects by associating clusters of attributes with clusters of objects.

Formally, coclustering is defined in [16] as follows: given a matrix $\mathcal{X} \in \mathbb{R}^{n \times m}$, in which n and m are respectively the number of lines and columns in the matrix, let x_{ij} be the element corresponding to line i and column j ; \vec{x}_i and \vec{y}_j indicate the vectors associated respectively with line i and column j . A coclustering generates a bi-partition $\mathcal{C}_{k \times l}$ on \mathcal{X} by producing a set of $k \times l$ coclusters, that is a partition \mathcal{C}_r with k clusters of rows associated with a \mathcal{C}_c partition with l clusters of columns. The bi-partition $\mathcal{C}_{k \times l}$ optimizes a given objective function.

Non-negative Matrix Factorization (NMF) is a class of algorithms designed to solve the coclustering task. It was studied as a method for data analysis able to extract knowledge about an object from the study of its parts. Later, researchers successfully applied NMF to extract useful information from textual data [18]. Orthogonal Non-negative Matrix Tri-Factorization [18], is a tri-factorization method that decomposes the original matrix $\mathcal{X} \in \mathbb{R}_+^{n \times m}$ into three new non-negative matrices, called “factors”, $U \in \mathbb{R}_+^{n \times k}$, $S \in \mathbb{R}_+^{k \times l}$ and $V \in \mathbb{R}_+^{m \times l}$ (under certain orthogonality restrictions) by iteratively adjusting these factors according to updating rules towards minimizing objective function $J = \frac{1}{2} \|X - USV^T\|^2$, $U^T U = I$, $V^T V = I$, until it finds rows and columns partitions that best explain data. The adjustment rules are:

¹ Since $\#\tau_i$ is independent of the rating qualification, a “bad” item can still be popular.

$$U = U \odot \frac{XVS^T}{USV^TX^TU}, V = V \odot \frac{X^TUS}{VSTU^TXV} \text{ and } S = S \odot \frac{U^TXV}{U^TUSV^TV},$$

where $\|\cdot\|^2$ is the *Frobenius* norm, U^T , S^T and V^T are the respective transposed matrices, \odot is the Hadamard product and a sequence of matrices (e.g. USV) is the classic matrix product.

4 Content-Based Strategies for Recommendation

We present two content-based recommender approaches: the first is purely based on the Jaccard similarity and follows a nearest neighbors fashion [2], which was chosen as the first baseline to be used for comparison purposes; the second, introduced herein, uses Jaccard similarity combined with information from coclustering models obtained by applying ONMTF.

(a) Jaccard Similarity Recommendation: The Jaccard index (J) is often used to determine similarity among sets. In the content-based recommendation scenario, an item i or a user profile UP_u is a set of n descriptive attributes, $I_i = \{att_1, att_2, \dots, att_n\}$. If an object is represented as a set of attributes, say one as the set of attributes A and another one as the set of attributes B , one can easily use Jaccard index to calculate the similarity between such objects applying $J(A, B) = \frac{A \cap B}{A \cup B}$. This calculation represents the core of this recommendation approach. For an arbitrary user u and her respective set of items ratings, the strategy using Jaccard index is built as follows: (1) split the set of known ratings from u (Tr_u) into Tr_u^+ and Tr_u^- with $\tau_{ui} > \mu_u$ and $\tau_{ui} \leq \mu_u$, respectively; (2) build two sets of attributes, POS_u and NEG_u from items in Tr_u^+ and Tr_u^- to represent the attributes of items considered respectively positive and negative according to u ; (3) build the set of attributes for the user profile: $UP_u = POS_u - NEG_u$; (4) build the set of attributes for each candidate item CI_u ; (5) calculate $J(CI_u, UP_u)$ and (6) select top-N items in CI_u with the highest scores for recommendation.

(b) ONMTF-Based Recommendation: Matrix tri-factorization results in a model represented by the matrices U , S , V . When coclustering movies (cf. section 5), U and V provide information about clusters of movies and clusters of tags respectively, and S provide information on the relationship between clusters of movies and clusters of tags. We propose to incorporate all this information into a recommendation approach geared toward serendipitous recommendations. Figure 1 illustrates how coclustering supports the recommendation approach. Following Fig. 1, to model user interests for an arbitrary user u , consider his set of movies (items) ratings, and: (1) determine the associations between movies and clusters of movies (M_{mc}) using information from matrix U ; repeat the process with US to find associations between movies and clusters of tags (M_{tc}); (2) build two set of movies, the *liked movies* set - those for which $\tau_{ui} > \mu_u$ - and the *disliked movies* set - those for which $\tau_{ui} \leq \mu_u$ (cf. section 2); (3) establish a positive

prototype of movie clusters associations (M_{mc}^+) and a positive prototype for tag clusters associations by averaging associations (M_{tc}^+) from *liked* movies; repeat the process to establish the negative counterparts (M_{mc}^- and M_{tc}^-) by averaging associations from *disliked* movies; (4) summarize the user profile (information about which cluster of movies a user likes the most and about the set of topics the user is most interested in by subtracting negative prototypes vectors from their corresponding positive prototype vectors; (5) associate such prototypes with movies and tags clusters models in order to establish the final user profile (UP_{mc} and UP_{tc}). After obtaining the user profile, the recommendation is a matter of finding movies which share content, in some level, with the user profile. This process is summarized as follows. For an arbitrary user u : (1) calculate $J(CM_{mc}, UP_{mc})$, where CM_{mc} is the association between candidate movies and clusters of movies; (2) calculate $J(CM_{tc}, UP_{tc})$ where CM_{tc} is the association between candidate movies and clusters of tags; (3) both similarities are averaged yielding scores for candidate movies; (4) truncate² the list of candidates in the score $J = 0.25$ and, (5) from the survivor candidates, select the top-N highest scored movies for recommendation.

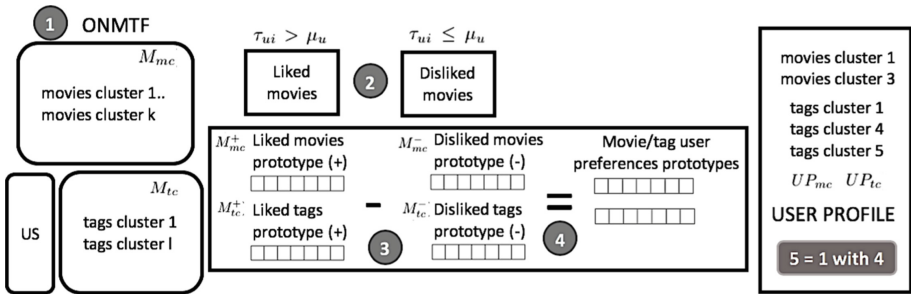


Fig. 1. Overview of cocustering for user interests modeling

5 Experiments, Results and Discussions

In this section, we describe the procedures for evaluating the recommendation approaches' performance in terms of three measures (cf. section 2), and present an example of recommendation list obtained from ONMTF-based approach.

² Since similarities among movies and tags have already been treated in the cocustering process, the maximization of J would insert an overspecialization in the process. The threshold score (0.25) was obtained empirically from extensive tests.

5.1 Dataset and Vector Representation

The MovieLens 2K dataset³ [3], used in the experiments, is a dataset with meta-data of movies and anonymous user ratings for movies. Movies are items that must be recommended by the system S in the experiments. Tags are assigned to movies by users so that each tag may have been assigned zero or multiple times. Users express their judgment by assigning ratings to movies in a ten-point scale ranging from 0.5 to 5 with 0.5 steps. Some statistics about MovieLens 2K dataset are: 2,113 users, 20,197 movies, 20 genres, 13,222 tags, 855,598 ratings, 22,696 tags per user (avg), 8,117 tags per movie (avg), 2,040 genres per movie (avg), 404,921 ratings per user (avg) and 84,637 ratings per movie (avg).

Preprocessing procedures were performed on the dataset. Such procedures involved the choice of a subset of movies, a subset of tags used as movies descriptors and a subset of users. Only tags associated with movies by more than one user were retained, since we considered that tags assigned only once to a movie carries no relevant meaning. There are movies with no meaningful tags assigned to it, and they were discarded from the original dataset. The final subset was composed by 2,004 movies and 1,101 tags. Only users who evaluated more than 25 movies (1,967 users) were maintained so that a cross-validation procedure could be performed in the experiment. A vector representation for movies was built following a vector space model in which movies are vectors and tags are features. Thus, the dataset was transformed in a movies/tags matrix $\mathcal{X}^{2004 \times 1101}$, with cells filled in according to presence or absence of a tag assigned to a movie, i.e., $\vec{x}_i = \{w_{i,1}, \dots, w_{i,j}, \dots, w_{i,M}\}$ where \vec{x}_i is a movie, N is the number of movies, $i = \{1 \dots N\}$, M is the number of tags, $j = 1 \dots M$ and $w_{i,j} \in \{0, 1\}$, depending on whether the tag j is assigned to the movie i .

5.2 Procedures and Results

To analyze the recommendation approaches, recommendation lists were generated and evaluated through a procedure inspired in those carried out in [5] and composed by six steps: (1) the subset of movies rated by u , \mathcal{X}_u , is split into five folds of movies; (2) the user profile is built from four folds (training set); (3) the lists L_a and L_b , with five items, are built by applying the recommendation approaches to select the most appropriate candidate movies from the remaining fold (test set); (4) L_a and L_b are evaluated in terms of relevance, unexpectedness and serendipity (cf. section 2); (5) the steps 2, 3 and 4 are repeated for each of the five folds. Results are averaged for each metric, and (6) common measures of position are extracted from 1,967 repetitions of the whole aforementioned procedure (one execution for each user). Wilcoxon test is run over final results to verify whether or not the means are significantly different.

Relevance, unexpectedness and serendipity were calculated for both recommendation approaches. Table 1 shows the distribution of all quality scores.

³ The original Movielens dataset is provided by GroupLens research group (<http://www.grouplens.org>). In this study, the following files were used: `movies.dat`, `movie_tags.dat`, `tags.dat` and `user_ratedmovies-timestamps.dat`.

According to relevance measures, the recommendation strategy based on Jaccard index yields better scores, with $p < 0.001$ in Wilcoxon test for *mean*. This result implies that, generally, Jaccard-based approach recommends movies that meet the interests of users more accurately, i.e., such recommendations suggest items whose rating would be higher than the average rating for that user. Even though ONMTF achieved lower relevance scores, it also provides fairly accurate recommendations most of the time, i.e., at least three relevant items out of five items in the recommendation list, in 50% of the time.

Table 1. Quality of recommendations

	Relevance		Unexpectedness		Serendipity	
	Jaccard	ONMTF	Jaccard	ONMTF	Jaccard	ONMTF
Min.	0.1600	0.1600	0.0000	0.0000	0.0000	0.0000
1st Qu.	0.6400	0.5200	0.0400	0.1200	0.0000	0.0400
Median	0.7200	0.6400	0.0800	0.2000	0.0400	0.0851
Mean	0.7194	0.6211	0.1122	0.2105	0.1235	0.1673
3rd Qu.	0.8400	0.7200	0.1600	0.3167	0.1167	0.2000
Max.	1.0000	0.9600	0.7200	0.7600	1.0000	1.0000

Regarding the evaluation of unexpectedness, the results in Table 1 shows that the ONMTF-based approach overcomes that based on Jaccard, with $p < 0.001$ in Wilcoxon test. In 50% of the recommendation lists, ONMTF-based approach provides at least one unexpected item while Jaccard recommends one unexpected item in approximately 12% of the time. When it comes to serendipity, ONMTF-based approach also surpass Jaccard-based approach, with $p < 0.001$ in Wilcoxon test. Most of the time, both approaches struggle to offer serendipitous recommendations. However, ONMTF-based approach shows at least one serendipitous recommendation 25% of the time, while Jaccard-based approach shows at least one serendipitous item in 17% of the time.

5.3 Analysis of a Recommendation List

According to the results presented in Sect. 5.2, the ONMTF-based approach is capable of contributing to the generation of serendipitous recommendations at least once every five recommendations. Since there is clearly room for improvement, it is desirable to better understand how a serendipitous recommendation can be achieved. For this purpose, consider the example related to a specific user who has requested a recommendation from the system S.

Movies are split and combined into prototype vectors according to positive and negative reviews (cf. section 4). These vectors reveal user interests in terms of genres and topics. User 11114 topic preferences (extracted from the factor US) and genre interests (extracted from factor U) are shown in the Figs. 2 and 3,

respectively. In the Fig. 2, the size of words corresponds to the relevance of topics to that user (the higher the value in US is, the more appropriate to represent a subset of movies a tag cluster is, and the more frequent in these movies a tag from such tag cluster is, the bigger the tag is in the cloud). In Fig. 3, each cluster is represented by its six more representative movies (the higher the value in U is, the more representative for the cluster the movie is). The colored dots indicate the genre associate to each cluster. Clusters were labeled based on genre most often associated with their six more representative movies.



Fig. 2. User 11114: Topic interests



Fig. 3. User 11114: Genre interests

The recommendation approach takes the captured interests as a basis and search for movies similar to the user profile. As described before, in order to balance relevance and unexpectedness, the strategy looks for moderate resemblance (between 0% and 25%) instead of maximize resemblance between candidate movies and user profile. Finally, the recommendation list is composed by the top- N most adequate items according to such resemblance. The recommendation list produced for the user 11114, the rating assigned to each item by the user and the quality reached by each recommended item are shown in Table 2.

Table 2. Recommendation list for user 11114

Movie title	Rating	Genre	Relevance	Unexpectedness	Serendipity
Waiting. . .	4.0	Comedy	x	x	x
Juno	4.5	Comedy	x		
Spy Kids 2	1.0	Adventure		x	
The Matrix Revolutions	2.5	Adventure			

The first recommendation - Waiting... - is a serendipitous recommendation. It adheres to the user profile as it features elements of comedy and it is also an unexpected recommendation (this is not a popular item in the system S). Juno is also aligned to the user profile, but is rather popular and thus, not an unexpected recommendation. Spy Kids 2 has elements of the user profile such as adventure and comedy veins and it is also unexpected, however it fails to meet the relevance criteria. Even though Matrix Revolutions shows some resemblance with the user profile (adventure, action, thriller, science fiction, dystopia, future, psychology), however it fails to meet both relevance and unexpectedness criteria.

6 Related Works

Much attention has been recently brought to the serendipity on RS. In [12], the authors outlined the state of the art in serendipity for RS. They suggested that serendipity-oriented algorithms and evaluation metrics should take into account both item popularity and similarity to a user profile. They highlighted the potential of context-aware and cross-domain RS for serendipitous recommendations, since such approaches can make use of additional information rather than just the user preferences. In [17], the authors defended the extraction of interests from user activity on Twitter to suggest serendipitous connections. Their algorithm extracts about 11% of serendipitous terms from user activity. They concluded that extractions from user's tweets are more likely to extract relevant terms, and the enrichment from web pages can bring the unexpectedness component for serendipity connections. In [10], authors presented a model that combines the cosine similarity and an unexpectedness model that recommends serendipitous news articles. A museum tour recommender is presented in [8], where the authors proposed a hybrid RS that combines a content-based approach and serendipity heuristics to provide serendipitous artwork recommendations.

In the movie recommendations context, a framework was developed by [19] which balance degrees of relevance and surprise in recommendations. Knowledge infusion process into a random walk algorithm was proposed in [5] in order to produce serendipitous recommendations. They conducted an *in vitro* experiment similar to the experiment described in this paper and achieved on average, 15% of serendipitous items in recommendation lists.

7 Conclusion

In this paper, we introduced the use of coclustering for producing serendipitous recommendations in CRSs. Experimental results showed that our approach can produce serendipitous recommendations, achieving one serendipitous recommendation 25% of the time. However, the lack of serendipity in CRSs is still an open research question, mainly because it is difficult to generate recommendations out of the obvious path, and that still remain relevant to the target user. Related works that have achieved advances in serendipitous recommendations reports success rates about 10% to 15%. At present, direct comparisons between our approach and those presented in related works are not feasible, since the conditions of experimentation are not fully compatible. In the future, we intend to explore the tri-factorization model in other ways to improve recommendations and allow direct comparisons. Besides, evaluation with user studies are planned.

References

1. Balabanović, M., Shoham, Y.: Fab: content-based, collaborative recommendation. *Commun. ACM* **40**(3), 66–72 (1997)
2. Billsus, D., Pazzani, M.J.: User modeling for adaptive news access. *User Model. User-Adapt. Interact.* **10**(2–3), 147–180 (2000)
3. Cantador, I., Brusilovsky, P.L., Kuflik, T.: 2nd Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec2011). ACM (2011)
4. Ge, M., Delgado-Battenfeld, C., Jannach, D.: Beyond accuracy: evaluating recommender systems by coverage and serendipity. In: *Proceedings of the 4th ACM Conference on Recommender Systems*, pp. 257–260. ACM (2010)
5. de Gemmis, M., Lops, P., Semeraro, G., Musto, C.: An investigation on the serendipity problem in recommender systems. *Inf. Process. Manag.* **51**(5), 695–717 (2015)
6. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.* **22**(1), 5–53 (2004)
7. Iaquina, L., De Gemmis, M., Lops, P., Semeraro, G., Filannino, M., Molino, P.: Introducing serendipity in a content-based recommender system. In: *8th International Conference on Hybrid Intelligent Systems (HIS 2008)*, pp. 168–173. IEEE (2008)
8. Iaquina, L., De Gemmis, M., Lops, P., Semeraro, G., Molino, P.: Serendipitous encounters along dynamically personalized museum tours. In: *Proceedings of the 1st Italian Information Retrieval Workshop (IIR 2010)*, pp. 101–102 (2010)
9. Ienco, D., Robardet, C., Pensa, R.G., Meo, R.: Parameter-less co-clustering for star-structured heterogeneous data. *Data Min. Knowl. Discov.* **26**(2), 217–254 (2013)
10. Jenders, M., Lindhauer, T., Kasneci, G., Krestel, R., Naumann, F.: A serendipity model for news recommendation. In: Hölldobler, S., Krötzsch, M., Peñaloza, R., Rudolph, S. (eds.) *KI 2015. LNCS (LNAI)*, vol. 9324, pp. 111–123. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24489-1_9
11. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. *Computer* **42**(8), 30–37 (2009)
12. Kotkov, D., Wang, S., Veijalainen, J.: A survey of serendipity in recommender systems. *Knowl.-Based Syst.* **111**, 180–192 (2016)

13. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Comput.* **7**(1), 76–80 (2003)
14. Lops, P., de Gemmis, M., Semeraro, G.: Content-based recommender systems: state of the art and trends. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.) *Recommender Systems Handbook*, pp. 73–105. Springer, Boston (2011). https://doi.org/10.1007/978-0-387-85820-3_3
15. Murakami, T., Mori, K., Orihara, R.: Metrics for evaluating the serendipity of recommendation lists. In: Satoh, K., Inokuchi, A., Nagao, K., Kawamura, T. (eds.) *JSAI 2007. LNCS (LNAI)*, vol. 4914, pp. 40–46. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78197-4_5
16. Pensa, R.G., Boulicaut, J.F., Cordero, F., Atzori, M.: Co-clustering numerical data under user-defined constraints. *Stat. Anal. Data Min.* **3**(1), 38–55 (2010)
17. Piao, S., Whittle, J.: A feasibility study on extracting twitter users' interests using NLP tools for serendipitous connections. In: 3rd International Conference on Privacy, Security, Risk and Trust and 3rd International Conference on Social Computing, pp. 910–915. IEEE (2011)
18. Yoo, J., Choi, S.: Orthogonal nonnegative matrix tri-factorization for co-clustering: multiplicative updates on Stiefel manifolds. *Inf. Process. Manag.* **46**(5), 559–570 (2010)
19. Zheng, Q., Ip, H.H.: Customizable surprising recommendation based on the tradeoff between genre difference and genre similarity. In: *International Conference on WEB Intelligence and Intelligent Agent Technology*, vol. 1, pp. 702–709. IEEE (2012)



Weighted Voting and Meta-Learning for Combining Authorship Attribution Methods

Smiljana Petrovic¹(✉), Ivan Petrovic², Ileana Palesi¹,
and Anthony Calise¹

¹ Iona College, New Rochelle, NY 10801, USA
spetrovic@iona.edu

² Bronx Community College of CUNY, Bronx, NY 10453, USA

Abstract. Our research concentrates on ways to combine machine learning techniques for authorship attribution. Traditionally, research in authorship attribution is focused on the development of new base-classifiers (combinations of stylometric features and learning methods). A large number of base-classifiers developed for authorship attribution vary in accuracy, often proposing different authors for a disputed document. In this research, we use predictions of multiple base-classifiers as a knowledge base for learning the true author.

We introduce and compare two novel methods that utilize multiple base-classifiers. In the Weighted Voting approach, each base-classifier supports an author in proportion to its accuracy in leave-one-out classification. In our Meta-Learning approach, each base-classifier is treated as a feature and methods' predictions in leave-one-out cross-validation are used as training data from which machine learning methods produce an aggregated decision.

We illustrate our results through a collection of 18th century political writings. Anonymously written essays were common during this period, leading to frequent disagreements between scholars over their attribution.

Keywords: Authorship attribution · Combining classifiers · Meta-Learning

1 Introduction

1.1 Authorship Attribution

Authorship attribution is the task of identifying the author of an anonymous text or a text whose authorship is in doubt [1]. While many text mining applications analyze the content of a document as an important indicator for classification, authorship attribution usually focuses on the style of the document rather than its contents; typically, all candidate authors write about related topics and often use similar, topic-specific words

This research was partially supported by the generous grant from the Robert David Lion Gardiner Foundation to Iona College's Institute for Thomas Paine Studies (ITPS).

© Springer Nature Switzerland AG 2018

H. Yin et al. (Eds.): IDEAL 2018, LNCS 11314, pp. 328–335, 2018.

https://doi.org/10.1007/978-3-030-03493-1_35

and phrases. However, stylistic features are often used unconsciously and consistently; when correctly identified, these may reveal the identity of the author.

In this research, we approach authorship attribution as a classification task. The candidate authors are represented in the training data by samples of their work. A selected machine learning algorithm uses documents of known authorship (training examples) to train the system to recognize each author's writing style. Upon completion of training, the created model may be used to associate each document of disputed authorship with one of the available candidate authors.

A majority of work in the authorship attribution field focuses on developing new stylistic features. Application papers often either use a single feature or compare the performance of several methods. We found only two applications of multiple methods in authorship attribution. Ensemble methods where different classifiers are formed by choosing different subsets of values of a single feature were employed in [2]. A mixture of three methods using simple majority voting was described in [3]. We developed a methodology to combine a large number of unrelated stylistic features and learning methods.

This work is part of a larger project by our interdisciplinary team whose primary goal is to investigate the authorship of unattributed political and economics writings from the latter half of the eighteenth century, which was a turbulent time in American and world history that led many political writings to be published anonymously [4, 5]. The correct attribution of writings allows us to gain a better understanding of the political and social ideology of the authors.

1.2 Base Classifiers

A *base-classifier* is a pair consisting of a stylistic feature and learning method. We considered seventeen different stylistic features, as outlined in Table 1, which have been adopted by the authorship attribution community [6, 7].

For each stylistic feature, the fifty most frequently used values are counted in each document, creating vectors of value frequencies which are then normalized, labeled by the author's name, and used as training examples for classification methods.

In our work, the primary learning method is *Support Vector Machines* (SVM), trained by Weka's implementation of the John Platt's Sequential Minimal Optimization algorithm [12]. SVM is an appropriate method for high-dimensional data, and it is widely used in authorship attribution research [7]. To add versatility and create larger ensembles, we also run experiments with the *Multilayer Perceptron* (MP) that implements a backpropagation neural network with the sigmoid activation function and the *Centroid Nearest-Neighbor* approach with cosine distance (NN). MP is another linear classifier, with accuracy similar to SVM, but much longer training time. NN has much lower accuracies compared to the two aforementioned methods, but it was included for two reasons: to have a method that was not based on linear separation in order to demonstrate that the improvement of Weighted Voting and Meta-Learning is not dependent on a specific learning method, and to investigate the effects of incorporating less accurate members on the accuracy of the combination.

Table 1. Features used in our analysis and their descriptions

Style marker	Abbreviation	Description
MW function words	MFWF	Function words are the most common words (articles, prepositions, pronouns, etc.) in the English language as defined by Mosteller-Wallace in their Federalist papers study [8]. They are topic-independent and often used in a subconscious manner
Word n-grams	WG2	Sequence of n items from a given sequence of words (in our case, n = 2)
Character n-grams	CG2, CG3	Sequence of n characters from a given sequence of characters (in our case, n is 2 or 3)
Part of speech	POS	Words in a text are identified as nouns, pronouns, adjectives, verbs, adverbs, prepositions, conjunctions, etc. Uses the Maxent tagger developed by the Stanford NLP Group [9]
POS n-grams	POSG2, POSG3	Sequence of n parts-of-speech tags (n is 2 or 3)
First word in sentence	FWIS	The first word in each sentence
Prepositions	PREP	The most common prepositions
Vowel-initial words	VIW	Words beginning with vowels
Suffices	SUF	The last three letters of every word
Coarse POS tagger	CPOST	A simplification of the normal part-of-speech tagger, neutralizing minor variations such as plural inflection (singular/plural words are grouped)
Lexical frequencies	LFREQ	Log-scaled frequencies of words from the general purpose HAL corpus as recorded in the English Lexicon Project (ELP) database [10]
Naming reaction times	NRT	Naming times from the ELP database; each word is converted to the time it takes to name that word in the database [10]
Sorted character n-grams	SCG2, SCG3	Alphabetically sorted characters in each n-gram (in our case, n is 2 or 3)
Word stems	WS	Stems of the words obtained from Porter's stemming algorithm [11]

1.3 Leave-One-Out for Assessing Accuracy and Obtaining Meta-data

Different base-classifiers often result in different attributions, and there is no optimal base-classifier that performs well in every case. To utilize all the different base-classifiers at once, we first consult them independently, record their results, and use different methodologies to aggregate their predictions into a combined classifier.

To evaluate the selected learning method, the available documents are usually divided into separate training and testing sets. A training set is used to build a model,

which is then tested on the remaining documents. In our work we adopted “leave-one-out” testing: $n - 1$ of the available n documents are used for training, while testing is carried out on the single remaining document. This procedure is repeated n times, in such a way that each document is used for testing exactly once. As a result, for each document, every base-classifier selects an author based on its learning from the remaining $(n - 1)$ documents.

1.4 Weighted Voting

One way to combine multiple base-classifiers is to implement a voting procedure where each base-classifier votes for the author it selected. In the initial, simplified version of our current methodology, we selected the author recommended by the majority of the base-classifiers (simple majority voting, results not reported). Our improved methodology evaluated the accuracy of each individual classifier, and, in voting, gave greater weights to those with higher accuracies.

For a given document, each method is assigned an accuracy according to the percentage of correctly classified documents among the remaining $n - 1$ documents in leave-one-out processing. Given k candidate authors, each method provides support (equal to its accuracy) to the author it selects, and distributes the remaining support $(1 - accuracy)$ among the remaining $(k - 1)$ authors (Eq. 1).

$$support_{method}(Author) = \begin{cases} accuracy & \text{if method selects Author} \\ \frac{1 - accuracy}{k - 1} & \text{otherwise} \end{cases} \quad (1)$$

This approach allows more accurate methods to have a greater contribution in voting for their choice of author. If we consider base-methods’ accuracies as a measure of probability that they made a correct choice, then the probability for each author is approximated by multiplying the supports the author received from each base-method (this is a simplification which assumes the base-methods are independent) (Eq. 2).

$$support(Author) = \prod support_{method}(Author) \quad (2)$$

The accuracy-weighted method selects the author with the highest overall support. To prevent any single method from overtaking the voting, we replaced accuracy 1 with 0.999.

1.5 Meta-Learning

As a result of the leave-one-out procedure, we obtained a table with rows corresponding to training documents and columns corresponding to base-classifiers, such that each intersecting cell represents the corresponding method’s predicted author for the corresponding document when learning was based on the remaining $n - 1$ documents.

In the Meta-Learning approach, we use this table as our training data: we assume that each base-classifier is an attribute, with each method’s prediction serving as a value of said attribute. For each document, the predictions of the base-methods provide a

vector of attribute values. For instance, using 17 features with a single learning method results in 17 discrete attributes whereas 17 features with 3 learning methods results in 51 discrete attributes. We then applied SVM to the attributes in order to learn the correct author.

2 Experimental Design and Results

2.1 Experimental Design

In all experiments, we used 12 groupings of authors selected among our pool of 42 authors. Some groups are based on the authors’ origin (American or European), some on the period in which the authors were writing (1790s–1800s American or 1770s–1780s American), and others by political beliefs (Whigs-American).

Instead of using the original papers, we represented each author with five documents, created by combining all available work of a given author and separating it into five documents of approximately equal size. We found that this approach was more robust, particularly in cases when authors were originally represented by only a few writings (data not provided).

Individual base-classifiers were trained using the leave-one-out schema. Weighted Voting and Meta-Learning were applied to selected ensembles. Results were obtained using the JGAAP (Java Graphical Authorship Attribution Program) open source software [13], implemented WEKA libraries [12], and programs written by members of our research team.

2.2 Experimental Results

Below we report the comparison of average accuracies of base-classifiers, the highest base-classifier accuracy (achieved by different methods in different experiments) with

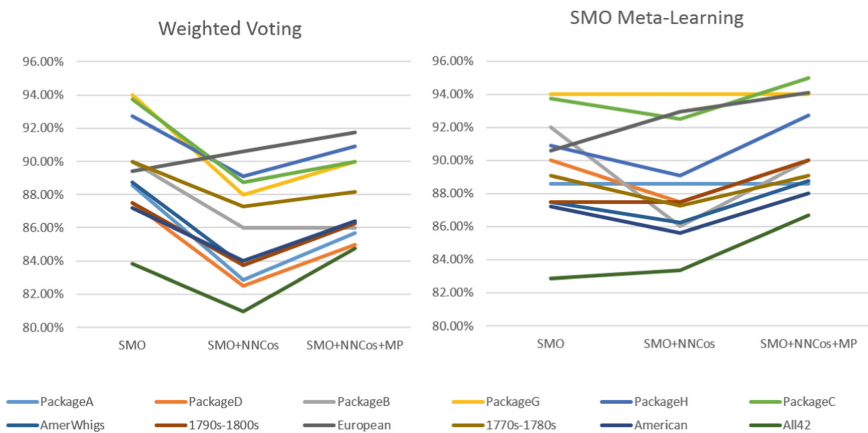


Fig. 1. The trend in accuracies when seventeen individual features are combined with the SVM, SVM + NN, and SVM + NN + MP learning on different size training examples.

the methods that achieved it, and the accuracies of both the Weighted Voting and Meta-Learning with SVM approaches. For each package, we report experiments where features are combined with SVM alone (17 base-classifiers), SVM and NN (34 base-classifiers) and a combination of all three SVM, NN and MP (51 base-classifiers) (Fig. 1 and Table 2).

Table 2. Accuracies when seventeen individual features are combined with the SVM, SVM + NN, and SVM + NN + MP learning on different size training examples.

Documents	Package name/ authors	Learning methods used	Average acc.	Highest acc.	Base-classifier with highest accuracy	Weighted voting	Meta-Learning
35	Pack A (7)	SVM	75.13%	82.86%	SVM + LFREQ, SVM + SCG2, SVM + NRT	88.57%	88.57%
		SVM + NN	68.57%	82.86%		82.86%	88.57%
		SVM + NN + MP	70.92%	85.71%	MP + LFREQ, MP + SCG3	85.71%	88.57%
40	Pack D (8)	SVM	72.65%	90.00%	SVM + CG3	87.50%	90.00%
		SVM + NN	68.31%	90.00%		82.50%	87.50%
		SVM + NN + MP	70.15%	90.00%		85.00%	90.00%
50	Pack B (10)	SVM	69.41%	94.00%	SVM + MFWF	90.00%	92.00%
		SVM + NN	59.71%	94.00%		86.00%	86.00%
		SVM + NN + MP	62.12%	94.00%		86.00%	90.00%
50	Pack G (10)	SVM	79.06%	90.00%	SVM + NRT	94.00%	94.00%
		SVM + NN	70.76%	90.00%		88.00%	94.00%
		SVM + NN + MP	72.24%	90.00%	MP + CG2, SVM + NRT	90.00%	94.00%
55	Pack H (11)	SVM	78.82%	90.91%	SVM + LFREQ	92.73%	90.91%
		SVM + NN	71.87%	90.91%		89.09%	89.09%
		SVM + NN + MP	73.90%	90.91%		90.91%	92.73%
80	Pack C (16)	SVM	71.99%	86.25%	SVM + SCG2	93.75%	93.75%
		SVM + NN	64.34%	86.25%		88.75%	92.50%
		SVM + NN + MP	66.25%	86.25%		90.00%	95.00%
80	AmerWhigs (16)	SVM	67.43%	80.00%	SVM + SCG2, SVM + SCG3	88.75%	87.50%
		SVM + NN	59.56%	80.00%		83.75%	86.25%
		SVM + NN + MP	61.76%	80.00%		86.25%	88.75%
80	1790s–1800s (16)	SVM	69.34%	85.00%	SVM + CG2	87.50%	87.50%
		SVM + NN	60.92%	85.00%		83.75%	87.50%
		SVM + NN + MP	63.87%	85.00%		86.25%	90.00%
85	European (17)	SVM	75.57%	87.06%	SVM + WS	89.41%	90.59%
		SVM + NN	71.18%	87.06%		90.59%	92.94%
		SVM + NN + MP	72.55%	87.06%		91.76%	94.12%
110	1770s–1780s (22)	SVM	68.07%	84.55%	SVM + SCG2	90.00%	89.09%
		SVM + NN	60.83%	84.55%		87.27%	87.27%
		SVM + NN + MP	62.44%	84.55%		88.18%	89.09%
125	American (25)	SVM	64.00%	76.00%	SVM + CG2, SVM + SCG2	87.20%	87.20%
		SVM + NN	56.40%	76.00%		84.00%	85.60%
		SVM + NN + MP	59.06%	76.00%		86.40%	88.00%
210	All (42)	SVM	61.76%	74.29%	SVM + LFREQ	83.81%	82.86%
		SVM + NN	55.71%	74.29%		80.95%	83.33%
		SVM + NN + MP	58.21%	74.29%	MP + MFWF, SVM + LFREQ	84.76%	86.67%

3 Analysis and Conclusion

Both Weighted Voting and SVM Meta-Learning (Table 2) are consistently better than the average base-classifier accuracy, are comparable to the highest base-classifier on smaller packages, and are considerably better in experiments with larger packages. There is no singular optimal base-classifier for all experiments.

For simple majority voting, Condorcet's Jury Theorem states that the accuracy of the jury formed of competent, independent, equally accurate voters is greater than the accuracy of a single voter. Weighted Voting increases accuracy, but without the independence and equal accuracy condition, the improvement over the best individual accuracy can no longer be guaranteed. The accuracy of a very accurate single voter may not be improved and can even be reduced when its vote is combined with much less accurate voters. For instance, in experiments when the highest individual accuracy was about 90% and average accuracy around 70%, neither Weighted Voting nor Meta-Learning improved the accuracy over that of the best base-classifier.

When the initial 17 base-classifiers based on SVM learning were combined with the less accurate 17 base-classifiers based on NN learning, the accuracy of the combination decreased (Fig. 1). When then the 17 MP based base-classifiers (about as accurate as their SVM counterparts) were added, average accuracy increased and Weighted Voting accuracy bounced back. Note that adding classifiers changed the average accuracy, while the highest accuracy remained unchanged (with one exception).

The increasing size of the candidate-author pool generally leads to harder classification problems, and adversely impacts the accuracy of the base-classifiers. The accuracy of Weighted Voting tends to decrease when the pool size increases, albeit with a considerably less pronounced drop. While Weighted Voting is comparable to the most accurate base-method on the small packages, it becomes markedly better as the size increases. Weighted Voting scales better (in terms of candidate pool size) than any single base-classifier. While adding more base-classifiers can be beneficial, it comes with the caveat that adding less accurate base-methods can degrade accuracy.

Our experiments confirm that Meta-Learning can use information contained in the votes of the base-classifiers more efficiently than Weighted Voting, resulting in more accurate combined classifiers. In most regards, Meta-Learning behaves similarly to Weighted Voting; the main difference is that Meta-Learning is less sensitive to adding inaccurate base-classifiers and it clearly benefits from the general increase in number of base-classifiers. While Meta-Learning and Weighted Voting performed about the same on 17 SVM base classifiers, Meta-Learning was evidently superior on 34 (SVM + NN) and 51 (SVM + NN + MP) base-classifiers. Furthermore, Meta-Learning on 51 base-classifiers outperformed that on 17 SVM base-classifiers, despite the lower average accuracy of the base-classifiers.

Meta-Learning with a range of base-classifiers represents an improvement over any single base-classifier (since it outperforms the best one in each experiment). There is an increased computational cost in running and combining multiple classifiers, but since unexpected attribution often inspires lengthy historical research, accuracy is generally of much more importance than computational cost.

4 Future Work

We plan to further investigate the impact of the correlation of base-classifiers in voting and the possibility of reducing the number of voters based on voting similarities. Another direction we are pursuing is the consideration of individual voters' strengths of preference toward candidate authors in voting schema.

References

1. Love, H.: *Attributing Authorship: An Introduction*. Cambridge University Press, Cambridge (2002)
2. Stamatatos, E.: Authorship attribution based on feature set subsampling ensembles. *Int. J. Artif. Intell. Tools* **15**, 823–838 (2006). <https://doi.org/10.1142/S0218213006002965>
3. Ryan, M., Noecker, J.: Mixture of Experts Authorship Attribution Notebook for PAN at CLEF 2012 (2012)
4. Berton, G., Petrovic, S., Ivanov, L., Schiaffino, R.: Examining the Thomas Paine Corpus: automated computer authorship attribution methodology applied to Thomas Paine's writings. In: Cleary, S., Stabell, I.L. (eds.) *New Directions in Thomas Paine Studies*, pp. 31–47. Palgrave Macmillan US, New York (2016). https://doi.org/10.1057/9781137589996_3
5. Petrovic, S., Berton, G., Campbell, S., Ivanov, L.: Attribution of 18th century political writings using machine learning. *J. Technol. Soc.* **11**, 1–13 (2015). <https://doi.org/10.18848/2381-9251/CGP/v11i03/56506>
6. Stamatatos, E.: A survey of modern authorship attribution methods. *J. Am. Soc. Inf. Sci. Technol.* **60**, 538–556 (2009). <https://doi.org/10.1002/asi.21001>
7. Koppel, M., Schler, J., Argamon, S.: Computational methods in authorship attribution. *J. Am. Soc. Inf. Sci. Technol.* **60**, 9–26 (2008). <https://doi.org/10.1002/asi.20961>
8. Mosteller, F., Wallace, D.L.: *Inference and disputed authorship: the Federalist*. Center for the Study of Language and Information (1964)
9. Toutanova, K., Klein, D., Manning, C.D., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: *NAACL 2003*, pp. 173–180. Association for Computational Linguistics, Morristown (2003)
10. Balota, D.A., Yap, M.J., Cortese, M.J., et al.: The English lexicon project. *Behav. Res. Methods* **39**, 445–459 (2007)
11. Porter, M.F.: An algorithm for suffix stripping. *Program* **14**, 130–137 (1980). <https://doi.org/10.1108/eb046814>
12. Hall, M., Frank, E., Holmes, G., et al.: The WEKA data mining software. *ACM SIGKDD Explor. Newsl.* **11**, 10 (2009). <https://doi.org/10.1145/1656274.1656278>
13. Juola, P.: Authorship attribution. *Found. Trends® Inf. Retr.* **1**, 233–334 (2008). <https://doi.org/10.1561/1500000005>



On Application of Learning to Rank for Assets Management: Warehouses Ranking

Worapol Alex Pongpech^(✉)

Faculty of Applied Statistics, NIDA, Bangkok 10240, Thailand
worapol@as.nida.ac.th

Abstract. The number of connected Internet of Things devices is forecast to grow to more than 31 billion globally in 2018. Many of these devices are/will be connecting to physical assets, and data collected from these devices can be used to model, manage, or describe these assets. We are motivated by utilizing these abundant data for assets' resources requirements ranking. Multiple Criteria Decision Analysis, MCDA, has often been utilized in finding ranking by computing on collected data and predefined criteria. However, changes in asset's environment have a direct impact on its resource requirements ranking and decisions on ranking must be constantly revised. This is a repetitive process where managers require to repetitively adjusting the decisions on the ranking. With machine learning, such repetitiveness can be heavily minimized by teaching machines how to rank not by instruction but rather by examples of the task being done. In this paper, we present Learn to Rank, LTR, machine learning framework in conjunction with MCDA for warehouse resources requirements ranking application. A framework for smart contract renegotiate resources allocation ranking for each asset on the blockchains is also discussed.

Keywords: IoT · Machine learning · Learn to Rank
Multiple criteria decision analysis · Resources allocation
Smart contract

1 Introduction

The number of connected Internet of Things devices is forecast to grow to more than 31 billion globally in 2018. Many of these devices are/will be connecting with physical assets, and data collected from these devices can be used to model, manage, or describe them. Recently, interests in coupled blockchains with IoTs to provide smart applications is on the rise. A solid fundamental discussion on blockchains and smart contracts for the internet of things can be found in [1]. [2–4] discuss and present some novel smart contracts applications.

Resource allocation is a fundamental to strategic asset management where it is a process and strategy involving a company deciding where scarce resources should be used in the production of goods or services. A key asset for most

logistic business is its warehouses, and managing resources allocation requirements effectively and efficiently for these warehouses is a very important business objective. Supermarket stores is a good example of a key asset for retailing businesses. Managing resource allocation effectively for these stores or warehouses is a fundamental requirement for business management. One solution for managing resource allocation is to rank assets based on their resources requirements. Such solution must allows managers to choose the types of resources to be monitored so that the collected data are sufficiently reflecting assets resources requirements.

Multiple-criteria decision analysis, MCDA, is concerned with structuring and solving decision and planning problems involving multiple criteria, and many resources allocation problems such as prioritizing, ordering, and ranking can be addressed through MCDA. Informed decision can be made by managers based on MCDA's calculation. However, changes in asset's environment have a direct impact on its resource requirements ranking and decisions on ranking must be constantly revised. This is a repetitive process where managers require to repetitively adjusting the decisions on the ranking. With machine learning, such repetitiveness can be heavily minimized by teaching machines how to rank not by instruction but rather by examples of the task being done. Machine learning approach that have been utilized successfully in information retrieval is Learn to Rank, LTR.

When consider both approaches, it could be suggested that both are tightly coupled. One is ranking these items most effectively while the other is predicting item's ranking from a given set of previous ranking lists. While there have not been many works combining the two approaches, an example is given in [5] whose work is on developing a hybrid methodology that integrates machine learning algorithms with multi-criteria decision analysis (MCDA) techniques to effectively conduct multi-attribute inventory analysis.

In this paper, we propose a Learn to Rank and MCDA application framework for warehouses ranking based on data collected from IoTs and stored on Blockchains. We also present renegotiate framework for each node on the blockchains. The paper is organized as follows: Sect. 2 describes related works in IoT, distributed ledger, and ranking process. Following that, Sect. 3 introduces warehouses ranking framework. Finally, Sect. 4 summarizes our discussion and highlights the main points presented.

2 Related Works

IoT standards are a set of established requirements on technical systems while IoT protocol is a common set of rules and instructions that each IoT follows when communicate with the others. There are a number of international standard bodies and companies heavily involved in guiding and shaping the future of IoT [6]. More detail on future direction on IoT can be found on [7].

IoT protocols dictate how devices communicate with the others. The goal is the interoperability of diverse communication systems with standard protocols between devices. Following the OSI model, Iot Protocol can be classed into four

main layers : data link, network, transport, and applications layers. The data link layer such as IEEE 802 provides node-to-node data transfer between two directly connected nodes. It defines the protocol to establish and terminate a connection between two physically connected devices, and defines the protocol for flow control between them. The network layer such as IPv4/IPv6 manages data addressing and delivery between networks. The transport layer such as TCP Transmission Control Protocol and UDP User Datagram Protocol manages the transfer of data, and assures that the received data are identical to the transmitted data. The application layer is the OSI layer closest to the end user, and allow users to interact with software applications that implement a communicating component.

Blockchains is an distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way. To use blockchains as a distributed ledger, a peer-to-peer network collectively that follows a protocol for inter-node communication and validating new blocks such as Ethereum is needed [1]. Once recorded, it is a write once and read many process where the data in any given block cannot be altered retroactively without the alteration of all subsequent blocks, which requires collusion of the network majority. Blockchains can be used to record events where each block is identifiable by a hash, generated using the SHA256 cryptographic hash algorithm on the header of the block. Each block also contains a cryptographic hash of the previous block, a timestamps, and transaction data.

A second generation of Blockchains is called Smart Contracts, which basically is blockchains with a programming capability integrated into each block. Much like physical contract, any set of rules can be assigned and enforced using Smart contracts. For example, a contract can execute trading between IoT nodes when certain events happen such as changes above threshold from the collecting data. [2,3] propose that a variety of applications such as financial instruments, autonomous governance applications, machine2machine for logistics operation can be deployed through smart contracts.

The main objective of MCDA is to structure and solve decision and planning problems when given multiple criteria. [8] presents a comprehensive review of multi criteria decision making (MCDA) towards sustainable renewable energy development. [9] investigated MCDA for resource allocation and priority Setting in Health Care. A good review on ranking methods can be found on [10] who provides a comparative overview on several rank ordering weights methods.

There are a number of methods available in MCDA such as weighted sum model, weight product model, and Promethee and Gaia method [11]. While WSM is one of the simplest, it applicable only when all the data are expressed in exactly the same unit. On the other hands, Weighted Product Model, WPM compares each decision alternative with the others by multiplying a number of ratios, one for each decision criterion. Each ratio is then raised to the power equivalent to the relative weight of the corresponding criterion. The weighted product model is often called dimensionless analysis because its mathematical structure eliminates any units of measure.

The purpose of Learn to Rank, LTR, is to learn a function automatically to rank results (items) effectively. There are three main categories of LTR are Point-wise approach, Pair-wise, and List-wise approaches. Pointwise approaches look at a single document at a time in the loss function. They essentially take a single document and train a classifier/regressor on it to predict how relevant it is for the current query. Pair-wise approach is based on a pair of item e.g., given two documents, predict partial ranking. List-wise approach is based on a ranked list of items by directly look at the entire list of documents and try to come up with the optimal ordering for it. While most of the works in LTR, is focusing on information retrieval, LTR have been applied successfully in areas other than information retrieval such as compare medical images [12].

3 Warehouses Ranking Framework

We considered three aspects of the application: input, processing, and output. The input data through IoT framework store onto the blockchains. The ranking process, MCDA and LTR pull data from the blockchains to compute ranking. Ranking output from the processing will be stored back onto blockchains where the renegotiate of the ranking can be triggered via a smart contract in the blockchains.

3.1 IoT

Data from IoT devices can be passed to MQTT broker using MQTT protocol, which is a publish/receiving message protocol working on top of TCP/IP internet protocol. MQTT broker acts like an interpreter translating a message from the formal messaging protocol of the publisher to the formal messaging protocol of the receive. In this manner, IoT devices are the publishers, warehouses are the receivers, and data are the messages as illustrate in Fig. 1.

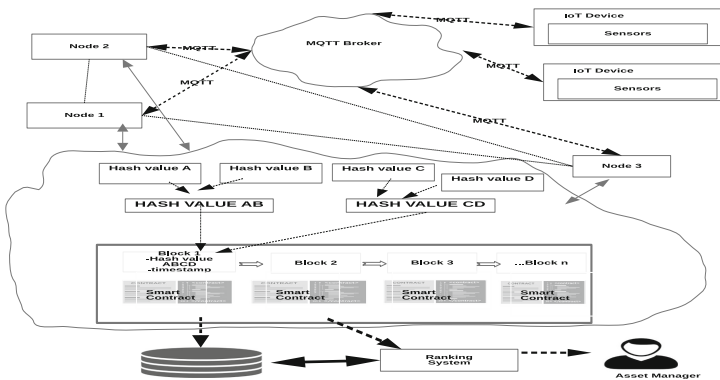


Fig. 1. IoT to blockchains

Each warehouse act as a node connecting to each other forming a connected warehouse network. With the communication happens between warehouses on the private network, there are no cryptocurrency needed and thus the need for miners is removed. The blockchains is basically a distributed data structure and replicated and shared data from warehouses among other members of the network. In this manner, the blockchains is used as a log whose records are grouped into timestamped blocks.

IoT nodes can authentication transactions using their private keys to sign the transaction. Each signed transaction can be broadcast to the nearest neighbor warehouse nodes. To ensure integrity of the transaction, each node will discards invalid transitions. The collected and validated transactions are ordered and packaged into a timestamped block.

In order to determine ranking, managers must predetermined type of features i.e. data are needed to be collected from these warehouses. To illustrate our concept, we have chose some standard keys warehouse attributes such as warehouse processing time, receiving time, put-away time, and days inventory as shown in Fig. 2.

Warehouse	Warehouse Processing Time	Receiving Time	Put-away Time	Days Inventory	
0	1	33.60	50.50	27	108
1	2	64.08	22.72	25	220
2	3	55.26	13.86	26	269
3	4	308.99	38.00	25	61
4	5	306.27	82.51	28	61
5	6	81.18	58.72	26	172
6	7	236.40	13.79	25	212
7	8	57.66	3.34	29	60
8	9	352.55	33.97	28	96
9	10	435.46	44.18	28	272
10	11	261.33	2.53	23	175
11	12	489.28	38.47	20	219
12	13	371.41	27.42	27	275

Fig. 2. Warehouses data

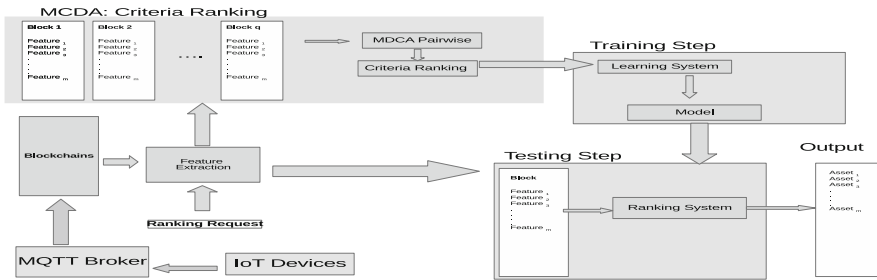


Fig. 3. Asset ranking framework credit <http://web.ist.utl.pt/~catarina.p.moreira/coursera.html>

3.2 Ranking

Warehouses ranking system framework is given in Fig. 3. Connected warehouses utilize MQTT broker to transfer data onto blockchains. When a ranking request is initiated either by managers or smart contract from nodes in the blockchains, feature extraction module send relevant criteria to the MCDA module and the LTR module. Once computed, the MCDA module send the ranking to the LTR’s training module to compute the last stage of the ranking with the testing module.

We illustrate ranking computation implement using scikit-criteria python lib, from the simulated data set using WPM, Weighted product model as shown in Fig. 4a. One of the main reason that WPM is utilized because its mathematical structure eliminates any units of measure and thus allow us to do dimensionless analysis. Another reason is its simplicity for implementation. When the initial ranking is computed through MCDA, it is passed to LTR component which compose of training and testing stages. The problem in LTR can be defined as a supervised learning task. We plan to use LTR to solve a ranking problem on a list of warehouses. The aim of using LTR is to come up with optimal ordering of those warehouse. Pair-wise approach will be implemented using scikit-learn LinearSVC where ranking is transformed into a pairwise classification task as shown in Fig. 4b. In other words, instead of giving a numeric rank to each warehouse, e.g., rank 1, 2, 3, 4 for four warehouses A, B, C, and D respectively, the classifier will judge if a warehouse is better than another for each pair of warehouse, e.g., if A is better than B, B is better than C, C is better than D, and A is better than D.

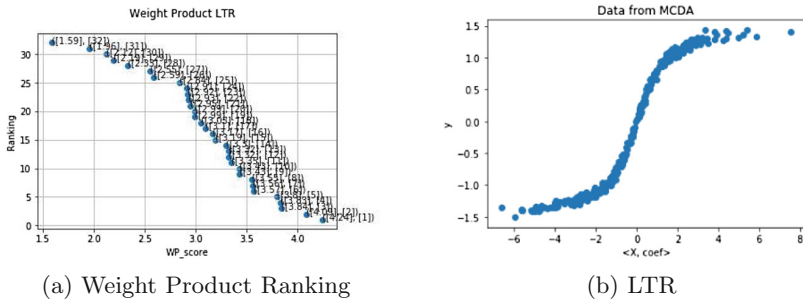


Fig. 4. MCDA and LTR ranking

Specifically, instead of working in the space of query-warehouses vectors, e.g., $x_1, x_2,$ and $x_3,$ we transform them into a new space in which a pair of warehouse is represented as the difference between their feature vectors (with respect to a query), e.g. $x_1 - x_2, x_2 - x_3,$ and $x_1 - x_3.$ For each vector $x_i - x_j,$ a label +1 is assigned if warehouse i is more important than warehouse j and for the reverse case, a label -1 is used.

Formally, training data for the ranking SVM is given as $(x_i^{(1)}, x_i^{(2)}, y_i)$, where $i = 1 \dots m$ where each instance consists of two feature vectors $(x_i^{(1)}, x_i^{(2)})$ and a label $y_i \in +1, -1$. For the pairwise approach, our goal is attempting to minimize the number of misclassifications between the pairs by using the equation below.

$$L'(F(\mathbf{x}), \mathbf{y}) = \sum_{i=1}^{n-1} \sum_{j=i+1}^n \phi(\text{sign}(y_i - y_j), f(x_i) - f(x_j)) \quad (1)$$

3.3 Output: Smart Contract

Each warehouse node has a copy of the blockchains and all interactions to the blockchains. Ethereum can be used as a decentralized platform that runs smart contract, and every node in the network is an Ethereum client responsible for networking, routing, maintain the blockchains, and running the smart contracts on the Ethereum virtual machine. Whenever data sent from a warehouse node triggers negotiation or when the warehouses manager requests new ranking, IoT nodes interact with the blockchains through a pair of private keys. The underlying events that trigger negotiation between IoT nodes have to be premeditated in advance by the asset managers. The negotiation framework can be divided into four stages as identification, Verification, Negotiation, and Execution.

1. The Identification stage focuses on identify events that would trigger the negotiation process. These events must be predetermined by the experts. The event basically correspond to change/unchanged in the data collected from sensors installed at various warehouses.
2. The Verification stage involves nodes on the blockchains network signed the contract with its address.
3. The Negotiation stage is correspond to the consensus protocol in the smart contract. The criteria for negation must be specified prior in the code. This is the part where programmer must work closely with the managers to capture sensible contract.
4. The Execution stage involves invoking the ranking process where the result of the new ranking is propagated back to the blockchains.

4 Conclusion

In this paper, we present a framework for warehouses ranking application based on MCDA and LTR. We discussed how data can be collected from IoT devices and stored onto blockchains. We illustrated the framework by implementing simple MCDA-WPM, and LTR-pairwise, approaches to demonstrate warehouse resources requirements ranking application. We concluded our work by proposed a four stages smart contract ranking request framework.

The multivariate data used in this paper is simulated, it is used to illustrated possibility of the concept. There are still a number of issues that must be carefully investigated implementing a prototype system. The first issue involves response

time of the system, the second issue involves physical aspects of the system i.e. iot devices. The last issue involves complexity versus benefit of the system in a real world setting. These issues must be investigated through an operated system, and we are currently implementing a prototype to learn more about these issues.

References

1. Christidis, K., Devetsikiotis, M.: Blockchains and smart contracts for the Internet of Things. *IEEE Access* **4**, 2292–2303 (2016)
2. Kehrlı, J.: Blockchain 2.0 - from Bitcoin transactions to smart contract applications. <https://www.niceideas.ch/roller2/badtrash/entry/blockchain-2-0-from-bitcoin>
3. Deloitte Page: Blockchain - the benefits of smart contracts. <https://www2.deloitte.com/nl/nl/pages/financial-services/articles/3-blockchain-the-benefits-of-smart-contracts.htm>
4. Yuan, Y., Wang, F. : Towards blockchain-based intelligent transportation systems. In: *IEEE 19th International Conference on Intelligent Transportation Systems (ITSC) Windsor Oceanico Hotel, Rio de Janeiro, Brazil, 1–4 November 2016*
5. Kartal, H., Oztekin, A., Gunasekaran, A., Cebi, F.: An integrated decision analytic framework of machine learning with multi-criteria decision making for multi-attribute inventory classification. *Comput. Ind. Eng.* **101**, 599–613 (2016)
6. Atzori, L., Iera, A., Morabito, G.: The Internet of Things: a survey. *Comput. Netw.* **54**(15), 2787–2805 (2010)
7. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of Things (IoT): a vision, architectural elements, and future directions. *Futur. Gener. Comput. Syst.* **29**(7), 1645–1660 (2013)
8. Abishek, K., et al.: A review of multi criteria decision making (MCDM) towards sustainable renewable energy development. *Renew. Sustain. Energy Rev.* **69**, 596–609 (2017). <https://doi.org/10.1016/j.rser.2016.11.191>
9. Aris Angelis, A., Kanavos, P., Montibeller, G.: Resource allocation and priority setting in health care: a multi-criteria decision analysis problem of value? **8**(5) 76–83 (2017)
10. Roszkowska, E.: Rank ordering criteria weighting methods - a comparative overview. *Optimum Studia Ekonomiczne* **Nr 5**(65), 14–33 (2013). <https://doi.org/10.15290/ose.2013.05.65.02>
11. Mardani, A., Jusoh, A., MD Nor, K., Khalifah, Z., Zakwan, N., Valipour, A.: Multiple criteria decision-making techniques and their applications - a review of the literature from 2000 to 2014. *Economic Research-Ekonomiska Istraživanja* **28**(1), 516–571 (2015)
12. Fabian Pedregosa, F., Gramfort, A., Varoquaux, G., Cauvet, E., Pallier, C., Thirion, B. : Learning to rank from medical imaging data. *CoRR* **1207.3598** (2012)



Single-Class Bankruptcy Prediction Based on the Data from Annual Reports

Peter Drotár¹(✉), Peter Gnip¹, Martin Zoričák², and Vladimír Gazda²

¹ Faculty of Electrical Engineering and Informatics,
Technical University of Košice, Letna 9, Košice, Slovakia
{peter.drotar, peter.gnip}@tuke.sk

² Faculty of Economics, Technical University of Košice,
Nemcovej 32, Košice, Slovakia
{martin.zoricak, vladimir.gazda}@tuke.sk

Abstract. The companies involved in all areas of the business and industry can due to the unfavourable financial situation or inappropriate investments face financial problems resulting in bankruptcy of the company. The ability to foresee imminent bankruptcy helps managers and stock holders to take the corrective actions. In this paper, we analyze annual reports of thousands of limited liability companies and propose the bankruptcy prediction model. The available dataset is strongly imbalanced that corresponds to the real-world situation where bankrupt companies constitute only a small fraction of all companies. The proposed model is based on single-class least-squares anomaly detection classifier achieving as high as 91% prediction accuracy.

Keywords: Anomaly detection · Novelty detection · Imbalanced learning
Bankruptcy · Least-squares anomaly detection

1 Introduction

The bankruptcy prediction is well-known topic that has been investigated already for several decades. The possibility to foresee the potential bankruptcy of the company, based on the data from previous time frame, is valuable for owners/investors, company management, financial institutions, lenders and state institutions. Importance of the topic is highlighted by reality that even though this issue is known for longer time it still is vivid area of research [1–3].

The first papers published on the topic considered financial ratios utilizing only one at the same time [2]. With the increasing size of the available data it was necessary to employ machine learning methods. Machine learning methods such as neural networks, support vector machines, genetic algorithm and ensemble methods showed notable improvement. The most successful appears to be random forest technique [4]. Recently, more methods were proposed for bankruptcy prediction, with extreme learning machines [5] and ensemble manifold learning [3] demonstrating promising potential.

From machine learning point of view, bankruptcy prediction is binary classification task. Considering real world scenario, where the number of bankrupt companies is

significantly lower than number of the functional companies, this is imbalanced learning problem.

Application of standard learning algorithms on imbalanced data usually does not provide the desired output. The majority class outnumbers the minority class and as such the rules induced for minority concept are fewer and weaker than those for majority class. Solution taking the imbalanced data into consideration have to be applied. Sampling methods are the frequently employed solution. Their application consists of the modification of the imbalanced dataset to achieve balanced, or close to the balanced distribution of the classes. Either the minority class is oversampled, rising number of samples, or the majority concept is undersampled. Here, the preferred solution is undersampling, since it reduces the computational complexity, and as was shown in several studies, it provides results competitive to the oversampling [7]. There are several approaches using oversampling, undersampling or combination of both. Probably, the most popular is the synthetic minority oversampling technique (SMOTE) [6]. Different approach, cost-sensitive learning, considers the different cost matrices that describe costs for the misclassifying examples [7]. The concept of cost matrix is the cornerstone of the cost sensitive learning. It is numerical representation of the penalty that characterize misclassification of particular sample (assigning sample to one class instead of the another). The proposed solutions are for example cost sensitive AdaBoost algorithms [17] or cost sensitive neural networks [8]. Even though the sampling and cost sensitive learning are the most frequently employed concepts to imbalanced data other approaches have been pursued too. The methodology that is showing significant potential is kernel based methods and active learning for imbalanced learning. Success of support vector machines (SVM) in solving many real-world problems can be naturally extended to imbalanced learning [9]. In addition to already mentioned approaches, one-class or novelty detection gained a lot of attention due to their natural suitability to imbalanced learning. One class methods, as the name indicate, tries to recognize the pattern by using only single class rather than differentiate between different classes. One class learning has been shown to be helpful in setups with extremely imbalanced dataset [10].

In this paper, we employ two single class methods, one-class SVM (OCSVM) and least-squares approach to anomaly detection (LSAD), to predict the bankruptcy of the company. The data used for analysis are strongly imbalanced containing only around one percent of samples from minority class. The features are 20 standard economic indicators derived from company annual report. The achieved results confirm that single class methods are capable of providing good prediction performance in the strongly imbalanced scenario.

The rest of the paper is organized as follows. The data are described in the next section illustrating the imbalanced nature of dataset. Then, we provide some preliminary statistical exploration of the data, where we identify the most relevant features. The experimental results on the prediction accuracy are provided in the fourth section. Finally, conclusions are drawn in the last section summarizing the contributions of the paper.

2 Data

We analyzed data containing thousands of records of limited liability companies operating in Slovak republic (European Union) within the years 2010–2016. We focus on companies from construction business. These data are part of bigger dataset consisting of different industries. Dataset provide data from annual reports three years prior to year of evaluation R . Prior to bankruptcy companies tend to display financial distress by not being able to fulfil obligations towards their creditors. Bankruptcy status is defined by law and usually has multiple forms which differs by jurisdictions. On the other hand, the non-bankrupt company is company that manages to continue to operate also for upcoming years. Annual reports analyzed in our paper are formal documents in tabular form standardized by Ministry of finance of the Slovak Republic (European Union) and are publicly accessible.

The state of each company is described by 20 standard economic variables derived from the annual report. These features are determined by the system from the annual reports. The list of considered attributes is provided in Table 1.

Table 1. Company's economic parameters used in this study

Return of Assets	ROA	Debt-to-Assets Ratio	DA
Return on Equity	ROE	Debt-to-Equity Ratio	DE
Return on Sales	ROS	Financial Leverage	FL
Cash Ratio (liquidity)	L1	Return on Investment	ROI
Quick Ratio (liquidity)	L2	Debt to Income Ratio	DIR
Current Ratio (liquidity)	L3	Debt Service Coverage Ratio	DCR
Total Asset Turnover	TAT	Asset Coverage Ratio	ACR
Asset Turnover Days	ATD	Bank Dept to Total Debt Ratio	BL
Days Total Receivables Outstanding	DTR	Labor-to-Revenue Ratio	LRR
Inventory Turnover Days	ITD	Wages to added value ratio	WAR

Table 2. Distribution of samples in classes for different years

Evaluation year R	2013	2014	2015	2016
Non-bankrupt companies	1 205	1 418	1 749	2 174
Bankrupt companies	25	30	20	14

The economical attributes are available three years prior to evaluation year i.e. $R-1$, $R-2$, $R-3$. The distribution of the samples to bankrupt class and non-bankrupt class are provided in Table 2. The non-bankrupt companies class severely out-represents the bankrupt company class. The fraction of bankrupt companies is ranging from 0.6% to 2.1% depending on the evaluation year. This represents the strong between-class imbalance.

3 Preliminary Statistical Analysis

To obtain some preliminary insight into the data we analyze importance of the features contained in dataset. There are 20 features gathered for the company every year, and three years of records so all together there are 60 predictive features available for every evaluation year R . Clearly not all features are equally important for the prediction of the target variable. We aim to identify the most important features from all available features. This is known as feature selection in the machine learning community. Vast number of feature selection methods were developed in the recent years. Here, we decided to use ReliefF [11, 12] since this is considered as one of the state-of-the art methods and provide competitive results compared to other FS methods.

We have identified five most predictive features for each year utilizing ReliefF. These are depicted in Table 3. The closer analysis reveals that one half of the selected features are from year $R-1$, i.e. one year before the evaluation. This indicates that $R-1$ year would have the strongest influence on bankruptcy prediction. This confirms our hypothesis that the most informative data comes from the year before the actual bankruptcy happens. We can expect that there are already some signs of company financial problems at this time.

Analyzing the selected parameters, it can be seen that BL was selected seven times. It was selected for each year besides 2015. Therefore, we can expect that BL parameter is one of the most significant indicators of company bankruptcy. Parameters DIR, LRR and ROI were selected twice. From other parameters only ITD, WAR, ROS, ITD, DA and ROA were selected at least once into subset of five most significant features, leaving ten attributes without selection. We can assume that to monitor company financial status one need to focus mainly on the BL and other parameters included in Table 3.

Table 3. The most significant features selected by ReliefF

Evaluation year R	Selected features		
	R-1	R-2	R-3
2013	BL	BL	LRR, ITD, DIR
2014	BL, ROS	BL	BL, WAR
2015	DA, ROS, ROI, LRR, ROA		
2016	DIR, BL ROI	DIR	BL

Based on the results provided in Table 3, we show in Fig. 1 the feature space determined by three most significant features. This partially illustrates the discriminative potential of the depicted features. Note that this does not take into account relationship between features that can be mapped or utilised by nonlinear multivariate classifier. Some samples of the bankrupt companies are distributed in the feature space as outliers and could be easily identified. There is significant part of bankrupt samples that lies in the same region as the non-bankrupt companies' samples.

4 Anomaly Detection Methods for Bankruptcy Prediction

Distribution of samples into the classes clearly demonstrates that analysed dataset represents the imbalanced learning problem. There are several approaches to imbalanced learning problem ranging from re-sampling methods through cost sensitive methods and kernel-based methods for imbalanced learning to novelty detection. Cost-sensitive techniques are gaining on popularity, however many of these techniques are specific to particular paradigm and there is no unifying framework [7]. In this paper, we focus on novelty detection approach that is expected to provide accurate results in scenarios where one class is drastically limited [10]. As can be seen from Table 3, there are only few bankrupt companies in contrast to thousands of non-bankrupt companies. This is suitable setup for novelty detection. Two approaches were utilized and compared OneClass Support Vector Machines (OCSVM) [9] and least-squares approach to anomaly detection (LSAD) [13]. In order to highlight the implication of imbalanced data scenario on classification performance, we assume the classifier that learn some concept from data and focus on achieving the highest accuracy of the classification. If all samples of dataset are classified as majority overall accuracy would be approximately 99% (assuming the distribution similar to our dataset, i.e. majority class is 99% of all samples). However, all bankrupt companies are marked as non-bankrupt.

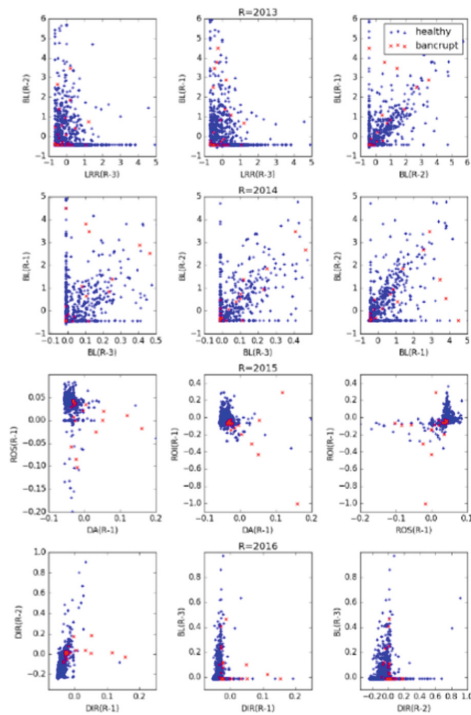


Fig. 1. The 2D map determined by the most significant features for different evaluation years

This is clearly not the wanted output. Therefore, it is evident that for this type of data we need a classifier that would provide the high accuracy for minority class while keeping the accuracy of the majority class on satisfying level. Furthermore, the conventional evaluation criteria such as accuracy score or error rate does not provide adequate information and metrics capturing imbalance needs to be employed.

To evaluate classification accuracy, we utilize geometric mean (GM). The GM is expressed as the root of the product of class-wise sensitivity and is defined as

$$GM = \sqrt{\frac{TP}{TP + FN} \times \frac{TN}{TN + FP}} \tag{1}$$

Here, the TP and TN means number of true positive and true negative, respectively. Similarly, FP stands for number of false positive and FN is false negative. Note, that in contrast to accuracy score, GM would drop to zero value if sensitivity score of one of the classes is equal to zero.

4.1 One-Class SVM

The OCSVM is built on famous Vapnik’s idea of support vector machines [14]. The starting assumption is that outliers occupy low-density region of the data space and kernel model can be used to characterize high density regions. The goal is to find function f that is able to identify points lying outside the region containing points from majority class. The strategy proposed in [9] is to map the data into the feature space corresponding to kernel and to separate them from origin with maximum margin. This can be achieved by solving quadratic programming task.

Let us first define training data as $\sqrt{x_1, x_2, \dots, x_l} \in X$ where $l \in \mathbb{N}$ is the number of observations. Additionally, let Φ be the map that maps X into inner product space F so the image of Φ is determined by evaluating kernel $k(x, y) = (\Phi(x) \cdot \Phi(y))$.

To separate data from the origin through the hyperplane the quadratic program that needs to be solved is

$$\min_{w \in F, \xi \in \mathbb{R}^l, \rho \in \mathbb{R}} \frac{1}{2} \|w\|^2 + \frac{1}{\nu} \sum_i \xi_i - \rho \tag{2}$$

$$\text{s.t. } (w \cdot \Phi(x_i)) \geq \rho - \xi_i, \xi_i \geq 0 \tag{3}$$

where $\nu \in (0, 1]$ characterizes the fraction of support vectors and outliers. It can be shown [9] that results proven for binary classification through SVM [15] are also valid for single class classifier. Then, assuming $\rho \neq 0$ holds for Eqs. (2) and (3) ν represents upper bond for fraction of outliers and lower bound on the fraction of SVM. Moreover, if the data are separable and generated independently from distribution P not containing discrete components and the kernel is analytic and non-constant, ν equals to the fraction of outliers and fraction of SVs.

For w and ρ that solve quadratics programming problem in (2) t is the decision function

$$f(\mathbf{x}) = \text{sgn}((w \cdot \Phi(x)) - \rho) \tag{4}$$

Positive for the most examples \mathbf{x}_i , while regularization term $\|w\|$ is still small. The tradeoff is controlled by variable v . As long as Φ is implicit the above optimization problem can be solved by its dual form

$$\min_a \frac{1}{2} \sum_{ij} a_i a_j k(\mathbf{x}_i, \mathbf{x}_j) \text{ subject to } 0 \leq a_i \leq \frac{1}{v^t} \quad \sum_i a_i = 1 \tag{5}$$

Then, the ρ can be determined as

$$\rho = (w \cdot \Phi(\mathbf{x}_i)) = \sum_j a_j k(\mathbf{x}_i, \mathbf{x}_j). \tag{6}$$

4.2 Least-Squares Approach for Anomaly Detection

The idea of LSAD is based on assumption similar to OCSVM but use the different loss function that makes LSAD faster and easier to train at no cost in the prediction performance. LSAD is extended application of the least squares probabilistic classification [13, 16]. Assume class labels $y_i \in Y$ corresponding to observations X and let $y_i \in \{1, \dots, c\}$ to be set of possible classes. Our aim is to estimate class conditional probabilities $p(y|\mathbf{x})$. To estimate $p(y = i|\mathbf{x})$ for each $i \in Y$ we can construct $q(y = i|\mathbf{x}, \theta_i = \theta_i^T \Phi(\mathbf{X}))$, where $\theta_i = (\theta_{i,1}, \dots, \theta_{i,B})^T \in \mathbb{R}$ for B parameters. Considering case when classes $\{c + 1, c + 2, \dots\}$ are represented only in test data but not in the training data, we need to assign value to estimate $\hat{p} = (y = *|\mathbf{x})$ for some test data \mathbf{x} . The $y = *, * \notin Y$ denote anomaly class. In this case conditional probability of an outlier can be estimated with

$$q(y = *|\mathbf{x}, \theta_*) = 1 - \theta_*^T \Phi(\mathbf{X}) \tag{7}$$

This is equal to searching for θ_* , such that (7) is close to zero when \mathbf{x} lies inside the region containing points from majority class and zero otherwise. To achieve this, we need to minimize the loss function

$$l_*(\theta_*) = \frac{1}{2} \int (1 - \theta_*^T \Phi(\mathbf{x}))^2 p(\mathbf{x}) d\mathbf{x} + \frac{\rho}{2} \|\theta_*\|^2. \tag{8}$$

It can be shown that (8) is minimized by [13]

$$\hat{\theta}_* = (\Phi^T \Phi + \rho \mathbf{I}_B)^{-1} \sum_{j \in Y} \Phi^T \mathbf{m}_j = \sum_{j \in Y} \hat{\theta}_j \tag{9}$$

and therefore $q(y = *|\mathbf{x}, \hat{\theta}_1, \dots, \hat{\theta}_S) = 1 - \sum_{j \in Y} q(y = *|\mathbf{x}, \hat{\theta}_j)$. The parameter ρ is used to regularize and to increase the sensitivity to outliers.

5 Numerical Results

We apply two one-class classification methods: OCSVM with polynomial kernel and LSAD to identify the bankrupt observations. To boost the performance we performed the grid search over the grid (degree, γ , μ) defined by the product of the sets degree = [1, 2, 3], γ = [0.01, 0.1, 1, 5], μ = [0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9], where degree is the degree of the polynomial kernel of the SVM, μ is the upper bound on the fraction of training errors and a lower bound of the fraction of support vectors, and γ is the kernel coefficient. In the case of LSAD we search through the parameters ρ = [0.01, 0.1, 1, 2, 3, 5, 10] and σ = [0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 1, 2, 5, 10]. Here, ρ controls the sensitivity to outliers and σ determines the smoothness of the boundary. Prior to further processing the data needs to be imputed since the dataset contains some amount of missing values. We use imputer on per feature basis employing mean as an imputing strategy replacing missing values using the mean along the feature. Additionally, data were scaled on per feature basis to have zero mean and unit standard deviation.

The 80% of the non-bankrupt companies were used to train classifier and the other 20% of the non-bankrupt companies together with the bankrupt companies' samples were used for testing the classifier. The prediction performance is expressed by the GM measure to take into account also imbalances in dataset. The whole procedure was repeated 1000 times with randomly selecting training and testing subset and the results were averaged over all loops. We selected this approach due to limited number of samples in minority class.

The GM scores for OCSVM classifier are summarized in Table 4. The highest GM score is achieved when data from R-1 year and R-1 & R-2 years are used for prediction. However, closer analysis reveals that the most significant contribution comes from R-1 year. This confirms our hypothesis and findings from preliminary statistical analysis that features from year R-1 were selected as the most significant. This pattern applies for all analyzed years. We can note that there are noticeable differences between GM scores for different years ranging from GM = 69.86% in 2013 to GM = 84.64% in 2015. From the economic perspective, strongest signals of a financial distress are recognizable only one year prior to the bankruptcy. Therefore, it is necessary for prediction purposes to monitor the financial situation at least on yearly basis. For small and medium-sized companies, which are subject to our analysis, a business loan is a common source of external financing and creditor is usually a bank. The most frequent factor of our analysis is bank debt to total debt ratio (BL). It implies that having a business loan increase risk of bankruptcy. There are, however, other triggering factors, which are not subject of our analysis. Another observation is that the years farther from the evaluation year contribute less to the discrimination between classes. This is expected since we can assume that financial problems are more accentuated towards the year when actual bankruptcy happens.

The second employed classifier was LSAD. We used the same data and the approach as for OCSVM. In this case, the highest GM scores are achieved for prediction using data from year R-1. This confirms that this year is the most important for

Table 4. GM of OCSVM classifier using data from various years prior to evaluation year R

Data from year	Evaluation year R			
	2013	2014	2015	2016
R-1	69.79 ± 3.41	74.14 ± 2.09	84.28 ± 2.17	80.13 ± 1.48
R-2	59.60 ± 2.87	59.52 ± 4.55	66.59 ± 5.48	67.77 ± 3.53
R-3	51.09 ± 2.08	60.55 ± 3.27	62.59 ± 2.53	64.94 ± 7.33
R-1 & R-2	69.86 ± 3.32	66.03 ± 3.69	84.64 ± 1.39	77.23 ± 2.10
R-2 & R-3	53.32 ± 2.73	56.92 ± 2.69	59.23 ± 2.76	65.37 ± 2.89
R-1 & R-2 & R-3	66.36 ± 2.86	61.85 ± 3.76	84.08 ± 2.04	73.19 ± 2.36

classification of bankrupt and non-bankrupt companies. On the other hand, we have expected that adding data from previous years would at least slightly contribute to the classification and the prediction performance would improve. However, as can be seen addition of more years prior to the bankruptcy is causing decrease in the GM score.

Even though OCSVM and LSAD have similar theoretical background, the LSAD provides much higher GM scores outperforming OCSVM by 2% in 2015 and by more than 11% in 2016.

6 Conclusions

Early identification of the imminent bankruptcy can be crucial in the decision-making process of managers and this in turn helps to save the company through the corrective actions. To provide decision support for such a situation we proposed the bankruptcy prediction model based on LSAD. The model is based on 20 economic features that are derived from company annual report and achieve more than 91% classification accuracy. Our findings indicate that the most relevant data for the bankruptcy prediction are coming from one year prior to the bankruptcy. Adding the data from other years did not improve the accuracy of the prediction, on the contrary, adding more data resulted in decrease of the GM score. In this paper, we focused solely on construction industry, but in the future, we plan to evaluate and validate our model on the dataset from other areas such as industry, services, agriculture or telecommunications and sectors that operate under oligopoly conditions.

Acknowledgements. This work was supported by the Slovak Research and Development Agency under contract No. APVV-15-0358.

References

1. Bellovary, J.L., Giacomino, D.E., Akers, M.D.: A review of bankruptcy prediction studies: 1930 to present. *J. Financ. Educ.* **33**, 1–42 (2007)
2. Altman, E.I.: Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *J. Financ.* **23**, 589–609 (1968)

3. Wang, L., Wu, C.: Business failure prediction based on two-stage selective ensemble with manifold learning algorithm and kernel-based fuzzy self-organizing map. *Knowl. Based Syst.* **121**, 99–110 (2017)
4. Barboza, F., Kimura, H., Altman, E.: Machine learning models and bankruptcy prediction. *Expert Syst. Appl.* **83**, 405–417 (2017)
5. Zhao, D., Huang, C., Wei, Y., Yu, F., Wang, M., Chen, H.: An effective computational model for bankruptcy prediction using kernel extreme learning machine approach. *Computat. Econ.* **49**, 325–341 (2017)
6. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **16**, 321–357 (2002)
7. He, H., Garcia, E.A.: Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.* **21**, 1263–1284 (2009)
8. Zhou, Z.H., Liu, X.Y.: Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Trans. Knowl. Data Eng.* **18**, 63–77 (2006)
9. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. *Neural Comput.* **13**, 1443–1471 (2001)
10. Japkowicz, N., et al.: Learning from imbalanced data sets: a comparison of various strategies. In: *AAAI Workshop on Learning from Imbalanced Data Sets* (2000)
11. Robnik-Šikonja, M., Kononenko, I.: Theoretical and empirical analysis of ReliefF and RReliefF, “ *Mach. Learn.* **53**, 23–69 (2003)
12. Kira, K., Rendell, L.A.: The feature selection problem: traditional methods and a new algorithm. In: *AAAI* (1992)
13. Quinn, J.A., Sugiyama, M.: A least-squares approach to anomaly detection in static and sequential data. *Pattern Recognit. Lett.* **40**, 36–40 (2014)
14. Vapnik, V.N.: *The Nature of Statistical Learning Theory*. Springer-Verlag, Heidelberg (1995). <https://doi.org/10.1007/978-1-4757-3264-1>
15. Schölkopf, B., Smola, A.J., Williamson, R.C., Bartlett, P.L.: New support vector algorithms. *Neural Comput.* **12**, 1207–1245 (2000)
16. Sugiyama, M.: Superfast-trainable multi-class probabilistic classifier by least-squares posterior fitting. *IEICE Trans. Inf. Syst.* **93**, 2690–2701 (2010)
17. Sun, Y., Kamel, M.S., Wong, A.K., Wang, Y.: Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognit.* **40**, 3358–3378 (2007)



Multi-dimensional Bayesian Network Classifier Trees

Santiago Gil-Begue^(✉), Pedro Larrañaga, and Concha Bielza

Universidad Politécnica de Madrid, Boadilla del Monte, Madrid, Spain
{sgil,pedro.larranaga,mcbielza}@fi.upm.es

Abstract. Multi-dimensional Bayesian network classifiers (MBCs) are probabilistic graphical models tailored to solving multi-dimensional classification problems, where an instance has to be assigned to multiple class variables. In this paper, we propose a novel multi-dimensional classifier that consists of a classification tree with MBCs in the leaves. We present a wrapper approach for learning this classifier from data. An experimental study carried out on randomly generated synthetic data sets shows encouraging results in terms of predictive accuracy.

Keywords: Multi-dimensional and multi-label supervised classification problems · Bayesian networks · Classification trees
Meta-classifiers · Hybrid classifiers · Performance evaluation measures

1 Introduction

In this paper we are interested in classification problems where there are multiple class variables. Multi-dimensional Bayesian network classifiers (MBCs) are probabilistic graphical models tailored to solving this kind of classification problem, which arises in many application domains. For example, MBCs have been used in the literature to estimate the health-related quality of life of Parkinson's disease patients [3], for sentiment analysis [13], to assist in the treatment of multiple sclerosis [16] and to predict human immunodeficiency virus inhibitors [4], among other applications.

Meta-classifiers combine different models before making a final decision motivated by the fact that there is no a learning algorithm that always induces the most accurate classifier [18]. A hybrid classifier is a meta-classifier that is induced taking into account two or more paradigms. Following this idea, we propose a novel multi-dimensional classifier that combines both well-known classification trees and MBCs. To the best of our knowledge, the *multi-dimensional Bayesian network classifier tree* (MBCTree) is the first hybrid model proposed in the context of multi-dimensional classification. We also present a wrapper approach for learning MBCTrees from data. As a wrapper strategy, we review existing performance evaluation measures to assess multi-dimensional classifiers.

The remainder of this article is organized as follows. Section 2 reviews the fundamentals of MBCs. Section 3 describes some performance measures suitable

for evaluating multi-dimensional classifiers. Section 4 describes our new multi-dimensional classifier and a wrapper algorithm for learning this classifier from data. Section 5 contains experimental results with synthetic data sets. Section 6 finally completes the article with a discussion and future work.

2 Fundamentals

2.1 Multi-dimensional Classification

We are interested in classification problems where there are multiple class variables C_1, \dots, C_d . The *multi-dimensional classification* problem consists of finding a function h that assigns a vector of d class values $\mathbf{c} = (c_1, \dots, c_d)$ to each instance given by a vector of m features $\mathbf{x} = (x_1, \dots, x_m)$:

$$h : \Omega_{X_1} \times \dots \times \Omega_{X_m} \rightarrow \Omega_{C_1} \times \dots \times \Omega_{C_d} \\ (x_1, \dots, x_m) \mapsto (c_1, \dots, c_d)$$

We assume that C_j is a discrete variable, for all $j \in \{1, \dots, d\}$, with Ω_{C_j} denoting its sample space and $I = \Omega_{C_1} \times \dots \times \Omega_{C_d}$, the space of joint configurations of the class variables. Analogously, Ω_{X_i} is the sample space of the discrete feature variable X_i , for all $i \in \{1, \dots, m\}$. When all the classes are binary, i.e., $|\Omega_{C_j}| = 2$ for all $j \in \{1, \dots, d\}$, the problem is called *multi-label classification*. Note that multi-label classification is just a particular setting of multi-dimensional classification. There are a lot of contributions to this multi-label paradigm. Two up-to-date reviews of the main proposals presented during the latest years are [7] and [19].

As stated by [2], multi-dimensional classification is a more difficult problem than the best-known single-class case. The main problem is that there is a large number of possible class label combinations, $|I|$, and a usual sparseness of available data. In a typical scenario where an instance \mathbf{x} is assigned to the most likely combination of classes (0–1 loss function), the aim is to compute $\arg \max_{c_1, \dots, c_d} p(C_1 = c_1, \dots, C_d = c_d | \mathbf{x})$. It holds that $p(C_1 = c_1, \dots, C_d = c_d | \mathbf{x}) \propto p(\bar{C}_1 = c_1, \dots, \bar{C}_d = c_d, \mathbf{x})$, which requires $|I| \cdot |\Omega_{X_1} \times \dots \times \Omega_{X_m}|$ parameters to be assigned. In the single-class, C , case, $|I|$ is just $|\Omega_C|$ rather than $|\Omega_{C_1} \times \dots \times \Omega_{C_d}|$. Besides it having a high cardinality, it is also hard to estimate the required parameters from a (sparse) data set in this d -dimensional space I . The factorization of this joint probability distribution when using a Bayesian network can somehow reduce this number of parameters, which has been studied with the MBCs.

2.2 Multi-dimensional Bayesian Network Classifiers

A Bayesian network [11, 14] over a set of discrete random variables $\{Z_1, \dots, Z_n\}$, $n \geq 1$, is a pair $\mathcal{B} = (G, \Theta)$. $G = (V, A)$ is a directed acyclic graph whose vertices V correspond to variables Z_i and whose arcs A represent direct probabilistic dependencies between the vertices. Θ is a vector of parameters such that $\theta_{z_i | \mathbf{pa}(z_i)} = p(z_i | \mathbf{pa}(z_i))$ defines the conditional probability of each possible value

z_i of Z_i given a vector value $\mathbf{pa}(z_i)$ of the parents of Z_i in G . \mathcal{B} represents a joint probability distribution $p_{\mathcal{B}}$ over the set of random variables factorized according to its structure G :

$$p_{\mathcal{B}}(z_1, \dots, z_n) = \prod_{i=1}^n p(z_i | \mathbf{pa}(z_i)).$$

Bayesian network classifiers [1] are Bayesian networks of restricted topology tailored to solving classification problems in which instances described by a number of features have to be classified in one of several distinct predefined classes. The finite set of vertices V of a Bayesian network classifier is partitioned into a set $V_X = \{X_1, \dots, X_m\}$, $m \geq 1$, of feature variables and a singleton set $V_C = \{C\}$ that corresponds to the class variable (i.e., $n = m + 1$).

An MBC is a Bayesian network specially designed to solve multi-dimensional classification problems. The graph $G = (V, A)$ of an MBC has the set V of vertices also partitioned into two sets $V_C = \{C_1, \dots, C_d\}$, $d \geq 1$, of class variables and $V_X = \{X_1, \dots, X_m\}$, $m \geq 1$, of feature variables (i.e., $n = m + d$). Note that Bayesian network classifiers are a particular setting ($d = 1$) of MBCs. The graph has also a restricted topology in which the set of arcs A is partitioned into three sets A_C , A_X and A_{CX} . The first time MBCs were proposed by [6], the three sets of arcs had the following properties:

1. The set $A_C \subseteq V_C \times V_C$ is composed of the arcs between the class variables having a subgraph $G_C = (V_C, A_C)$, *class subgraph*, of G induced by V_C ;
2. The set $A_X \subseteq V_X \times V_X$ is composed of the arcs between the feature variables having a subgraph $G_X = (V_X, A_X)$, *feature subgraph*, of G induced by V_X ;
3. The set $A_{CX} \subseteq V_C \times V_X$ is composed of the arcs from the class variables to the feature variables having a subgraph $G_{CX} = (V, A_{CX})$, *feature selection subgraph*, of G induced by V , such that for each $X_i \in V_X$, there is a $C_j \in V_C$ with the arc $(C_j, X_i) \in A_{CX}$ and for each $C_j \in V_C$, there is an $X_i \in V_X$ with the arc $(C_j, X_i) \in A_{CX}$.

MBCs were later extended by [2], such that the two conditions of the set of arcs A_{CX} were removed, and the resulting subgraph was renamed to another term:

3. The set $A_{CX} \subseteq V_C \times V_X$ is composed of the arcs from the class variables to the feature variables having a subgraph $G_{CX} = (V, A_{CX})$, *bridge subgraph*, of G induced by V .

This last definition has been mainly adopted in the literature. Figure 1 shows an example of an MBC structure and its three subgraphs. Note that the initial definition by [6] does not recognize this structure as an MBC, because for $X_3 \in V_X$, there is no $C_j \in V_C$ with $(C_j, X_3) \in A_{CX}$. The extension of [2] can be seen as a more general definition.

3 Performance Evaluation Measures for Multi-dimensional Classifiers

The evaluation of models in a multi-dimensional context needs a special approach because the simultaneous performance over all class variables should be taken

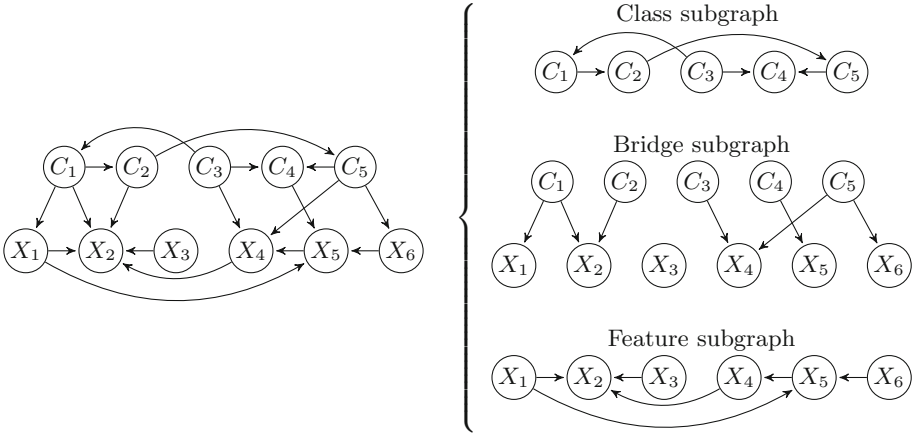


Fig. 1. An example of an MBC structure with its three subgraphs.

into account. Several performance evaluation measures have been extended to the particular multi-label setting, but only few extensions to the more general multi-dimensional classification problem are found in the literature. The most frequent measures for multi-label classification are summarized in [7].

[2] proposed the following multi-dimensional performance measures that extend those in the multi-label domain:

- *Global or joint accuracy* over the d -dimensional class variable, which extends the *multi-label 0/1 subset accuracy* [21] by computing the fraction of correctly classified examples, i.e., those whose all predicted class values are exactly the same as their corresponding true values. It is a very strict evaluation measure, especially when the size of the class space, $|I|$, is large. Let \mathbf{c}'_i be the d -dimensional binary prediction for case i in the test data set of N cases, \mathbf{c}_i its corresponding true value, and $\delta(\mathbf{c}'_i, \mathbf{c}_i) = 1$ if $\mathbf{c}'_i = \mathbf{c}_i$ and 0 otherwise, then the global accuracy is defined as:

$$Acc = \frac{1}{N} \sum_{i=1}^N \delta(\mathbf{c}'_i, \mathbf{c}_i). \tag{1}$$

- *Mean or average accuracy* over the d class variables, which evaluates the fraction of correctly classified example-class pairs. Let c'_{ij} be the C_j class value predicted by the model for case i in the test data set, c_{ij} its corresponding true value, and $\delta(c'_{ij}, c_{ij}) = 1$ if $c'_{ij} = c_{ij}$ and 0 otherwise. Then, the mean accuracy is defined as:

$$\overline{Acc} = \frac{1}{d} \sum_{j=1}^d Acc_j = \frac{1}{d} \sum_{j=1}^d \frac{1}{N} \sum_{i=1}^N \delta(c'_{ij}, c_{ij}).$$

This measure is the complementary of the multi-label *Hamming loss* [17], i.e., $mean\ accuracy + Hamming\ loss = 1$, but extended to the multi-dimensional paradigm.

4 Multi-dimensional Bayesian Network Classifier Trees

Meta-classifiers combine different models motivated by the *no free-lunch theorem* [18], which states that there is no a learning algorithm that in any domain always induces the most accurate classifier. A hybrid classifier is a meta-classifier that is induced taking into account two or more paradigms. An example of a hybrid classifier is the naive Bayes tree (NBTree) of [10], which deploys a naive Bayes model on each leaf node of a classification tree. Following a similar idea, [12] proposed the logistic model tree, with logistic regressions instead of naive Bayes classifiers in the leaves of the classification tree. Another example is the lazy Bayesian rules proposed by [20], which builds a most appropriate rule for each test instance with a local naive Bayesian classifier as its consequent.

In this paper we propose a hybrid of *classification trees* [5] and MBCs. To the best of our knowledge, this is the first hybrid model proposed in the context of multi-dimensional classification. An MBCTree is a classification tree with MBCs in the leaves (Fig. 2):

- An internal node of an MBCTree corresponds to a feature variable X_i as in standard classification trees, and has a labelled branch to a child for each of its possible values in Ω_{X_i} . The labels are the possible values of X_i .
- A leaf node of an MBCTree corresponds to an MBC over all the class variables and those feature variables not present in the path from the root to the leaf. An MBCTree may be asymmetric, so the MBCs at the leaves may have different feature variables. A new instance will be classified by sorting down the tree from the root to some MBC leaf node according to the outcome of the tests along the path.

4.1 A Wrapper Approach for Learning MBCTrees from Data

We propose a wrapper approach guided by the global accuracy (Eq. (1)) for learning MBCTrees from data, although any other multi-dimensional measure could be used. The proposed approach is detailed in Algorithm 1.

The MBCTree is learned by recursively choosing as internal node the variable X_{best} that best splits the data, i.e., that achieves the highest global accuracy [steps 1–11], until splitting no longer adds value to the predictions [steps 12–14]. The proposed approach is seen as a greedy top-down algorithm [15]. It starts at the root node of the tree and computes the global accuracy, $Acc_{MBCTree_i}$, of each feature variable X_i to split on [steps 2–8]. For this, an MBC is learned for each possible value of X_i with the corresponding portion of data [steps 3–6]. The accuracy of the split is computed by sorting the test data set to the learned

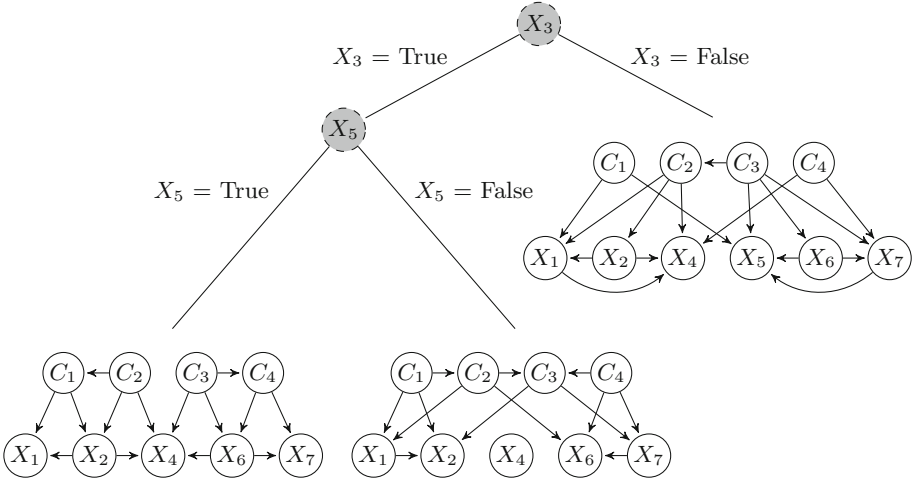


Fig. 2. An example of an MBCTree structure.

MBCs testing X_i [step 7]. We have chosen a wrapper strategy guided by the global accuracy to learn the MBCs as in [2]. This process is repeated on each derived subset of the best split in a recursive manner [step 11], until there is no split that improves the global accuracy achieved by an MBC learned with the data that reach the current node [step 1]. In that case, this MBC is placed as a leaf node [step 13]. A leaf node must be also created if there is only one feature variable left to split on, as there could not be an MBC at the leaf with no feature variables in its structure, and also if there is no enough data to keep growing the tree, i.e., to learn split MBCs. To avoid overfitting the training set, at least a minimum improvement can be required in [step 10] as a prune strategy, such that the recursion ends in [step 13] if no significant improvement is achieved.

5 Experimental Results

In order to evaluate our proposed approach we performed an experimental study on synthetic data sets. For the benefit of the community, we make the source code public¹. The study follows the steps shown in Fig. 3:

- First, a random MBCTree with fixed depth is generated. The feature variables associated to the internal nodes are randomly chosen. The MBCs leaf are also randomly generated, such that the class and feature subgraphs are uniformly distributed samples of directed acyclic graphs [9] and each class variable C_j is connected to each feature variable X_i with a probability p . We chose $p = 0.5$ in order to uniformly sample from the MBC structure space. The parameters of MBCs are forced to be extreme, i.e., lower than 0.3 and greater than 0.7.

¹ Code available at <https://github.com/ComputationalIntelligenceGroup/MBCTree>.

Algorithm 1. Wrapper algorithm for learning MBCTrees from data.

Input: A labelled data set D

Output: An MBCTree learned with the data set D

```

1: Learn an MBC and compute its global accuracy,  $Acc_{MBC}$ , with the data set  $D$ 
2: for each feature variable  $X_i$  do
3:   Split  $D$  into  $|\Omega_{X_i}|$  subsets  $D_{ij}$  based on the possible values  $j$  of  $X_i$ 
4:   for each subset  $D_{ij}$  do
5:     Learn an  $MBC_{ij}$  with the subset  $D_{ij}$ 
6:   end for
7:   Compute the global accuracy,  $Acc_{MBCTree_i}$ , of the split on  $X_i$ 
8: end for
9:  $best = \arg \max_i Acc_{MBCTree_i}$ 
10: if  $Acc_{MBCTree_{best}} > Acc_{MBC}$  then
11:   Create an internal node  $X_{best}$ . For each child node under a branch with label
        $j \in \Omega_{X_{best}}$ , call the algorithm recursively, from step 1, on the data subset  $D_{best_j}$ 
12: else
13:   Create a leaf node with the MBC of step 1
14: end if

```

- Second, a data set is simulated from the MBCTree. For this, a data subset of random size is simulated for each MBC leaf by using probabilistic logic sampling [8]. It is imposed that each subset contributes at least a fixed percentage to the whole data set.
- Third, an MBC and an MBCTree are learned following the wrapper approach in [2] and Algorithm 1, respectively. Then, they are compared in terms of predictive accuracy with the simulated data set, which is divided in a training set containing 80% of the instances and a test set with 20%. The training set is also divided in the same percentages for learning the MBCTree: 80% of the instances are used for learning the MBCs and 20% for evaluating them so that the best split can be computed. Cross-validation could have been used in both cases, but we followed a train and test strategy together with a larger simulated data set because of computational efficiency.

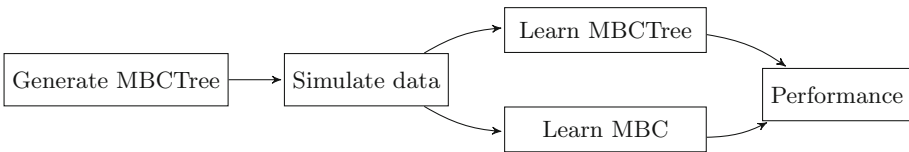


Fig. 3. Steps of the experimental study performed with synthetic data.

We established two different configurations following the aforementioned scheme. Each configuration was executed ten times, leading to the results shown in Table 1.

- Configuration A: an MBCTree of depth 1 with 11 feature variables in the MBC leaf (11 + 1 features altogether) and 4 class variables to be predicted. All the variables are binary. The simulated data set consists of 100,000 instances, with at least 20% in each branch.
- Configuration B: an MBCTree of depth 2 with 10 feature variables in the MBC leaf (10 + 2 features altogether) and 4 class variables to be predicted. All the variables are binary. The simulated data set consists of 100,000 instances, with at least 20% in each branch in a recursive manner.

Table 1. Comparison in terms of global accuracy of the MBCs and MBCTrees.

Configuration A				Configuration B			
MBC	MBCTree	Diff.	Learned	MBC	MBCTree	Diff.	Learned
0.6649	0.6941	+0.0292	101	0.7903	0.8113	+0.0210	177
0.7719	0.7972	+0.0253	101	0.7696	0.8005	+0.0309	139
0.7898	0.8053	+0.0155	59	0.7444	0.7866	+0.0422	177
0.8098	0.8216	+0.0118	37	0.7876	0.8183	+0.0307	101
0.7280	0.7382	+0.0102	79	0.8422	0.8692	+0.0270	157
0.7796	0.8070	+0.0274	81	0.7892	0.8350	+0.0458	179
0.8461	0.8630	+0.0169	59	0.7772	0.7948	+0.0176	159
0.8584	0.8664	+0.0080	37	0.7764	0.8066	+0.0302	141
0.8367	0.8474	+0.0107	121	0.7051	0.7383	+0.0332	231
0.8314	0.8577	+0.0264	37	0.7352	0.7762	+0.0410	177
<i>Average</i>		+0.0181	71	<i>Average</i>		+0.0320	164

As an encouraging result of the experimental study performed, our MBCTree model has achieved higher global accuracies than MBCs in all the executions for both configurations. Similar profits were obtained by the NBTree with respect to standard naive Bayes classifiers [10]. The improvement is more noticeable in Configuration B executions because the simulated data set comes from more different probability distributions, and the MBCTree is able to discover this data partition. Following this idea, using a configuration C with a random MBCTree of depth 3 has improved the results even more, with an average improvement of 0.0364 over ten executions. In all three cases, the accuracy improvement is statistically significant with a p -value = 0.001953 when using the Wilcoxon signed-rank test.

As a positive aspect of the proposed learning approach, all internal nodes of an initial randomly generated MBCTree have been always recovered following the same structure in the learned MBCTree, for both Configurations A and B. In addition, the algorithm has often included other extra internal nodes besides this main structure. Most internal nodes in Configuration C have been also recovered, but in a different order. For example, the root node of the initial model has been sometimes included at the second level. This is explained because of the greedy nature of the proposed learning algorithm.

In contrast, the computational burden for learning an MBCTree has to be remarked. The critical point is to compute at each node which feature variable best splits the data, as an MBC is learned for each possible value of each feature variable. On average, 71 and 164 MBCs have been needed to be learned for inducing the MBCTrees for Configuration A and B, respectively. Our method is then adding two orders of magnitude in this particular problem. This complexity could be alleviated if the MBCs were learned in parallel. Another drawback of our model is that an MBCTree needs a larger training data set because the recursive partitioning of the data makes the MBCs leaf to be learned with fewer data.

6 Conclusions and Future Research

In this paper we have proposed a novel multi-dimensional classifier that consists of a classification tree with MBCs in the leaves. An improvement in terms of predictive accuracy has been shown on randomly generated synthetic data sets, but at the expense of adding more complexity to the model learning.

As future work, we would intend to carry out a more extensive experimental study using real data sets. We have seen that learning MBCTrees is high data demanding, so a thoughtful choice of a real multi-dimensional problem is required. Moreover, it will be interesting to extend MBCTrees to deal with the challenging task of multi-dimensional classification for concept-drifting data streams. Finally, we would like to alleviate the computational burden of our proposed wrapper algorithm, for what we plan (1) to learn the MBCs for each feature value in parallel since they are independent processes, as well as to execute a parallel recursion for the branches of a split node, (2) to use a filter learning technique, and also (3) to investigate a model based on the random selection of the features to split on, within an ensemble approach.

Acknowledgements. This work has been partially supported by the Spanish Ministry of Economy, Industry and Competitiveness through the Cajal Blue Brain (C080020-09; the Spanish partner of the Blue Brain initiative from EPFL) and TIN2016-79684-P projects, by the Regional Government of Madrid through the S2013/ICE-2845-CASI-CAM-CM project, and by Fundación BBVA grants to Scientific Research Teams in Big Data 2016.


References

1. Bielza, C., Larrañaga, P.: Discrete Bayesian network classifiers: A survey. *ACM Comput. Surv.* **47**(1), 5 (2014). <https://doi.org/10.1145/2576868>
2. Bielza, C., Li, G., Larrañaga, P.: Multi-dimensional classification with Bayesian networks. *Int. J. Approx. Reason.* **52**(6), 705–727 (2011). <https://doi.org/10.1016/j.ijar.2011.01.007>
3. Borchani, H., Bielza, C., Martínez-Martí, P., Larrañaga, P.: Markov blanket-based approach for learning multi-dimensional Bayesian network classifiers: An application to predict the European quality of life-5 dimensions (EQ-5D) from the 39-item Parkinson's disease questionnaire (PDQ-39). *J. Biomed. Inform.* **45**(6), 1175–1184 (2012). <https://doi.org/10.1016/j.jbi.2012.07.010>

4. Borchani, H., Bielza, C., Toro, C., Larrañaga, P.: Predicting human immunodeficiency virus inhibitors using multi-dimensional Bayesian network classifiers. *Artif. Intell. Med.* **57**(3), 219–229 (2013). <https://doi.org/10.1016/j.artmed.2012.12.005>
5. Breiman, L., Friedman, J.H., Olshen, R., Stone, C.J.: *Classification and Regression Trees*. Wadsworth, California (1984)
6. van der Gaag, L.C., de Waal, P.R.: Multi-dimensional Bayesian network classifiers. In: *Proceedings of the 3rd European Workshop in Probabilistic Graphical Models*, pp. 107–114 (2006)
7. Gibaja, E., Ventura, S.: A tutorial on multi-label learning. *ACM Comput. Surv.* **47**(3), 52 (2015). <https://doi.org/10.1145/2716262>
8. Henrion, M.: Propagating uncertainty in Bayesian networks by probabilistic logic sampling. *Mach. Intell. Pattern Recognit.* **5**, 149–163 (1988). <https://doi.org/10.1016/B978-0-444-70396-5.50019-4>
9. Ide, J.S., Cozman, F.G.: Random generation of Bayesian networks. In: Bittencourt, G., Ramalho, G.L. (eds.) *SBIA 2002. LNCS (LNAI)*, vol. 2507, pp. 366–376. Springer, Heidelberg (2002). <https://doi.org/10.1007/3-540-36127-8.35>
10. Kohavi, R.: Scaling up the accuracy of naive-Bayes classifiers: A decision-tree hybrid. In: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, vol. 96, pp. 202–207 (1996)
11. Koller, D., Friedman, N.: *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, Cambridge (2009)
12. Landwehr, N., Hall, M., Frank, E.: Logistic model trees. *Mach. Learn.* **59**(1–2), 161–205 (2005). <https://doi.org/10.1007/s10994-005-0466-3>
13. Ortigosa-Hernández, J., Rodríguez, J.D., Alzate, L., Lucania, M., Inza, I., Lozano, J.A.: Approaching sentiment analysis by using semi-supervised learning of multi-dimensional classifiers. *Neurocomputing* **92**, 98–115 (2012). <https://doi.org/10.1016/j.neucom.2012.01.030>
14. Pearl, J.: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Burlington (1988)
15. Quinlan, J.R.: Induction of decision trees. *Mach. Learn.* **1**(1), 81–106 (1986). <https://doi.org/10.1023/A:1022643204877>
16. Rodríguez, J.D., Pérez, A., Arteta, D., Tejedor, D., Lozano, J.A.: Using multidimensional Bayesian network classifiers to assist the treatment of multiple sclerosis. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **42**(6), 1705–1715 (2012). <https://doi.org/10.1109/TSMCC.2012.2217326>
17. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. *Mach. Learn.* **37**(3), 297–336 (1999). <https://doi.org/10.1023/A:1007614523901>
18. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1**(1), 67–82 (1997). <https://doi.org/10.1109/4235.585893>
19. Zhang, M., Zhou, Z.: A review on multi-label learning algorithms. *IEEE Trans. Knowl. Data Eng.* **26**(8), 1819–1837 (2014). <https://doi.org/10.1109/TKDE.2013.39>
20. Zheng, Z., Webb, G.I.: Lazy learning of Bayesian rules. *Mach. Learn.* **41**(1), 53–84 (2000). <https://doi.org/10.1023/A:1007613203719>
21. Zhu, S., Ji, X., Xu, W., Gong, Y.: Multi-labelled classification using maximum entropy method. In: *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 274–281. ACM (2005). <https://doi.org/10.1145/1076034.1076082>



Model Selection in Committees of Evolved Convolutional Neural Networks Using Genetic Algorithms

Alejandro Baldominos^(✉) , Yago Saez , and Pedro Isasi 

Computer Science Department, Universidad Carlos III de Madrid,
Avenida de la Universidad 30, 28911 Leganes, Spain
{abaldomi,ysaez}@inf.uc3m.es, isasi@ia.uc3m.es

Abstract. Neuroevolution is a technique that has been successfully applied for over three decades in order to optimize certain aspects of neural networks by applying evolutionary algorithms. However, only in the recent years, the increase of computational resources has enabled to apply such techniques to deep and convolutional neural networks, where the number of hyperparameters is significantly large.

In recent years, deep and convolutional neural networks are outperforming classical machine learning for many different tasks, including computer vision, natural language processing, signal processing, activity recognition, etc. In this context, neuroevolution can be useful since there are no analytic approaches for determining optimal network architectures or hyperparameters, therefore attaining better performance. Moreover, in some cases, committees (also called ensembles) are used, which combine two or more models in order to improve results even more. Neuroevolution can be of particular importance in this case, since evolutionary algorithms evolve a whole population of individuals, making it easy to build ensembles out of models chosen among those in the evolved population.

In this paper, we explore the application of genetic algorithms for carrying out model selection in a context of neuroevolution. Thus, the best models will be selected from a population of evolved individuals in order to maximize an overall objective function. This approach is tested using the well-known MNIST database as benchmark, and it obtains results which are highly competitive when compared with the state of the art.

Keywords: Genetic algorithms · Neuroevolution
Convolutional neural network · Committees · Ensembles

1 Introduction

In recent years, deep learning techniques, and most remarkably those involving deep and convolutional neural networks, have arisen to become common solutions for solving a large variety of artificial intelligence (AI) problems, including

computer vision [24], natural language processing [22], sentiment analysis [28], big data analysis [29] or activity recognition [4, 18], among many others. In fact, these techniques have exceeded the boundaries of academia and are being applied pervasively in the industry, with some key players such as Google, Microsoft or Facebook using them for some core functionalities in their products.

The development and improvement of hardware specifically designed for deep learning (especially graphical processor units, GPUs) have made it feasible to train models with large amounts of data in reasonable amounts of time, thus facilitating its widely adoption by the industry. However, an important drawback of neural networks is that there are no analytic procedures for determining their optimal architecture or hyperparameters. Instead, trial and error is often the best approach for determining suitable, yet in many cases suboptimal, values for these hyperparameters.

In the late 1980s, a field later known as “neuroevolution” arose whose aim was to optimize some aspects of neural networks using evolutionary computation. Some remarkable classical works in neuroevolution comprise EPNet by Yao and Liu [26], NEAT by Stanley and Miikkulainen [20] or EANT by Kassahun and Sommer [13]. In many cases, these works were focused on learning architectures as well as weights of the neural network, although these networks were much simpler in terms of structure and number of parameters than those that we can find in today’s standards of deep learning. In particular, convolutional neural networks (CNNs) nowadays can commonly comprise several convolutional layers with a different number of filters, filter sizes, and activation functions, may implement several pooling layers, and can involve in some cases fully connected layers which can be of different types (feed-forward, LSTM, GRU...), with a different number of hidden neurons in each layer and different activation functions. And not to mention that the learning hyperparameters can be also tuned, such as the training algorithm, the learning rate or the batch size, which can be defined independently from the network architecture itself.

Classical neuroevolution techniques cannot be directly applied to such modern models, although the same underlying principles remain in both cases. In modern works; however, it is not common to use neuroevolution for learning the network parameters, since the search space would be extremely large, and the advancement of the state of the art in training algorithms have lead to efficient backpropagation-based optimizers that perform well for a very broad number of cases, such as Adam [14] or Rmsprop [12]. The improvement of hardware resources has made it possible to explore neuroevolution of these complex deep learning models, resulting in a few works published since 2014, with an important increase of this research line since 2017.

In some cases, works can be found in the literature in which committees of neural networks are used to improve performance even further, such as for medication safety analysis [1] or computer vision [7]. Committees, also known as ensembles, are a simple idea which consist on combining several models in order to return a single output. Ensembles have been well explored with other machine learning algorithms, more notoriously with decision trees, leading to

well-known techniques such as random forests [5] or extremely randomized trees [11]. In neural networks, the outputs of the different individual models need be combined into one single output, for which some policy must be defined. Usual policies involve returning the average or the median in regression problems, or the mode (most frequent output) in classification problems.

Committees of CNNs; however, remain mostly unexplored in the literature. The main reason is that, although they will often outperform single models, they require significantly more time to train and to predict (linear to the number of models), and the difficulty for determining the most suitable architecture and topologies also grows significantly. Nevertheless, neuroevolution can be useful at this point: since evolutionary algorithms work by evolving a population of individuals, all diverse models in this population can be combined to conform a committee, with no additional cost besides the evolutionary process itself. The only remaining question is how these models should be combined, or which subset of models are relevant for the ensemble.

In this paper, we will start from a previous work done by Baldominos et al. [3] in which neuroevolution is used to optimize the architecture of CNNs to solve the problem of handwritten digit recognition, and will use genetic algorithms (GA) to determine the optimal set of models to form an ensemble. This paper is structured as follows: Sect. 2 describes the state of the art of neuroevolution, citing some related works in which this technique has been applied to the evolution of ensembles. Later, Sect. 3 describes the proposal in further detail, explaining how the GA can be applied for model selection, and the results of this proposal evaluated against the MNIST database will be shown in Sect. 3. Finally, Sect. 5 will provide some conclusive remarks.

2 State of the Art

As stated in the previous section, only in recent years have hardware improvements and advances in deep learning research enabled the feasibility of neuroevolution of deep and CNNs. In 2014, a work was published by Koutník et al. [15] which to the best of our knowledge is the first documented attempt to neuroevolve CNNs. However, in this work, a fixed topology is used and only a small number of parameters is evolved. In 2015, two more works were published by Verbancsics and Harguess [23] and by Young et al. [27], and in 2016 two more works by Fernando et al. [10] and Loshchilov and Hutter [16] respectively can be found in the literature. The number of applications of neuroevolution to deep learning grows significantly in 2017, where we can find works by Xie and Yuille [25], Miikkulainen et al. [17], Desell [9], Real et al. [19], Davison [8] and Saganuma et al. [21]. Also in 2018, two works can be found by Baldominos et al. [3, 4], where neuroevolution is successfully used to optimize CNNs for handwriting recognition and human activity recognition respectively.

To the best of our knowledge, evolution of CNN committees remains still mostly unexplored. One remarkable exception is the work by Real et al. [19], where combining neural networks in an ensemble allow authors to report an

accuracy of 95.6% in the CIFAR10 dataset, instead of 94.6% which is the best accuracy they found for a single model. Authors do not provide much detail in how ensembles were built, although they mention that they work following a majority-voting policy and models were chosen by validation accuracy. Another work where neuroevolution of committees is observed is the one by Baldominos et al. [4], where authors are able to improve the F1 score over the OPPORTUNITY dataset from 91.85% to 92.75% by using a committee of 11 CNNs instead of a single model. In this paper, authors sorted the neuroevolved models by descending F1 score and added them one-by-one to the ensemble until reaching a maximum performance. Also, authors describe a niching scheme for increasing genetic diversity during the neuroevolutionary process, which may improve the ensemble performance.

Finally, it is worth mentioning the work by Baker et al. [2], which despite not using neuroevolution, also optimizes ensembles of CNNs, in this case using reinforcement learning. In their work, they increase the accuracy on MNIST from 99.56% to 99.68%, although using ensembles do not improve the results on the CIFAR10 dataset. However, authors only reported the performance of the top-5 models ensembled together, and when they tested the performance of the top-10 models with MNIST, they increased accuracy even further, up to 99.72%.

3 Proposal

In this paper, we test how it works to perform model selection over a population of neuroevolved CNN models in order to build an optimal CNN committee. As we have seen in the previous section, the problem of building committees (or ensembles) after neuroevolution has been tackled in only a couple of works, and in those cases a simple approach is followed, in which models are just sorted based on their performance and after that they are added to the ensemble until reaching maximum performance.

This “greedy” approach can be a suitable one, but will in most cases lead to suboptimal ensembles. An important issue when building an ensemble is that for it to work properly, besides models performing well, it is useful that models are diverse enough. To put it otherwise, when there is few diversity among the models, they will most likely work just as one individual model. This can be easily illustrated with an extreme case: if all models were identical, then, regardless of the size, the ensemble will perform just as this one model. However, even following a greedy procedure based on some measure of diversity or variability is not enough, since this can again produce suboptimal committees.

For this reason, we propose an algorithm that optimizes an ensemble of CNNs, whose only input is a population of CNN models whose architecture has been evolved using neuroevolution, returning as an output of the algorithm a set of models to be included in the optimal ensemble. Considering that the algorithm itself is a GA, this process is illustrated in Fig. 1.

To state this problem formally, let P be the population of evolved CNNs, which will have N models ($|P| = N, N > 1$). Also, let E be the set of individuals

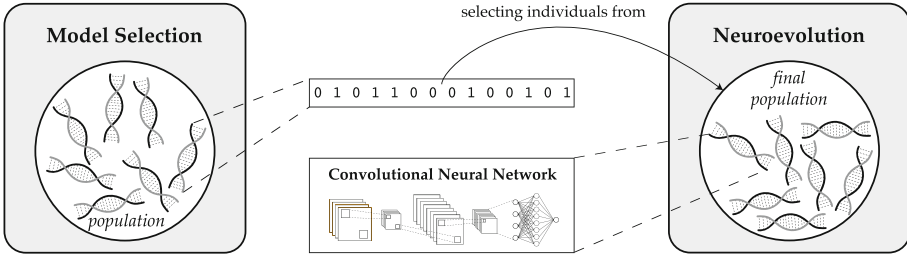


Fig. 1. Schematic illustration of the process for achieving neuroevolved model selection using genetic algorithms.

conforming the ensemble of CNNs. Of course, we need $E \subseteq P$, and also by definition $|E| > 1$ (the ensemble cannot be the empty set nor cannot be formed of only one model). Now, let $P = \{p_1, p_2, \dots, p_N\}$, with $p_i \in [1, N]$ being the neuroevolved models. We can now define E by means of a binary string of length N , as follows: $E \equiv \{b_1, b_2, \dots, b_n\}$. By this definition, we will consider p_i to be in E if and only if $b_i = 1$. Conversely, bits b_j set to zero imply that the corresponding models p_j will not be added to E . As a resulting property, $|E| = \sum_{i=1}^N (b_i)$, meaning that the binary string will require at least two bits to be set to one.

It is worth noting that for a size of N models, the total number of possible ensembles E is $2^N - N - 1$ (considering that the ensemble cannot be neither a single model nor the empty set). This leads to an exponential computational cost of $\mathcal{O}(2^N)$ for testing all possible values, which will be unfeasible as the value of N grows due to the combinatorial explosion. For this reason, a heuristic (or metaheuristic) approach is required for solving this optimization problem.

Favorably, having the problem of model selection represented by means of a binary string makes it simple to be optimized using GAs, which lie within the field of biologically-inspired metaheuristics. In particular, our approach adheres to the following genetic operators, which are inspired by natural selection and evolution dynamics:

- Tournament selection, where a small random subset of the population is chosen to compete against each other, and the fittest individual will be allowed to breed.
- Crossover, where a pair of individuals will reproduce by mixing their genetic material (the chromosome, which is given by the binary string described above) producing an offspring of two individuals.
- Mutation, where a small fraction of the offspring’s genetic material is modified at random.

These three operations performed sequentially constitute a generation. Throughout the generations, it is expected that fittest individuals will breed more often, leading to fitter individuals.

What remains to be defined is the fitness function, which provides a measure of how fit an individual is. Since individuals are ensembles, we can define the fitness function as the overall performance of the ensemble; i.e., its accuracy over a validation set using a certain ensemble policy. Invalid individuals (the empty set and ensembles comprising only one model) must have a minimal fitness of 0, so that their chance to breed and establish in the population are negligible. Given this metric, we expect the fitness function to be maximized as generations happen.

4 Evaluation

We have tested the previous proposal atop a previous work published by Baldominos et al. [3] in 2018. In this work, authors carried out neuroevolution to evolve a population of CNNs in order to optimize handwritten digit recognition using the well-known MNIST database. Additionally, the authors describe the use of a niching scheme for preserving diversity in the population, which makes it convenient for building ensembles out of it. To evaluate our proposal, we will perform model selection for building committees of CNNs which outperform single models.

4.1 Background and Experimental Setup

The first decision to make is what to consider the pool P of possible models from which those constituting the ensemble will be selected. A possible option would be to choose the population from the last generation, which is expected to be the best. However, Baldominos et al. [3] describe the use of a “hall-of-fame” where all-time best individuals are stored, and these are fully-trained using a larger number of epochs. For this reason, we find it natural to use the models in this hall-of-fame to define the set P .

Because the hall-of-fame, and therefore our candidate set P , has a total of 20 models, the search space totalizes to $2^{20} = 1,048,576$ possible ensembles. Since evaluating the fitness of an ensemble requires using all models contained in it to predict, computing the fitness may take a few minutes, thus requiring several years to test all possible combinations.

The GA used to perform model selection will involve a population of 100 individuals, with tournaments of size $\tau = 4$. The chromosome length is determined by the number of models in P , which is $|P| = N = 20$. A single-point crossover operation is chosen, which means that the chromosome of both parents will be split in two by selecting a random point, and recombined to create two children, each having genetic material from both ancestors. Finally, a bit-flipping mutation operator will occur with a mutation rate of $\beta = 0.03$, meaning that for each child 3% of the genes (bits) will be selected at random and be flipped.

The process will run for a maximum of 150 generations, but a stop criterion is implemented to halt the process if the best individual does not improve for more than 25 generations. The population from the first generation is randomly

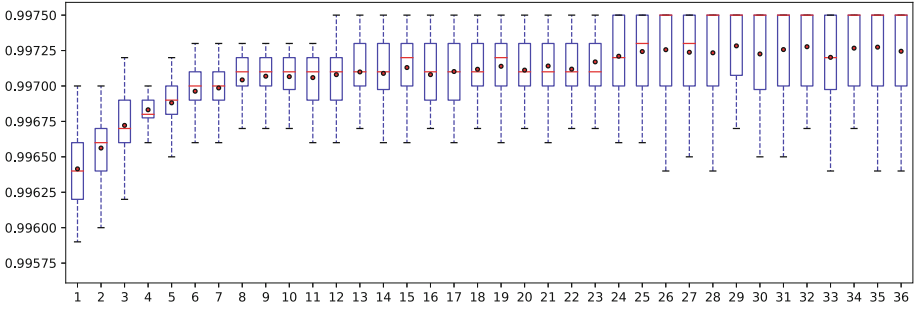


Fig. 2. Evolution of the populations throughout the generations.

generated. Elitism is introduced in order to keep the best individual from each generation in the following generation.

4.2 Results and Discussion

The GA has been running for 36 generations before halting. The evolution of the populations throughout the generations can be seen in Fig. 2, where a boxplot showing distributions of accuracy is shown for each population. In such figure, the horizontal axis represents the generations and the vertical axis represents the accuracy. The boxplot shows the median value as a line within the box and the mean as a dot. For simplicity, outliers are hidden in the figure. It can be seen how there is an improvement during the early generations, until reaching a plateau where mean and median do not vary significantly as generations happen.

The best ensemble found by the GA is composed of seven models, and when operating under a majority-voting policy it provides an accuracy of 99.75% over the MNIST test set used out-of-the-box, without any further preprocessing or data augmentation. This value outperforms the best result attained by Baldominos et al. [3] with a single neuroevolved model (99.63%), and also improves the value reported by Baker et al. [2] when building an ensemble of ten models out of a set of CNNs optimized using reinforcement learning (99.72%). We have found that when the ensemble is built in a greedy manner, ranking models by descending performance and adding them one at a time to the ensemble, then the best committee found also comprises seven models, but only reaches a peak accuracy of 99.72%. To the best of our knowledge, the result reported in this paper is only slightly outperformed by the work of Chang and Chen [6] using batch-normalized maxout network-in-network (99.76%), when considering only works without any kind of preprocessing or data augmentation.

In order to get a better understanding of the recognition mistakes made by the best ensemble found by the GA, we can take a look at the confusion matrix in Fig. 3. As we can see, most errors involve misclassifying a ‘9’ as a ‘4’ (four errors), a ‘5’ as a ‘3’ (three errors) or a ‘7’ as a ‘1’ (two errors). In all other cases, there is at most one instance misclassified or none at all.

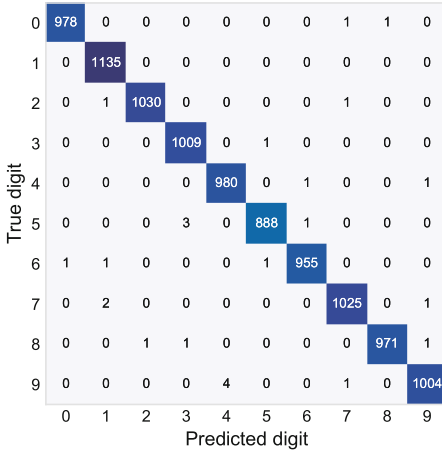


Fig. 3. Confusion matrix for the MNIST dataset with the best ensemble.

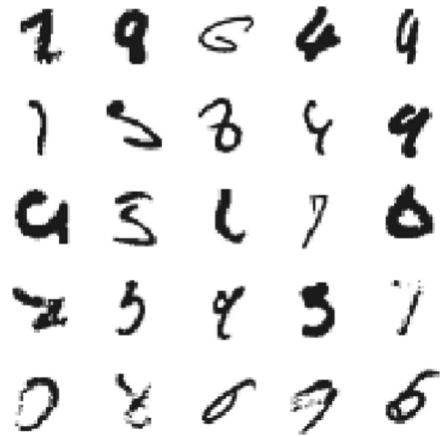


Fig. 4. Exhaustive set of misclassified test instances with the best example.

Additionally, since the MNIST test set comprises 10,000 instances, an accuracy of 99.75% is equivalent to a test error rate of 0.25%, thus resulting in 25 misclassified instances. The whole set of recognition errors performed by the best ensemble are shown in Fig. 4. When observing these digits, it can be seen how they are difficult to distinguish even for a human: some digits seem like a ‘4’, but could be a ‘9’ opened at its top. A similar problem arises for establishing a difference between a ‘5’ and a ‘3’, and to a lesser extent, to other digits. These misclassified instances could be tackled with the inclusion of data augmentation and noise in the training set.

5 Conclusions

CNNs are becoming a widely spread technique for solving a variety of AI problems, with many applications in academia and industry. However, deciding their best architecture and hyperparameters is unfeasible with analytic procedures, and in many cases requires an expensive trial-and-error process. However, in very recent years, neuroevolution (a technique with successful results since the late 1980s) have been applied to the automatic design of deep and CNNs, due to the increase in computing power and advances in learning algorithms.

In this paper we have proposed the application of GAs in order to select models from a population of neuroevolved CNNs. While some papers describe the use of committees (or ensembles) of neural networks, they are not widely used, mostly due to two reasons: (1) the cost of training and running an ensemble grows linearly to the number of models and (b) choosing the most suitable topologies is very complicated, since the different choices grow following a combinatorial explosion. However, in most cases committees have been documented

to outperform individual models, and we have found neuroevolution to be a natural source of candidate models for building ensembles, since the process generates a set of optimized models. Moreover, by implementing certain strategies for preserving genetic diversity, the set of candidate models can be even more interesting, because a certain degree of variability is guaranteed between them.

We have applied this proposal on top of a previous work where neuroevolution was used to optimize models for handwritten digit recognition, tested against the well-known MNIST database. By using GAs to optimize the models conforming the ensembles, we have managed to increase accuracy from 99.63% to 99.75%, which results in a significantly better recognition performance. This improvement is also noticeable, although in a slighter manner, over other common approaches for building the ensembles, such as a greedy “best-first” approach, that has led to committees with an accuracy of at most 99.72%.

Acknowledgement. This research is supported by the Spanish Ministry of Education, Culture and Sport through FPU fellowship with identifier FPU13/03917.

References

1. Akhtyamova, L., Ignatov, A., Cardiff, J.: A Large-scale CNN ensemble for medication safety analysis. In: Frasinca, F., Ittoo, A., Nguyen, L.M., Métais, E. (eds.) NLDB 2017. LNCS, vol. 10260, pp. 247–253. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59569-6_29
2. Baker, B., Gupta, O., Naik, N., Raskar, R.: Designing neural network architectures using reinforcement learning. In: Proceedings of the 5th International Conference on the Learning Repr. (2017)
3. Baldominos, A., Saez, Y., Isasi, P.: Evolutionary convolutional neural networks: an application to handwriting recognition. *Neurocomputing* **283**, 38–52 (2018)
4. Baldominos, A., Saez, Y., Isasi, P.: Evolutionary design of convolutional neural networks for human activity recognition in sensor-rich environments. *Sensors* **18**(4), 1288 (2018)
5. Breiman, L.: Random forests. *Mach. Learn.* **45**(1), 5–32 (2001)
6. Chang, J.R., Chen, Y.S.: Batch-normalized maxout network in network. *arXiv arXiv:1511.02583* (2015)
7. Ciresan, D.C., Meier, U., Gambardella, L.M., Schmidhuber, J.: Convolutional neural network committees for handwritten character classification. In: Proceedings of the 2011 International Conference on Document Analysis and Recognition, pp. 1135–1139 (2011)
8. Davison, J.: DEvol: Automated deep neural network design via genetic programming (2017). <https://github.com/joeddav/devol>. Accessed 01 July 2017
9. Desell, T.: Large scale evolution of convolutional neural networks using volunteer computing. In: Proceedings 2017 Genetic Evolutionary Computation Conference Companion, pp. 127–128 (2017)
10. Fernando, C., et al.: Convolution by evolution: differentiable pattern producing networks. In: Proceedings 2016 Genetic Evolutionary Computation Conference, pp. 109–116 (2016)

11. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. *Mach. Learn.* **63**(1), 3–42 (2006)
12. Hinton, G., Srivastava, N., Swersky, K.: RMSProp: divide the gradient by a running average of its recent magnitude. *Neural networks for machine learning*, Coursera lecture 6e (2012)
13. Kassahun, Y., Sommer, G.: Efficient reinforcement learning through evolutionary acquisition of neural topologies. In: *Proceedings of the 2005 European Symposium on Artificial Neural Networks*, pp. 259–266 (2005)
14. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. *arXiv arXiv:1412.6980* (2014)
15. Koutník, J., Schmidhuber, J., Gomez, F.: Evolving deep unsupervised convolutional networks for vision-based reinforcement learning. In: *Proceedings 2014 Annual Conference on Genetic and Evolutionary Computation*, pp. 541–548 (2014)
16. Loshchilov, I., Hutter, F.: CMA-ES for hyperparameter optimization of deep neural networks. *arXiv abs/1604.07269* (2016)
17. Miikkulainen, R., et al.: Evolving deep neural networks. *arXiv abs/1703.00548* (2017)
18. Ordóñez, F.J., Roggen, D.: Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. *Sensors* **16**(1), 115 (2016)
19. Real, E., et al.: Large-scale evolution of image classifiers. *arXiv abs/1703.01041* (2017)
20. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. *Evol. Comput.* **10**(2), 99–127 (2002)
21. Suganuma, M., Shirakawa, S., Nagao, T.: A genetic programming approach to designing convolutional neural network architectures. In: *Proceedings 2017 Genetic and Evolutionary Computation Conference*, pp. 497–504 (2017)
22. Sultana, J., Rani, M.U., Farquad, M.: An extensive survey on some deep learning applications. In: *Proceedings of the 2018 IADS International Conference on Computing, Communications & Data Engineering* (2018)
23. Verbancsics, P., Harguess, J.: Image classification using generative neuroevolution for deep learning. In: *Proceedings 2015 IEEE Winter Conference on Applications of Computer Vision*, pp. 488–493 (2015)
24. Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E.: Deep learning for computer vision: a brief review. *Comput. Intell. Neurosci.* **2018**, 13 (2018)
25. Xie, L., Yuille, A.: Genetic CNN. *arXiv abs/1703.01513* (2017)
26. Yao, X., Liu, Y.: A new evolutionary system for evolving artificial neural networks. *IEEE Trans. Neural Netw.* **8**(3), 694–713 (1997)
27. Young, S.R., Rose, D.C., Karnowsky, T.P., Lim, S.H., Patton, R.M.: Optimizing deep learning hyper-parameters through an evolutionary algorithm. In: *Proceedings Workshop on Machine Learning in High-Performance Computing Environments* (2015)
28. Zhang, L., Wang, S., Liu, B.: Deep learning for sentiment analysis: a survey. *Interdiscip. Rev. Data Min. Knowl. Discov.* **8**, e1253 (2018)
29. Zhang, Q., Yang, L.T., Chen, Z., Li, P.: A survey on deep learning for big data. *Inf. Fusion* **42**, 146–157 (2018)



Chatbot Theory

A Naïve and Elementary Theory for Dialogue Management

Francisco S. Marcondes^(✉), José João Almeida, and Paulo Novais

Algoritmi Centre, ISLab, University of Minho, Braga, Portugal
id7515@alunos.uminho.pt, {jj,pjon}@di.uminho.pt
<http://algoritmi.uminho.pt/>

Abstract. Due to the increasing interested and use of chatbot, its properties and operation possibilities shall be proper realized matching both safety and security issues as well as present the several uses and compositions that this technology supports. This paper focus is on dialogue management since it is considered the core of a chatbot. The dialogue manager is responsible to, more than to transform an input sentence into an output one, hold the illusion of a human conversation. In this sense, it is presented an inceptive theoretical framework through a formal way for chatbots that can be used as a reference to explore, compose, build and discuss chatbots. The discussion is performed mostly on ELIZA since, due to its historical records, it can be considered an important reference chatbot, nevertheless, the proposed theory is compatible with the most recent technologies such those using machine and deep learning. The paper then presents some sketchy instances in order to explore the support provided by the theory.

Keywords: Chatbot · Formal theory · Dialogue manager
Mechanical dialogue

1 Introduction

There is a growing interest in chatbots (see Fig. 1a) as it uses in the internet increases (it may reach 85% of chat interactions in 2020 [10]). These applications are being used through a wide spectrum and, as an outcome, raising issues as when SIMSIMI starts to threat kidnapping [6].

To allow the use of chatbots in increasingly diverse contexts and also to handle the concerns that arise from this, a theory is needed [13].

This paper has been supported by COMPETE: POCI-01-0145-FEDER-0070 43 and FCT - Fundação para a Ciência e Tecnologia - Project UID/CEC/ 00319/2013.

© Springer Nature Switzerland AG 2018
H. Yin et al. (Eds.): IDEAL 2018, LNCS 11314, pp. 374–384, 2018.
https://doi.org/10.1007/978-3-030-03493-1_40

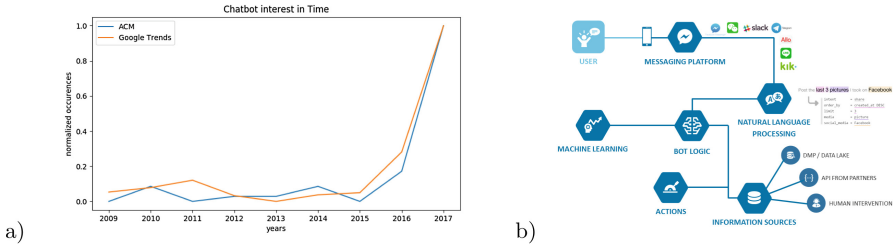


Fig. 1. (a) The chart shows the variation of chatbot interest in time by ACM and Google Trends. The search keyword used on both was *chatbot* and the values were normalized by Min-Max scaling to provide a better comparison. (b) A chatbot reference architecture [7].

Paper Objective. This paper intention is then to present an inceptive chatbot theory as proof of concept on how a theoretical view may help on matters as such. A theory as such provides an underlying framework that can be used when examining mechanical dialogue phenomena and explore design and behaviour decisions when building a chatbot application.

1.1 A Brief Chatbot Review

Dialogue Illusion. A chatbot is a device intended to perform *social interactions* like a human would do [6, 17, 18]. This is the cornerstone for chatbots, otherwise, it may easily become a question and answer, information retrieval, or another system as such. In other words, to be a chatbot an application must create the illusion of an actual dialogue between people [6].

Mechanical Dialogue. Every chatbot as a computer program is reducible to a Turing Machine which implies that it may only perform a mechanical behaviour [1, 17]. Within the dialogue domain, this results in a *mechanical dialogue* conception as a narrower version of scripted dialogues as those used for telemarketing. The art is how to hold the illusion of human conversation through mechanical procedures.

Dialogue Manager Conception. The responsibility to create and sustain the illusion of social interaction is of the *dialogue manager*. Figure 1b present a reference stack for chatbot technology aiding the understanding of how the different components interact; the dialogue manager is exhibited as “Bot Logic”. Then to fulfil its responsibility the dialogue manager articulates all the other components to generate a “proper” output [8]. The word “proper” means an output that nurtures the illusion of a conversation.

Dialogue Manager Operation. Roughly, the behaviour of the dialogue manager can be described as a mapping function $f : I \rightarrow O$ where I stands for input and O for output; both being sets of sentences. The mapping is used to be performed by a deterministic or statical procedures. Roughly, on the deterministic approach, all possible mappings are defined *a priori* by a set of rules resulting in a behavior as *if $i \in I$ then $o \in O$* . On the statistical approach, first, a dataset is turned into a dictionary of words and probability relations among them; when an input is received is then determined the probability of the next word as $f(f(f(f(i) : o_1) : o_2) : \dots) : o_n)$. Deterministic approaches perform a straightforward mapping while statistical approaches a probabilistic one. On the edge, where an input-sentence matches perfect proximity (equals '1.0'), it tends to work as a deterministic chatbot.

Formal Theory. A mathematical theory is a set of rules that determine the production of mathematical models, *e.g.* the number theory [9]. In this sense, the set of rules that settle a chatbot model is its theory and a chatbot is an instance of that model. Even intuitionism being the suitable mathematical paradigm to programming [16], as it is not yet possible to define each element *a priori* and also avoid supposition, generality and infinity it suits better develop the theory through formalism [3] tending to intuitionism as it nears to become an instance as a computer program.

2 The Dialogue Manager Theory

Elementary and Naïve Theory. As this paper intends to present an inceptive mathematical theory, it is kept as simple as possible. The proposed calculus is elementary (does not support variables) and the presentation is naïve (the focus holds on the probe and not in full formalization). Let's then proceed with the presentation.

Chat Phrase Set. The smallest unit of expression in English is a phrase (or an utterance on speaking). A phrase requires neither subject or predicate as long it express something, *e.g.* the interjection "woah!" is a phrase. Therefore a dialogue is a sequence of phrases and phrases assemblages among actors. During a dialogue, each actor says a phrase that answers to a previous one, start a new conversation topic, or both. Within a dialogue, there are also relevant non-verbal markers as the silence that can be associated with many semantics given a context.

The proposed formalization is composed of a set of phrases name P (the definition of the phrase conception is wittingly postponed to each dialogue manager design). Linked to P there is a Φ set of functions over P as $\{\varphi : P \rightarrow P \mid \varphi \in \Phi\}$. The difference between two functions φ and φ' in Φ is given by the mapping rule bounded to each one. The aimed formalism requires another set to hold the ending phrases denoting the end of a conversation, this is called the F set. These lead to the formulation stated in Definition 1.

Definition 1 (Dialogue Manager). A dialogue manager is an extensible 3-tuple $\{P, \Phi, F\}$ where,

P is the set of chat-phrases plus ξ as silence mark.

Φ is the set of attached functions¹ over a set as $\varphi \bowtie P$ and of the commonly used set operators.

$F \subseteq P$ is a set of chat-phrases expressing the end of dialogue (followed by \odot).

The symbol \cdot denotes the dialogue start and symbol \odot the dialogue end.

Mechanical Dialogue Definition Then, given a dialogue manager $B = \{P, \Phi, F\}$ a mechanical dialogue is a sequence

$$\cdot p \in P \quad \overset{(\varphi \bowtie P) \in \Phi}{\vdash} \quad p' \in P \quad \overset{(\varphi' \bowtie P) \in \Phi}{\vdash} \quad \dots \quad \overset{(\varphi'' \bowtie P) \in \Phi}{\vdash} \quad p'' \in P \quad \overset{(\varphi'' \bowtie F) \in \Phi}{\vdash} \quad f \in F \odot$$

Whenever the chosen paradigm, the dialogue manager definition turns out to be something like $\cdot Hi \vdash Hello \vdash \dots \vdash Bye \vdash See'ya \odot$ (refer to Example 1 for a full presentation). This example suites the mechanical dialogue definition as φ relates to a dialogue start protocol that then turns in to φ' as “common dialogue” and then φ'' relates to a dialogue end protocol over the F set.

Definition Extensions. Definition 1 is said to be extensible as it supports the construction of other subsets of P than just F . It would be a S set to handle the dialogue start protocol which would result in the function $(\varphi \bowtie S) \in \Phi$ within the mechanical dialogue definition.

Dialogue Manager Builder. Since $B = \{P, \Phi, F\}$ and $F \subseteq P$; therefore $B = \{P, \Phi\}$ (the F set is required to trace the end of the dialogue). So, given a set-builder in the form $X = \{p \in P | \Xi(p)\}$, that results into a subset X with elements on P that $\Xi(p)$ is true; then $F = \{p \in P | p \text{ is a finishing phrase}\}$. Any other subset can be defined as such. Each subset must have a function bound to it, nevertheless, a function as $\varphi \bowtie F \subset P$ is also a function $\varphi \bowtie P$. This is used to trace special conditions and perform separation of concerns as intended by [5, 12].

A function implies the existence of a Cartesian product on P , *i.e.* $(\varphi : P \rightarrow P) = P \times P$ producing an implicit Δ set composed by binary relations in the form $(p_a, p_b) \in \Delta$. Suppose $P = \{p_a, p_b, p_c\}$; it requires the existence of an implicit $P \times P$ set called Δ to a function operates on it. Let $\Delta = [(p_a[p_b, p_c]), (p_b[p_a, p_c]), (p_c[p_a, p_b])]$, if $\delta = (p_a[p_b, p_c]) \in \Delta$ is called, shall it result in p_b, p_c or $[p_b, p_c]$? An answer to it depends on the chatbot model under design and therefore such is also kept wittingly undefined avoiding to fast to a particular approach.

¹ In this paper a function $f(a) : b$ is denoted by $a \vdash b$.

Dialogue Manager Operations. In addition to the functions, Φ set is also formed by the commonly used set operators. Suppose a dialogue manager A and a dialogue manager B . If they are disjoint ($A \neq B$), i.e. if $A \cap B = \emptyset$ then the formation of a joint dialogue manager is quite straightforward as $K_{A \cup B} = \{P_{(P' \in A) \cup (P'' \in B)}, \Phi_{(\Phi' \bowtie A) \cup (\Phi'' \bowtie B)}, F\}$, it however may need some refactoring on the F set and attached functions. If there are overlapping data or rules a more profound refactoring may be needed, anyway as the whole proposal is formal a tool may aid the conflict resolution. Otherwise, if $P' \in A = P'' \in B$ ($\forall x((x \in P') \leftrightarrow (x \in P''))$), the two dialogue managers can be considered similar ($A \equiv B$). If the same happens on Φ as $\forall \varphi((\varphi' \bowtie P' \in A) = (\varphi'' \bowtie P'' \in B))$ then A and B are the same chatbot.

Dialogue Turns. Natural deduction structure is a suitable way to handle dialogue “turns” in the form (strict) $\frac{\text{Premises}}{\text{Output-Phrase}}$ Input-Phrase, for instance $\frac{P \quad \varphi \bowtie P \quad \Gamma}{b}$ a (Γ denotes the preceding turns). Since it is possible to provide more than one function to P, the function itself is a premiss. It can be also in the form (rough) $\frac{\text{Input-Phrase} \in P}{\text{Output-Phrase} \in P} \varphi$. See Example 1.

Example 1 (Dialogue Manager).

Let $B = \{P, \Phi, F, S\}$

P is a set of any phrases
 $\Phi = \{\varphi \bowtie S, \varphi' \bowtie P, \varphi'' \bowtie F, \dots\}$
 $F = \{Bye, See'ya, \dots\}$
 $S = \{Hi, Hello, \dots\}$

$$\frac{S = \{Hi, Hello, \dots\} \quad \varphi \bowtie S}{\text{Hello}} \text{Hi} \dots$$

$$\frac{P \quad \varphi' \bowtie P \quad \Gamma}{y} \dots$$

$$\frac{F = \{Bye, See'ya, \dots\} \quad \varphi'' \bowtie F \quad \Gamma}{\text{See'ya} \odot} \text{Bye}$$

All this is valid both to deterministic and statistical approaches. Within the deterministic approach, the use of the presented formalism is straightforward. In the statistical approaches, the sets can be established by annotations in the data set; the Φ set is defined by the statistical method being used. The use of several functions within the dialogue manager definition is an illustration, however, most of the current chatbots use just one function.

3 Dialogue Manager Framework

In addition to the theory, the composition of a dialogue manager requires five concerns to be handled, as presented in Table 1, ranging from operation to behaviour related issues (an operation is a behaviour declaration and behaviour is an operation realization [14]).

ELIZA, for instance, is a deterministic (paradigm) boolean (realization) chatbot modelled as an SWM (strategy) pattern-template (approach) with input priority defined by Rogerian psychology (rule-set) and output priority by a randomized queue (rule-set). Therefore, once the chatbot concerns are classified, it is then possible to think about variations and compositions to be performed.

Table 1. Dialogue Manager Framework.

Concern	Instances	Affair
Paradigm	Deterministic, Statistical, <i>etc.</i>	Operation
Realization	Boolean, Fuzzy, Markov, Deep Learning <i>etc.</i>	↓
Strategy	Direct Map, SWM, Seq2Seq, <i>etc.</i>	
Approach	Phrase-Reply, Pattern-Template, Frame-slot, <i>etc.</i>	↑
Rule-set	Rogierian, Eckmannian, queue, <i>etc.</i>	Behaviour

The proposed theory shall then provide an underlying support for the treatment and discussions of all those concerns. In other words, a full chatbot must address all those concerns grounded in a theory that supports it. In this sense, the Dialogue Manager Framework is the theory as presented in Definition 1 and the concerns as exhibited in Table 1.

3.1 Boolean Dialogue Manager Realization Instance

The Boolean operators *cf.* [15] can be described as functions on Φ and stated as in Definition 2 being an example of deterministic dialogue manager.

Definition 2 (Boolean Dialogue Manager). *A boolean dialogue manager is 3-tuple $\{P, \Phi, F\}$ as in Definition 1 and*

- $p \vee p'$ is an expectation of p or p' due to an input.*
- $p \wedge p'$ is a sentence composed by the phrases p and p' .*
- $p \vdash p'$ is a function f that produces p' from p .*
- $\neg p$ mean that there is no p which is meaningless the $\neg P \vdash \perp$*
- \perp is an absurd statement within a chat context that breaks the illusion of dialogue.*

Boolean Operators. The implicative operator can be used in the same sense of specific defined functions in Φ and results as well in a sequence on P as $p \vdash p' \vdash \dots \vdash p''$. The alternative operator can be used when choosing between phrases as $\frac{((p \vdash p') \vee (\xi \vdash p''))}{p'}$ p or $\frac{((p \vdash p') \vee (\xi \vdash p''))}{p''}$ ξ . The conjunctive operator can be used to handle sentences composition and splitting of sentences to handle information anticipation and dialogue initiative turns $\frac{P \quad \varphi \quad \Gamma}{(p'' \wedge p''')} (p \wedge p')$.

However, the negative operator may result into a nonsense output as presented in the Lemma 1 and as a weak claim, negation shall be avoided within boolean dialogue managers since it leads to a [pragmatic] contradiction.

Lemma 1 (The negation chat-implication leads to an absurd). *Let $P = \{p_a, p_b, \dots, p_n\}$
 Since $\frac{p_a \vdash p_b}{p_b}$, Thus $\frac{p_a \not\vdash p_b}{\bar{P}}$, Then $S = \{p_a, \dots, p_n\}$ being an absurd chat-sentence to output. $\therefore (p_a \not\vdash p_b) \rightarrow \perp$*

It shall be highlighted that these operators refer to the logical relations between phrases and not to their semantics. Let $p =$ ‘the snow is white’ then $p \wedge p'$ is TRUE if both are $\in P$, are called, etc. As well $\neg p$ does not means that ‘the snow is not white’ but that there is no such statement.

3.2 Markovian Dialogue Manager Realization Instance

The Markov chains cf. [15] can be described as functions on Φ and stated as in Definition 3 being an example of statistical dialogue manager.

Definition 3 (Markovian Dialogue Manager). *A markovian dialogue manager is 3-tuple $\{P, \Phi, F\}$ as in Definition 1 where*

*Φ includes the probability function as $\varphi_\delta(P_{t+1} = p | P_t = p_t)$
 Δ is the implicit set that holds the probabilities of $\varphi_\delta \bowtie P$*

Markov Example. According to the Markov Chain formalism, the function denoted by φ_δ is a transition probability function and is used to calculate the phrase p in the time $t + 1$ given a sequence of p_t phrases. As an example, let

$$\Delta = \begin{matrix} & \begin{matrix} Hi & Hello & Bye & See'ya & \dots \end{matrix} \\ \begin{matrix} Hi \\ Hello \\ Bye \\ See'ya \end{matrix} & \begin{pmatrix} 0.27 & 0.29 & 0.12 & 0.10 & \epsilon \\ 0.28 & 0.26 & 0.14 & 0.12 & \epsilon \\ 0.13 & 0.11 & 0.26 & 0.28 & \epsilon \\ 0.10 & 0.11 & 0.29 & 0.25 & \epsilon \end{pmatrix} \end{matrix} \text{ then } \varphi_\delta(P_1 = Hello | P_0 =$$

$Hi)$ and $\varphi_\delta(P_{t+n+1} = Bye | P_{t+n} = \dots)$, $\varphi_\delta(P_{t+n+2} = See'ya | P_{t+n+1} = Bye)$. As *Bye* and *See'ya* and in the set F , it can be attached the end symbol \odot in the time P_{t+n+2} .

3.3 ELIZA and MARK

Historical Chatbots and Current Tech. Two historical representative chatbots are ELIZA and Mark v Shaney (MARK). They are said to be representative since ELIZA inspired approaches are currently being used in top of hedge chatbots as MITSUKU, the 2017 Loebner Prize winner. MARK inspired approaches are also currently being used as by the seq2seq machine learning chatbots.

ELIZA is a deterministic chatbot built to simulate a Rogerian psychotherapist [18] which was also considered to be actually used as a first line of psychological support [2]. MARK is a statistical chatbot built to discuss in the Usenet newsgroups [4], it took other users positions and through Markov Chains generates a reply that was able to fool several users.

Behaviour Description. Roughly, the behaviour of ELIZA is based on SWM (Split-Weight-Map), consisting in split an input-sentence into phrases according to punctuation marks. Match them to a pattern weighting them and by polling the phrase with the highest weight raises its template.

Also roughly, The behaviour of MARK is given a data set, it calculates the sequence probability of each word (phrase here is realized as word). Then, choosing the first word as the time P_0 it creates a sequence p_0, \dots, p_n to time P_n (each “time” is a word).

3.4 Dialogue Manager Sketches

The Framework Rationale. It can be realized that ELIZA uses the SWM strategy with the pattern-template approach and Rogerian rule-set. These, however, are design decisions of ELIZA bot. There are several variations that can be performed generating several chatbots.

It is possible as an instance to change ELIZA’s approach from pattern-template to phrase-reply yielding an elementary chatbot. It is also possible to change the deterministic mapping procedure of SWM to a probabilistic one as of MARK. As well it is possible to build variations on MARK to handle phrases (not words) or perform weighting as ELIZA does to guide the probabilistic choice.

This suggests that allied to the proposed theory, the framework helps in the exploration of dialogue management conceptions. To a further inception it is present some sketches to be considered. To an easier organization of ideas, the discussion will be focused on ELIZA-like dialogue managers.

Dialogue Protocols. There is an aforementioned statement for “dialogue protocols” to handle special circumstances such as the dialogue start, end, *etc.* A deterministic way to perform such scripts is by a state machine based approach as, let $\varphi \bowtie S$ produce the implicit $\Delta' =$

$$\Delta' = \begin{matrix} & s_0 & s_1 & s_2 & p \in P \\ \begin{matrix} s_0 \\ s_1 \\ s_2 \end{matrix} & \left(\begin{array}{ccc|c} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array} \right) \end{matrix}$$

that acts as an incidence matrix, and then resulting in the chat sequence $\Gamma \vdash s_0 \vdash s_1 \vdash s_2 \vdash p \vdash \Gamma$.

Emotion Weighting. In addition, the ELIZA Rogerian rule-set can be replaced by an Eckmannian one. Roughly, Eckmann’s model of basic emotions suggests that human emotions range between anger, disgust, fear, happiness, sadness, and surprise as positive or negative; it is then possible to classify an emotional value of some keywords creating a word-emotion lexicon as the one proposed in [11]. For an instance, Rogerian-based set can be something like $P_r = \{(\text{‘sorry’}, 0), (\text{‘remember’}, 5), (\text{‘dream’}, 3), (\text{‘my’}, 3), (\text{‘name’}, 15), \dots\}$ becoming $P_e = \{(\text{‘sorry’}, -1), (\text{‘remember’}, \text{neutral}), (\text{‘dream’}, \text{neutral}), (\text{‘my’}, -1), (\text{‘name’}, \text{neutral}), \dots\}$.

Blended Weighting. In addition to substitute a particular strategy it is also possible to perform a mixed approach. Let P_r be a Rogerian set and P_e an Eckmann set, and a resulting $P = P_r \cup P_e$ becomes $P_{r \cup e} = \{(\text{'sorry'}, (0, -1)), (\text{'remember'}, (5, \text{neutral})), (\text{'dream'}, (3, \text{neutral})), (\text{'my'}, (3, -1)), (\text{'name'}, (15, \text{neutral})), \dots\}$, assuming that P is accordingly fashioned, it is possible to perform a $\vdash_{r \wedge e}$ function returning $P_r = ((\text{'Is it yours?'}, 1), (\text{'Why bother saying that is yours?'}, 0))$, $P_{e_1} = (\text{neutral}, [(\text{'Is it yours?'}, 1), (\text{'Why bother saying that is yours?'}, 0)])$ and $P_{e_2} = (\text{neutral}, [(\text{'My name is Bot'}, 1)])$. Then $(P_{e_1} \cup P_{e_2}) \cap P_r$ may be the result P set.

Composed Input Weighting. It is then possible to perform a mixed strategy as $P = P_r \cup P_e$ becoming $P_{r \cup e} = \{(\text{'sorry'}, (0, -1)), (\text{'remember'}, (5, \text{neutral})), (\text{'dream'}, (3, \text{neutral})), (\text{'my'}, (3, -1)), (\text{'name'}, (15, \text{neutral})), \dots\}$ providing a $\vdash_{r \wedge e}$ implication. It is also possible to define alternative strategy in the form of $\vdash_{r \vee e}$. This means that to each interaction the bot shall respond according to Rogerian or Eckmannian.

Safety Awareness. A possible application of such in addition to the dialogue protocols is to perform safety awareness monitoring *fear* if a fear phrase is detected then Eckmann strategy arises otherwise Rogerian strategy is used. Then, being $Z \subset P$ a set of fear keywords as $Z = (\text{afraid}, \text{affliction}, \text{adverse}, \dots)$ as *if* $\exists z_i$ in p_j then z_i script start. Such script may lead to a decision tree modelled as a *dialogue protocol* to establish the possibility of risk of hazard. An example for fire awareness dialogue is presented in Example 2. Note that a *ad hoc* dialogue *pragmatics* is being performed.

Example 2 (Fire Alert Chat Example). Let Z a set of fear keywords as $Z = (\text{afraid}, \text{affliction}, \dots)$

$$\frac{\Gamma \vdash_r \Gamma}{\text{I'm afraid about my oven } \vdash_e \text{ Why do you say so?}}$$

$$\vdots$$

$$\frac{\Gamma, \text{Fire is starting... } \vdash_e \text{ Calling the fire brigade.}}$$

\vdots . This may continue with fire-fight specific advices

4 Results and Discussion

This paper starts from the assumption that the dialogue manager is the component responsible for conducting the mechanical dialogue in order to provide a proper social interaction by creating and sustaining the illusion of an actual human conversation. In this sense, the dialogue manager is considered the core of the chatbots that articulates the other components to accomplish its task (refer to Fig. 1b). It then presents a partially defined yet sufficient dialogue management theory.

The proposal is helpful to handle the scientific unsettling provoked by the several terminologies and behaviour used in different chatbot IDEs. Reducing each IDE to a theory as proposed in this paper leads to a common ground allowing discussions and comparisons among them.

Also, as it is based on formal paradigm this paper theory allows a more flexible exploration of chatbot concepts and issues than those that would be performable through IDEs. It likewise provides the choice and synthesis possibility of chatbots but keeping the relationship with computation.

Another outcome of this paper for chatbots is the possibility of a development process. Artificial intelligence systems commonly lack the support of software processes when using “usual” approaches; roughly the specification turns to be quite simple and most of the complexity raises during implementation. The use of the proposed theory as a chatbot framework allow a proper setting for separation of concerns and project life cycles.

Mechanical dialogues and dialogue management are related subjects since the former performs a realization of the prior. As an application building, they refer to different concerns; during analysis what is being analysed is the intended or convenient *mechanical dialogue* to be realized by the *dialogue manager* during design. As a research theme, the first is of interest mainly by linguistics and cognitive scientists while the second being primarily technological artefact concerns to informatics related people. Nevertheless, the presented theory is able to provide support to both of them.

As related works comparison, performing a brief survey on the theme, the key-word “chatbot” in ACM digital library returns 110 entries (the dispersion in time is depicted in Fig. 1a). The 10% of the most relevant papers (according to ACM criteria) are centred in two main interests: dialogue improvements, development process and frameworks (6 papers); and applications, human-computer interfaces and post-human concerns (5 papers). The presented theory fits in neither of those groups as it is a theory and both groups are mainly concerned with implementations, in this sense, it may be placed on the meta-level as it provides underlying support for both of them.

5 Conclusion

This paper presented an inceptive formal theory for dialogue management that was able to support the discussion on how the chatbot issues can be approached both when examining mechanical dialogues and exploring design decisions; leading to useful insights. This suggests that the theory succeeded at the proof of concept level. In further works, this proposal requires relevant extensions to properly handle a fully developed chatbot.

References

1. Bertalanffy, L.: *General System Theory: Foundations, Development, Applications*. G. Braziller, Brooklyn (1968)
2. Colby, K., Watt, J., Gilbert, J.: A computer method of psychotherapy: preliminary communication. *J. Nerv. Ment. Dis.* **142**(2), 148–152 (1966)
3. da Costa, N.: *Introdução aos Fundamentos da Matemática*. HUCITEC (1992)
4. Dewdney, A.K.: Computer recreations. *Sci. Am.* **260**(6), 122–125 (1989)
5. Dijkstra, E.: On the role of scientific thought. In: Dijkstra, E. (ed.) *Selected Writings on Computing: A personal Perspective*. Texts and Monographs in Computer Science, pp. 60–66. Springer, New York (1982). https://doi.org/10.1007/978-1-4612-5695-3_12
6. Ferrara, E., Varol, O., Davis, C., et al.: The rise of social bots. *CACM* **59**(7), 96–104 (2016)
7. Fourault, S.: *The ultimate guide to designing a chatbot tech stack* (2017)
8. Jurafsky, D., Martin, J., Norvig, P., Russell, S.: *Speech and Language Processing*. Pearson, London (2014)
9. Kleene, S.: *Introduction to Metamathematics*. Bibliotheca Mathematica. Wolters-Noordhoff, Alphen aan den Rijn (1952)
10. Levy, H.: Gartner predicts a virtual world of exponential change (2016). Accessed June 2018
11. Mohammad, S., Turney, P.: Emotions evoked by common words and phrases: using mechanical turk to create an emotion lexicon. In: *Proceedings of the CAAGET. Association for Computational Linguistics* (2010)
12. Newell, A.: The knowledge level. *Artif. Intell.* **18**(1), 87–127 (1982)
13. Parnas, D.: The real risks of artificial intelligence. *Commun. ACM* **60**(10), 27–31 (2017)
14. Rumbaugh, J., Jacobson, I., Booch, G.: *Unified Modeling Language Reference Manual*. Pearson, London (2004)
15. Sedgewick, R., Wayne, K.: *Algorithms*. Pearson Education, London (2014)
16. Sørensen, M., Urzyczyn, P.: *Lectures on the Curry-Howard Isomorphism*. Elsevier, Amsterdam (2006)
17. Turing, A.: Computing machinery and intelligence. *Mind* **59**, 433–460 (1950)
18. Weizenbaum, J.: Eliza—a computer program for the study of natural language communication between man and machine. *CACM* **9**(1), 36–45 (1966)



An Adaptive Anomaly Detection Algorithm for Periodic Data Streams

Zirije Hasani¹(✉), Boro Jakimovski², Goran Velinov²(✉),
and Margita Kon-Popovska²

¹ Faculty of Computer Science, University “Ukshin Hoti”, Prizren, Kosovo
zirije.hasani@uni-prizren.com

² FSCE, Ss. Cyril and Methodius University, Skopje, Macedonia
{boro.jakimovski,goran.velinov,margita.kon-popovska}@finki.ukim.mk

Abstract. Real-time anomaly detection of massive data streams is an important research topic nowadays due to the fact that a lot of data is generated in continuous temporal processes. Holt-Winters (HW) and Taylor’s Double Holt-Winters (TDHW) forecasting models are used to predict the normal behavior of the periodic streams, and to detect anomalies when the deviations of observed and predicted values exceeded some predefined measures. In this work, we propose an enhancement of this approach. We implement the Genetic Algorithm (GA) to periodically optimize HW and TDHW smoothing parameters in addition to the two sliding windows parameters that improve Hyndman’s MASE measure of deviation, and value of the threshold parameter that defines no anomaly confidence interval. We also propose a new optimization function based on the input training datasets with the annotated anomaly intervals, in order to detect the right anomalies and minimize the number of false ones. The proposed method is evaluated on the known anomaly detection benchmarks NUMENTA and Yahoo datasets with annotated anomalies and real log data generated by the National education information system (NEIS) (<http://ednevnik.edu.mk/>) in Macedonia.

Keywords: Anomaly detection · Periodic time series
Holt winters algorithm · Genetic algorithm · MASE

1 Introduction

Anomaly detection in real-time massive data streams (practically infinite flow of data, pouring in as time goes, each piece of data having its own time stamp) is one of the important research topics nowadays due to the fact that the most of the world data generation is a continuous temporal process. Many sophisticated and highly effective anomaly detection methods exist that run in batch mode, where the data is collected and processed after the occurrence. However, identifying anomalies long after they happened isn’t our primary goal. On the contrary, real-time data processing, requests continual input, time-critical manner processing, and instant output (e.g. alarm) if anomaly happened. Instead of searching

for the unknown anomalies we can, in advance, model a normal behavior of the data stream and compare it to the observed one. Consequently, predicting the values of a stream one-time step ahead are used, the deviation between the predicted values and the observed values are measured, and a decision mechanism, if an observed value exceeds normal behavior, is established. Yet other questions arise. The real-time streams are infinite, can have a high rate of data appearance in time unite (high volume, high velocity) and can evolve over time. Thus the development of the model of normal behavior must adapt to this challenges to maintain detection accuracy: be iterative, use only a part of the stream (even before it is permanently stored), and be implemented as a positive feedback in the learning process (e.g. repeated anomalies labeling in the supervised process). Due to the need of the real-time detection process, detection algorithms have to be robust, with low processing time (low complexity), even at the cost of the accuracy. Currently, the most intensively developed anomaly detection methods that consider underlined challenges are based on machine learning, neural networks, predictive and statistical time series forecasting models.

In this paper, we are interested in anomaly detection of real data streams that have seasonal patterns. There is a number of studies in this area. The most adequate and often used models are Moving Average (MA) [20], the AutoRegressive Moving Average (ARMA) and AutoRegressive Integrated Moving Average (ARIMA) [20], exponential smoothing algorithms HW [5] and TDHW [11], Hierarchical Temporal Memory (HTM) [18] algorithm and sliding windows [8, 13]. However, our work brings several benefits:

- We propose an enhancement of the additive HW and TDHW algorithms that answers the stated challenges. The algorithm is implemented as a positive feedback optimization with a periodic adaptation of the algorithm parameters.
- Starting with ideas of numerous papers [9–13, 16, 17, 19], we use the GA optimization process, to optimize α , β , γ , ω , the HW and TDHW smoothing parameters, where we added optimization of the three new parameters k , n and δ .
- Improvement is made in the new definition of the optimization function based on the input training datasets with the annotated anomaly intervals, enhanced Hyndman's MASE [14] definition where k and n define the two sliding windows intervals, and δ is the threshold parameter.
- The positive feedback learning process is achieved if the anomalies detected in the next time frame, by the proposed detection engine based on the computed optimal parameters from the annotated anomalies of previous one, are verified/acknowledged by human and reused for parameter optimization.
- The results of the experiments performed on the sets of synthetic and real data periodic streams show that our proposed HW algorithm, with GA optimized parameters and with improved MASE, outperforms the other algorithms.

The data used for experiments are known anomaly detection benchmarks NUMENTA [25] and Yahoo [24] datasets with annotated anomalies and our real log data from the Macedonian national education system e-dnevnik¹.

The rest of this paper has the following structure: in the second section is related work; in the third section proposed a model for real-time data streams anomaly detection is described, in the fourth section are the experimental results; and the last section contains conclusions and further work.

2 Related Work

Autoregressive (AR) and Moving Average (MA) forecasting models have been in existence since the early 1900's. Exponential Smoothing Methods, as a forecasting tool, are introduced in the 1950's. Detailed history, statistical theory and classification depending on the stream (time series) characteristics can be found in [1]. Anomaly detection methodologies are classified in a broad field of multiple research areas ranging from clustering, classification, nearest neighbor, statistical, information theory based and other [2,3].

In this work, we are reviewing papers dealing with Holt-Winters and Taylor's Double Seasonality Holt-Winters forecast modeling of normal data streams behavior. Papers are grouped in studies where HW and TDHW models are used for anomaly detection and their model parameters calculated by exponential formula or decided experimentally [4,5], studies that deal with optimization of the model parameters for the best fitted forecast [9–12], parameter optimization are done using classical optimization methods (e.g. using excel solver) [7] or different metaheuristic algorithms as GA [9,10], Particle Swarm (PS) [10], Artificial Bee Colony algorithm (BEE) [11], Hill climber (HC) and Simulated Annealing (SA) [10], etc.

Brutlag [4] for the first time in 2000 year, used a model based on HW forecasting. He integrated it into the Cricket/RRDtool open source monitoring tools to detect automatically, in the real-time, aberrant behavior of the WebTV services streams. He proposed usage of the exponential formulas for calculation of the smoothing parameters given with Eq. 7 see below. The anomaly is detected if the new observed data stream value y_t falls outside the interval, determined by the measure of deviation d_t for each time point in the seasonal cycle. Deviation d_t is a weighted average of absolute deviation, updated via exponential smoothing (calculated with the same parameter γ as sessional factor in HW). While perhaps not optimal, this solution was shown as flexible, efficient, and effective tool for automatic detection. Authors in [5] implement the same idea in multiplicative HW forecasting model, as a part of a test platform that collects real IP flow, based on open source software Nfsen/RRDtool. Calculation of the parameters was as in [4]. They used Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) as suitable to compare different forecasting methods and Mean Absolute Percentage Error (MAPE), to compare how a forecasting method suits forecasting different time series. The most authors emphases the need of close

¹ <http://ednevnik.edu.mk/>.

examination of the stream behavior before choosing the forecast model: trend existence, characterization of single/ multiple seasons, threshold determination concerning the importance of a number of correct and false detections, and a number of detected anomalies in time unit to signal an alarm.

Optimization of parameters in forecasting model is dating back in 1996 [9]. GA optimization is applied to determine HW smoothing parameters α , β , γ , including variable s , a seasonality interval, and corresponding start-up values for level, trend and seasonality, by minimizing the evaluation function forecasting Mean Square Error (MSE). As the forecasting task presented in this paper did not require a great precision for the parameters and the start-up values, a binary GA (not a real-valued one) is used. Authors underline the great applicability of GA in such type of prediction tasks, especially when a large number of parameters is required.

In some of the works, authors used classical non-linear optimization methods with constrained values of variables, to optimize HW parameters. In [7] authors used the MS Excel Nonlinear Solver, an spreadsheet-based non-linear optimizer, to find the values of the smoothing parameters, together with initial forecast that minimize a measure of forecast error MAD or MSE. A detailed description is given to avoid problems reported by several other authors.

In Shahin [11] authors improved prediction accuracy MSE by employing Artificial Bee Colony algorithm to optimize smoothing parameters of the multiplicative multi sessional HW forecasting model. Cloud workload with multi-seasonal cycle's data stream is foretasted to scale in advance computational resources. Performance of the proposed algorithm has been evaluated with double and triple exponential smoothing methods using MAPE and RMSE.

In [10] authors optimize α , β , γ , δ , smoothing parameters, ϕ damped parameter and λ adjustment for the first order autocorrelation error, of the multiplicative double seasonality and additive damped trend forecast HW. They compare results of minimization of the sum of squared errors equation (SSE) by several meta-heuristic methods: local improved procedure HC and SA, Evolutionary Algorithms (EA), GA, PS. Optimization is implemented in MATLAB for Portuguese three months electricity demand stream of data. The conclusion is that the values obtained for the forecasting equation 's parameters using different meta-heuristic algorithms were similar as well as the post-sample forecasting performance which suggests that HC algorithm for its simplicity is a good solution.

In [12] authors use PS meta-heuristic minimizing the Residual Standard Error (RSE), Sum of Squared Errors (SSE), Mean of Squared Errors (MSE) or Mean Absolute Deviation (MAD) to determine optimal smoothing parameters of the additive Holt model. The direction of the exchange rate and the actual exchange rate values for the Dollar-Peso and Euro-Peso is accurately foretasted.

In [13] work is interesting due to proposed ideas of optimization of the sliding time windows that defines set of time legs used to build a various forecasting methods and also define the number of the model inputs, using the GA and EA with real-valued representative.

(false negative). The result of the verification/acknowledgment stage is then used again in the second stage for further optimization of the anomaly detection parameters.

In rest of this section, we present the improved algorithm for anomaly detection of real data streams with sessional patterns, based on well-known HW and TDHW [5, 11] additive forecasting models. The first improvement is done by modification of the Mean Absolute Scaled Error (MASE) [14], and the second one by optimization of the model parameters.

3.1 Algorithms for Anomaly Detection and MASE Modification

Additive HW trend forecast prediction \hat{y}_{t+1} is defined iteratively (1) by three components, level l_t , trend b_t and seasonality s_t using restricted real smoothing constants $0 \leq \alpha, \beta, \gamma \leq 1$:

$$\begin{aligned}
 &\text{forecast equation: } \hat{y}_{t+1} = l_t + b_t + s_{t-m+1} \\
 &\text{level : } l_t = \alpha(y_t - s_{t-m}) + (1 - \alpha)(l_{t-1}b_{t-1}) \\
 &\quad \text{trend : } b_t = \beta(l_t - l_{t-1}) + (1 - \beta)b_{t-1} \\
 &\quad \text{seasonality : } s_t = \gamma(y_t - l_t) + (1 - \gamma)s_{t-m}
 \end{aligned} \tag{1}$$

where m is the periodicity of the one whole seasonal cycle, i.e. the number of time steps of one season. Good initial values l_0, b_0 and s_0 (2) can be achieved having y_t streaming data of two full sessional cycles $2m$.

$$\begin{aligned}
 &\text{initial level component: } l_0 = \frac{y_1 + y_2 + \dots + y_m}{m} \\
 &\text{initial trend component: } b_0 = \frac{\sum_{t=m+1}^{2m} y_t - \sum_{t=1}^m y_t}{m^2} \\
 &\text{initial seasonal component: } s_i = y_i - l_0, i = 1, 2, \dots, m.
 \end{aligned} \tag{2}$$

Additive TDHW, trend forecast prediction \hat{y}_{t+1} (3) is defined iteratively by four components: level l_t , trend b_t , m_1 seasonality and m_2 seasonality, using restricted real smoothing constants $0 \leq \alpha, \beta, \gamma, \omega \leq 1$:

$$\begin{aligned}
 &\text{forecast equation: } \hat{y}_{t+1} = l_t + b_t + D_t + W_t \\
 &\text{level : } l_t = \alpha(y_t - D_{t-m_1} - W_{t-m_2}) + (1 - \alpha)(l_{t-1} + b_{t-1}) \\
 &\quad \text{trend : } b_t = \beta(l_t - l_{t-1}) + (1 - \beta)b_{t-1} \\
 &\text{daily seasonality: } D_t = \gamma(y_t - l_t - W_{t-m_2}) + (1 - \gamma)D_{t-m_1} \\
 &\text{weekly seasonality: } W_t = \omega(y_t - l_t - D_{t-m_1}) + (1 - \omega)W_{t-m_2}
 \end{aligned} \tag{3}$$

For example, if the stream values y_t are observed every minute a daily cycle $m_1 = 24 * 60 = 1440$ and a weekly cycle $m_2 = 24 * 60 * 7 = 10080$ [6]. Possible initial values are:

$$\begin{aligned}
 &l_0 = y_1; b_0 = 0 \\
 &D_{0,1} = D_{0,2} = \dots = D_{0,m_1} = 0; W_{0,1} = W_{0,2} = \dots = W_{0,m_2} = 0
 \end{aligned}$$

Measurement of the forecast accuracy (by using MASE), defined by Hyndeman [14], is calculated as follows:

$$q_t = \frac{|y_t - \hat{y}_t|}{\frac{1}{l-1} \sum_{i=2}^l |y_i - y_{i-1}|} \quad MASE = \frac{1}{t} \sum_{i=1}^t q_i \quad (4)$$

Where l is a number of values in the training stream. In the anomaly detection models based on HW or TDHW models [4–6], if $MASE > \delta$, where δ is a pre-defined threshold, the new arrived stream data y_t is determined as an anomaly.

We propose an adoption of the MASE definition (5) by adding two window parameters k and n , to the current iterative processes (1) and (3) with smoothing parameters α, β, γ and ω . For the HW forecast, MASE depends on parameters $\alpha, \beta, \gamma, \delta, k, n$ and for TDHW, MASE depends on parameters $\alpha, \beta, \gamma, \omega, \delta, k, n$.

$$\begin{aligned} q_{t_{(\alpha, \beta, \gamma, \delta, k)}} &= \frac{(|Y_t - \hat{Y}_t|)}{\frac{1}{k} (\sum_{i=t-k}^t |Y_i - Y_{i-1}|)}, \quad q_{t_{(\alpha, \beta, \gamma, \omega, \delta, k)}} = \frac{(|Y_t - \hat{Y}_t|)}{\frac{1}{k} (\sum_{i=t-k}^t |Y_i - Y_{i-1}|)} \\ MASE_{t_{(\alpha, \beta, \gamma, \delta, k, n)}} &= \frac{1}{n} \sum_{i=t-n}^{t-k} q_{i_{((\alpha, \beta, \gamma, \delta, k, n))}}, \\ MASE_{t_{(\alpha, \beta, \gamma, \omega, \delta, k, n)}} &= \frac{1}{n} \sum_{i=t-n}^{t-k} q_{i_{((\alpha, \beta, \gamma, \omega, \delta, k, n))}} \end{aligned} \quad (5)$$

where $k < t$ and $n < t$.

The anomaly is declared if $MASE_t > \delta$, where δ is threshold.

3.2 Finding the Optimal Values of the Algorithm Parameters

The goal of our proposed algorithm is to find the optimal parameter values for the anomaly detection algorithm in order to achieve the correct TP and zero FP and FN. The evaluation of the optimization parameters for the anomaly detection is based on input datasets and annotated anomaly intervals. We define the following procedures for counting the TP, FP and FN:

- TP (true positive) is the number of anomalies annotated intervals with at least one detected anomaly
- FP (false positive) is the number of detected anomalies outside of all annotated intervals
- FN (false negative) in the number of annotated intervals with 0 detected anomalies.

Having defined these values, we use the following evaluation function for our genetic algorithm optimization:

$$EF_{((\alpha, \beta, \gamma, \omega, \delta, k, n, w_1, w_2, w_3, w_4))} = TP * w_1 - FP * w_2 - FN * w_3 - \delta * w_4 \quad (6)$$

where w_1, w_2, w_3 and w_4 are weight factors (constants) that are given based on the importance of the targeted goals. In our case we favor to achieve correct TP, and minimal FP and FN, hence the w_1 is 100 and w_2, w_3 and w_4 are 1.

Based on the defined EF Eq. 6, we use a real-valued GA optimization for parameters optimization using the following constraints:

$$0 < \alpha \leq 1 \quad 0 \leq \beta, \gamma, \omega \leq 1 \quad 0 < \delta < \delta_{max} \quad 0 < n, k \leq 2 * m$$

EF starts with a calculation of a prediction using additive HW (1). Then based on this prediction, we calculate $MASE_t$ (4) and evaluate its value against δ . δ_{max} is defined experimental based on the dataset (in our case 50). If our algorithm detects an anomaly, we add the timestamp to a list of anomalies for further evaluation. The next step is an evaluation of the list of anomalies against the anomaly annotated intervals, thus deriving TP, FP and FN, and finally calculating the EF value. The GA optimization is very effective: we use small populations with less than 100 individuals, and achieve the optimal solutions in less than 20 iterations. The proposed algorithm is implemented in R language.

4 Experimental Results

In this section, the datasets used in the experiments are described. The main part of the section is a comparison of the results (TP, FP, FN, detection rate, precision) achieved with our proposed algorithms HW GA and DTHW GA, compared to several older variations of HW, DTHW and ARIMA, MA, HTM algorithms.

4.1 Experimental Datasets

To evaluate the proposed algorithm we have used the most known benchmarks from the Yahoo, Webscope dataset “*data-labeled-time-series-anomalies-v10*” [24], NAB [25] “*artificialWithAnomaly*” and our real data log-file, generated by the NEIS.

We have exploited the first 4 out of 100 Yahoo synthetic A2 and real A3 and A4 time-series benchmarks, with tagged anomaly points. The datasets are suitable for testing the detection accuracy of various anomaly-types including outliers and change-points. The synthetic dataset consists of time-series with the varying trend, noise and seasonality, while the real one consists of time-series representing the metrics of various Yahoo services. Some datasets have a weekly and some a weekly and daily seasonality Part of the datasets A4 is shown in the Fig. 2 below. NAB contains artificially-generated datasets with varying types of tagged anomalies and a daily seasonality. The NEIS dataset has weekly and daily seasonality. Anomalies are unknown but are analyzed and tagged by a human. All the datasets contain a timestamp and single value based on the log.

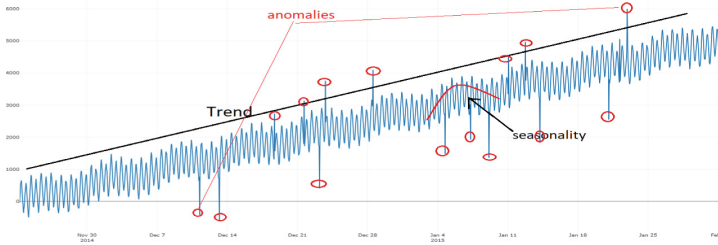


Fig. 2. Yahoo A4 benchmark time series

4.2 Results

In order to evaluate that the optimization of the parameters works well, we have separated the datasets into training (TrS) and test sets (TsS). The optimal values of the parameters are determined on the training set and then they are verified on the test set. Our proposed algorithm (HW GA) with GA optimized parameters $(\alpha, \beta, \gamma, \delta, k, n)$ and with improved $MASE_{t(\alpha, \beta, \gamma, \delta, k, n)}$ is compared with ARIMA, MA (implemented in our previous work [23]), HTM [18] algorithm, HW where smoothing parameters are calculated by formula and default MASE (HW calc.MASE), HW by default smoothing parameters (optimized in R) and default MASE (HW def.MASE), HW by default smoothing parameters and improved $MASE_{(k, n)}$ (HW def.MASE(k,n)).

HW GA counts automatically the number of TP, FP and FN that is not possible with other compared algorithms. The smoothing parameters can be calculated by the formula Eq. 7 were for the total weight we take 0.95:

$$\alpha = 1 - \exp \frac{\log(1 - totalweightsas\%)}{\#oftimepoints} \tag{7}$$

A number of points (frequency) for Yahoo benchmark stream, with week seasonality, is $24 * 7 = 168$, having data each hour. A number of points for the Numenta benchmark stream is $12 * 24 = 288$ having data every 5 min.

To be able to compare the results we use detection rate (recall) in % (d.r.) and precision (prec.), the statistical performance measures of a binary classification test. Due to the big number of the TN-True Negative values, specificity (the true negative rate) and accuracy are not applicable measures for the time series data. In the Tables 1, 2 and 5 below a number of detected TP, FP and FN for NUMENTA, Yahoo, and NEIS on training and test sets are given.

Similarly the Taylor’s Double Holt Winters GA (TDHW GA) with optimized parameters $(\alpha, \beta, \gamma, \omega, \delta, k, n)$ and with improved $MASE_{t(\alpha, \beta, \gamma, \omega, \delta, k, n)}$, is compared with the same algorithms as for HW, where HW type algorithms are replaced with TDHW. In Table 6 below are shown experiments for double seasonality for both training sets and test sets for NEIS data.

Table 1. The result from all tested algorithms for Yahoo benchmark

NUMENTA Benchmark	art daily flatmiddle				art daily jumpup				art increase spike density				art daily jumpdown				art daily nojump						
	TsS		TsS		TsS		TsS		TsS		TsS		TsS		TsS		TsS						
	TP	FP	FN	d.r.	TP	FP	FN	d.r.	TP	FP	FN	d.r.	TP	FP	FN	d.r.	TP	FP	FN	d.r.			
Annotated	0	0	0	100	100	0	0	100	100	1	9	0	3	-	-	0	0	0	100	100	0	0	100
ARIMA	0	3	0	100	20	0	5	110	0	100	9	1	10	0	0	0	115	0	100	6	5		
MA	0	8	0	100	14	0	10	0	150	0	100	17	1	9	0	0	0	100	10	0	3	0	
HTM	0	0	0	100	100	0	0	100	100	1	0	0	0	-	-	0	0	0	100	100	0	0	100
HW calc. MASE	0	0	0	100	100	0	0	100	100	1	45	0	0	40	0	0	0	100	100	0	0	100	
HW def. MASE	0	0	0	100	100	0	0	100	100	1	44	0	0	38	0	0	0	100	100	0	0	100	
HW def. MASE(k,n)	0	0	0	100	100	0	7	0	12	0	33	1	1	0	0	0	100	100	0	0	100		
HW GA	0	0	0	100	100	0	0	100	100	1	0	0	0	0	0	0	100	100	0	0	100		

Table 2. The result from all tested algorithms for Yahoo benchmark

A2 benchmark	Yahoo training and test sets																
	synthetic_1				synthetic_2				synthetic_3				synthetic_4				
	TP	FP	FN	d.r.	TP	FP	FN	d.r.	TP	FP	FN	d.r.	TP	FP	FN	d.r.	
Annotated	4	0	0	100	100	9	0	0	100	100	1	0	0	100	2	0	0
ARIMA	3	2	1	75	60	9	0	0	100	100	1	2	0	100	23	2	3
MA	4	3	0	100	57	9	0	0	100	100	1	3	0	100	25	1	1
HTM	4	0	0	100	100	6	0	3	67	100	1	0	0	100	2	0	0
HW calc. MASE	4	0	0	100	100	9	0	0	100	100	1	0	0	100	100	2	0
HW	4	1	0	100	80	9	1	0	100	90	1	0	0	100	2	0	0
def. MASE	4	0	0	100	100	0	0	9	0	-	0	0	1	0	-	0	0
HW def. MASE(k,n)	4	0	0	100	100	9	0	0	100	100	1	0	0	100	100	2	0
HW GA	4	0	0	100	100	9	0	0	100	100	1	0	0	100	100	2	0
A3 benchmark	A3Benchmark-TS1				A3Benchmark-TS2				A3Benchmark-TS3				A3Benchmark-TS4				
Annotated	11	0	0	100	100	16	0	0	100	100	6	0	0	100	6	0	0
ARIMA	8	7	3	73	53	6	0	10	38	100	3	2	3	50	60	3	4
MA	9	3	2	82	75	4	0	12	25	100	5	3	1	83	63	6	3
HTM	5	0	6	45	100	1	0	15	6	100	0	0	6	0	-	3	0
HW calc. MASE	4	84	7	36	5	16	5	0	100	76	6	0	0	100	100	6	30
HW	10	233	1	91	4	16	150	0	100	10	6	180	0	100	3	6	205
def. MASE	2	26	9	18	7	16	0	0	100	100	6	0	0	100	100	6	0
HW def. MASE(k,n)	7	12	4	64	37	16	0	0	100	100	6	0	0	100	100	6	0
HW GA	11	0	0	100	100	16	0	0	100	100	6	0	0	100	100	6	0
A4 benchmark	A4Benchmark-TS1				A4Benchmark-TS2				A4Benchmark-TS3				A4Benchmark-TS4				
Annotated	13	0	0	100	100	5	0	0	100	100	6	0	0	100	6	0	0
ARIMA	7	3	6	54	70	4	3	1	80	57	4	0	2	67	100	6	5
MA	6	5	7	46	55	3	2	2	60	60	3	1	3	50	75	5	3
HTM	1	0	12	8	100	0	0	5	0	-	2	0	4	33	100	3	0
HW calc. MASE	13	10	0	100	57	5	0	0	100	100	5	0	1	83	100	6	0
HW	13	20	0	100	39	5	3	0	100	63	5	13	1	83	28	6	2
def. MASE	8	2	5	62	80	5	0	0	100	100	4	2	2	67	67	6	0
HW def. MASE(k,n)	2	38	11	15	5	3	0	2	60	100	0	0	6	0	-	2	0
HW GA	13	0	0	100	100	5	0	0	100	100	6	0	0	100	100	6	0

The last rows indicated by gray color show the results of our HW GA. As can be seen in all the cases it outperforms or is equal to the results of the other algorithms. Direct comparison of the result achieved on the same benchmark datasets can be done between proposed HW GA algorithm and HTM anomaly detection algorithm [18] (online implemented on [25]). HW GA and HTM have given equally good results on NUMENTA datasets, while HW GA (100% detection rate and 0% false positive) significantly outperform HTM on all the Yahoo benchmark datasets as also our e-ndevnik dataset. HW GA outperforms the best results (detection rate 84.67%, and false positive 10.12%) of HW forecasting algorithm with parameter maximum likelihood estimates optimization in [6], as also results of another type of algorithms (sliding windows) applied on the similar type of data streams reported in [8]. The other important achievement of the HW GA is that the algorithm is self-learning and can be implemented as a positive feedback optimization with a periodic adaptation of the parameters of the algorithm. In Table 2 the first dataset is used as a training set. Anomalies detected on the second dataset (test set) are verified/acknowledged by human

and reused for new parameter optimization. With such newly optimized parameters detection is implemented on the third set and so on.

Correct results are achieved even in the case when there are no anomalies in the training set, while the test set has anomalies (example in Table 3). In Tables 3, 4, 5 and 6 above, the parameters used by various algorithms are shown. Parameters δ , k , n , tagged by (*) are defined experimentally.

Table 3. Part of Numenta training set and test set optimal parameters

NUMENTA Benchmark												
art_daily_flatmiddle						1-7 Training set			8-14 Test set			
Annotated						0	0	0	1	0	0	
HTM						0	0	0	1	0	0	
	α	β	γ	δ	k	n	TP	FP	FN	TP	FP	FN
HW calc. MASE	0.2209	0.0103	0.3482	22*	/	/	0	0	0	1	0	0
	0.7302	0	0.0257	15*	/	/	0	0	0	1	1	0
HW def. MASE	0.7302	0	0.0257	20*	/	/	0	0	0	1	1	0
	0.7302	0	0.0257	25*	/	/	0	0	0	1	0	0
HW def. MASE(k,n)	0.7302	0	0.0257	4.5*	150*	4*	0	0	0	1	0	0
HW GA	0.1415	0.2648	0.2102	3.1437	75.2621	6.8445	0	0	0	1	0	0

Table 4. Part of Yahoo training set and test set optimal parameters

Yahoo Webscope_S5																		
A3Benchmark						A3Benchmark TS1			A3Benchmark TS2			A3Benchmark TS3			A3Benchmark TS4			
Annotated						11	0	0	16	0	0	6	0	0	6	0	0	
HTM						5	0	6	1	0	15	0	0	6	3	0	3	
	α	β	γ	δ	k	n	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN
HW calc. MASE	0.95	0.1173	0.3482	1*	/	/	4	84	7	16	5	0	6	0	0	6	30	0
	0.1548	0.1163	0.0434	0.1*	/	/	10	233	1	16	150	0	6	180	0	6	205	0
HW def. MASE	0.1548	0.1163	0.0434	0.5*	/	/	6	124	5	16	60	0	6	18	0	6	100	0
	0.1548	0.1163	0.0434	1*	/	/	2	44	9	16	0	0	6	0	0	6	9	0
	0.1548	0.1163	0.0434	1.2*	/	/	2	26	9	16	0	0	6	0	0	6	0	0
HW def. MASE(k,n)	0.1548	0.1163	0.0434	0.9*	12*	8*	7	12	4	16	0	0	6	0	0	6	0	0
HW GA	0.7120	0.6218	0.1068	2.2235	15.6346	4.7449	11	0	0	16	0	0	6	0	0	6	0	0

Table 5. e-dnevnil training set and test set TDHW GA optimal

DETECTORS	e-dnevnik							Training set (two weeks)					Test set (one week)				
	α	β	γ	ω	δ	k	n	TP	FP	FN	d.r.	prec.	TP	FP	FN	d.r.	prec.
Annotated	-	-	-	-	-	-	-	6	0	0	100	100	3	0	0	100	100
ARIMA	-	-	-	-	-	-	-	6	7	0	100	46	3	4	0	100	43
MA	-	-	-	-	-	-	-	6	13	0	100	32	3	11	0	100	21
HTM	-	-	-	-	-	-	-	0	0	6	0	-	0	0	3	0	-
TDHW calc. MASE	0.95	0.1173	0.3482	0.0021	20*	/	/	6	23	0	100	3	3	0	100	50	
	0.6579	0	0	10*	/	/	/	6	230	0	100	3	3	0	100	50	
HW def. MASE	0.0746	0.0169	0.0040	0.2655	10*	/	/	6	341	0	100	2	3	12	0	100	20
	0.0746	0.0169	0.0040	0.2655	40*	/	/	2	35	4	33	5	0	0	3	0	-
TDHW def. MASE(k,n)	0.0746	0.0169	0.0040	0.2655	5.1*	1000*	11*	4	1770	2	67	0.2	1	0	2	33	100
TDHW GA	0.8490	0.2853	0.0125	0.6798	7.0886	322.4625	8.5165	5	100	1	83	5	2	0	1	67	100

Table 6. e-dnevnil training set and test set HW GA optimal parameters

DETECTORS	e-dnevnik							Training set (two weeks)					Test set (one week)				
	α	β	γ	δ	k	n	TP	FP	FN	d.r.	prec.	TP	FP	FN	d.r.	prec.	
Annotated	-	-	-	-	-	-	-	6	0	0	100	100	3	0	0	100	100
ARIMA	-	-	-	-	-	-	-	6	7	0	100	46	3	4	0	100	43
MA	-	-	-	-	-	-	-	6	13	0	100	32	3	11	0	100	21
HTM	-	-	-	-	-	-	-	0	0	6	0	-	0	0	3	0	-
HW calc. MASE	0.95	0.0487	0.3482	10*	/	/	/	6	230	0	100	3	3	0	100	50	
	0.6579	0	0	10*	/	/	/	6	230	0	100	3	3	0	100	50	
HW def. MASE	0.6579	0	0	20*	/	/	/	5	50	1	83	9	1	0	2	83	100
	0.6579	0	0	30*	/	/	/	3	10	3	50	23	1	0	2	50	100
	0.6579	0	0	40*	/	/	/	2	4	4	33	33	0	0	3	33	-
HW def. MASE(k,n)	0.6579	0	0	3*	115*	10*	3	13	3	50	18.8	3	30	0	50	9	9
HW GA	0.4075	0.5093	0.5325	7.2826	330.6001	11.0024	6	0	0	100	100	3	0	0	100	100	

5 Conclusion

Based on the experimental evaluation of the detection rate and precision, performed on sets of synthetic and real data periodic streams, we can conclude that our proposed HW with GA optimized parameters ($\alpha, \beta, \gamma, \delta, k, n$) and with improved MASE outperforms the other algorithms. This can't be concluded for the TDHW with GA optimization. Due to the HW iterative procedures, detection time is appropriate for the real-time anomaly detection. Optimization with GA that is also rather fast, with rather a small number of iterations (about 25–30 iterations are needed to achieve all tagged anomalies recognition in the training sets), can be done in batch mode on training sets, as also re-optimization with verified newly detected anomalies. In our future work, we will incorporate HW GA in our implemented infrastructure [22] for anomaly detection in massive data streams. We plan further investigation and tuning of the TDHW with GA optimization and generalization of the optimization function by including additional parameters in optimization like seasonality and initial values. Ongoing work is motivated by the need of real-time alarm in the case of anomalies in the national online educational system.

References

1. De Gooijer, J.G., Hyndman, R.J.: 25 years of time series forecasting. *Int. J. Forecast.* **22**(3), 443–473 (2006)
2. Adhikari, R., Agrawal, R.K.: *An introductory study on time series modeling and forecasting*. LAP Lambert Academic Publishing, Germany (2013)
3. Chandola, V., Banerjee, A., Kumar, V.: Outlier detection: a survey. *ACM Comput. Surv.* **41**(3), 1–58 (2009)
4. Brutlag, J.D.: Aberrant behavior detection in time series for network monitoring. In: *LISA 2000 Proceedings of the 14th USENIX Conference on System Administration*, pp. 139–146. ACM, Louisiana (2000)
5. Ekberg, J., Ylinen, J., Loula, P.: Network behaviour anomaly detection using Holt-Winters algorithm. In: *6th International Conference on Internet Technology and Secured Transactions*, pp. 627–631. IEEE, Piscataway (2011)
6. Andrysiak, T., Saganowski, L., Maszewski, M.: Time series forecasting using Holt-Winters model applied to anomaly detection in network traffic. In: Pérez García, H., Alfonso-Cendón, J., Sánchez González, L., Quintián, H., Corchado, E. (eds.) *SOCO/CISIS/ICEUTE-2017. AISC*, vol. 649, pp. 567–576. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-67180-2_55
7. Ravinder, H.V.: Determining the optimal values of exponential smoothing constants - does solver really work? *Am. J. Bus. Educ.* **6**(3), 347–360 (2013)
8. Li, G., Wang, J., Liang, J., Yue, C.: The application of a double CUSUM algorithm in industrial data stream anomaly detection. *Symmetry* **10**(7), 264 (2018). 2–14
9. Agapie, A., Agapie, A.: Forecasting the economic cycles based on an extension of the Holt-Winters model. a genetic algorithms approach. In: *Proceedings of the IEEE/IAFE Computational Intelligence for Financial Engineering (CIFER)*, pp. 96–99. IEEE, New York City (1997)

10. Eusébio, E., Camus, C., Curvelo, C.: Metaheuristic approach to the Holt-Winters optimal short term load forecast. *Renew. Energy Power Qual. J.* **10**(13), 708–713 (2015)
11. Shahin, A.A.: Using multiple seasonal Holt-Winters exponential smoothing to predict cloud resource provisioning. *Int. J. Adv. Comput. Sci. Appl. (IJACSA)* **7**(11), 91–96 (2016)
12. Ortiz, R.R.L.: The accuracy rate of Holt-Winters model with particle swarm optimization in forecasting exchange rates. *J. Comput.* **11**(3), 216–224 (2016)
13. Cortez, P., Rocha, M., Neves, J.: Genetic and evolutionary algorithms for time series forecasting. In: Monostori, L., Váncza, J., Ali, M. (eds.) *IEA/AIE 2001. LNCS (LNAI)*, vol. 2070, pp. 393–402. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45517-5_44
14. Hyndman, R.J., Koehler, A.B.: Another look at forecast-accuracy metrics for intermittent demand. *Int. J. Forecast.* **22**(4), 679–688 (2006)
15. Scrucca, L.: GA: a package for genetic algorithms in R. *J. Stat. Softw.* **53**(4), 1–53 (2013)
16. de Assis, M.V.O., Carvalho, L.F., Rodrigues, J.J.P.C., Proença, M.L.: Holt-Winters statistical forecasting and ACO metaheuristic for traffic characterization. In: 2013 IEEE International Conference on Communications (ICC), Budapest, Hungary, pp. 2524–2528 (2013)
17. Maarof, M.Z.M., Ismail, Z., Fadzli, M.: Optimization of SARIMA model using genetic algorithm method in forecasting Singapore tourist arrivals to Malaysia. *Appl. Math. Sci.* **8**(170), 8481–8491 (2014)
18. Ahmad, S., Purdy, S.: Real-time anomaly detection for streaming analytics. arXiv, pp. 1–10 (2016)
19. Hamamoto, A.H., Carvalho, L.F., Sampaio, L.D.H., Abrão, T., Proença Jr., M.L.: Network anomaly detection system using genetic algorithm and fuzzy logic. *Expert. Syst. Appl.: Int. J.* **99**(C), 390–402 (2017)
20. Hyndman, R.J., Athanasopoulos, G.: *Forecasting: Principles and Practice*, 2nd edn. Texts Monash University, Australia (2018)
21. Hasani, Z., Jakimovski, B., Kon-Popovska, M., Velinov, G.: Real-time analytic of SQL queries based on log analytic. In: Loshkovska, S., Koceski, S. (eds.) *ICT Innovations 2015, Web Proceedings*, pp. 78–87. ICT ACT, Macedonia (2015)
22. Hasani, Z.: Implementation of infrastructure for streaming outlier detection in big data. In: Rocha, Á., Correia, A.M., Adeli, H., Reis, L.P., Costanzo, S. (eds.) *WorldCIST 2017. AISC*, vol. 570, pp. 503–511. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56538-5_51
23. Hasani, Z.: Robust anomaly detection algorithms for real-time big data: comparison of algorithms. In: 6th Mediterranean Conference on Embedded Computing (MECO), pp. 1–6. IEEE, Montenegro (2017)
24. Yahoo: S5 - A Labeled Anomaly Detection Dataset, version 1.0(16M). <https://webscope.sandbox.yahoo.com/catalog.php?datatype=s%5c&did=70>. Accessed 28 Apr 2018
25. NUMENTA anomaly benchmark with labeled anomalies. <https://github.com/numenta/NAB/tree/master/data/artificialWithAnomaly>. Accessed 28 Apr 2018



Semantic WordRank: Generating Finer Single-Document Summarizations

Hao Zhang^(✉) and Jie Wang

Department of Computer Science, University of Massachusetts, Lowell, MA, USA
hao_zhang@student.uml.edu, jie_wang@uml.edu

Abstract. We present Semantic WordRank (SWR), an unsupervised method for generating an extractive summary of a single document. Built on a weighted word graph with semantic and co-occurrence edges, SWR scores sentences using an article-structure-biased PageRank algorithm with a Softplus function adjustment, and promotes topic diversity using spectral subtopic clustering under the Word-Movers-Distance metric. We evaluate SWR on the DUC-02 and SummBank datasets and show that SWR produces better summaries than the state-of-the-art algorithms over DUC-02 under common ROUGE measures. We then show that, under the same measures over SummBank, SWR outperforms each of the three human annotators (aka. judges) and compares favorably with the combined performance of all judges.

Keywords: Single-document summarizations · Word Embedding
Topic Clustering · Semantic Similarity · Unsupervised Learning

1 Introduction

An extractive summarization algorithm of a single document aims to produce a much shorter text file to cover all the main points of the document without redundancy. Such an algorithm, unsupervised or supervised, extracts a few critical sentences from the document to form a summary.

An unsupervised method typically solves an optimization problem of selecting sentences subject to the underlying length constraint of a summary through sentence scoring and topic diversity. Unsupervised methods have three advantages: (1) They do not need training data; (2) they are in general independent of the underlying languages; and (3) they are much more efficient. Most unsupervised methods use easy-to-compute counting features, such as term frequency-inverse document frequency (TF-IDF) and co-occurrences of words, discarding semantic information. Some methods do incorporate semantic features including Semantic Role Labeling, WordNet, and Named-Entity Recognition when selecting sentences. These methods rely on specific language features which can be a potential problem when adapting across languages.

Supervised methods, on the other hand, may either require handcrafted features or learn feature representations automatically with a deep neural network model trained on a significant amount of data and is often time-consuming.

These considerations motivate us to investigate unsupervised summarization algorithms using semantic features that can be computed readily for any language.

Word embedding representation maps a word into a high-dimensional vector representing continuous relations with other words, which is easy to obtain and does not rely on specific language features. A good pre-trained word embedding representation can help capture both useful semantic and syntactic information [15]. Word Mover’s Distance (WMD) [12] utilizes this property to measure the distance between two documents, which brings semantics to sentence similarity. Our method, called Semantic WordRank (SWR), is an unsupervised graph-based algorithm with semantic features for generating extractive single-document summaries. It adds word-embedding representations as semantic relations to a graph with word co-occurrence relations. It then uses an article-structure-biased PageRank algorithm to compute a score for each word and a score for each sentence after an adjustment using the Softplus function. To facilitate coverage diversity, our approach uses a spectral subtopic clustering method to group subtopics under the WMD. Finally, it selects sentences with the highest scores over subtopic clusters in a round-robin fashion until the length constraint of the summary size fails.

The rest of the paper is organized as follows: We present in Sect. 2 a brief overview of related work on single-document extractive summarizations. Section 3 provide a detailed description of SWR. We describe in Sect. 4 our evaluation datasets, experiment settings, and comparison results under ROUGE measures on DUC-02 and SummBank datasets. We conclude the paper in Sect. 5.

2 Related Work

Unsupervised Methods. Relations between words play a central role in unsupervised extractive methods, with the underlying idea that related words “promote” each other. In particular, TextRank [14] and LexRank [6] model a document as a sentence graph, and use the PageRank algorithm to rank sentences. TextRank offers robust performance over the DUC-02 dataset. These methods extract sentences based only on sentence scores, without considering topic diversity.

UniformLink [23] builds a sentence graph on a set of similar documents, where a sentence’s score is computed based on both with-in document score and cross-document score. URank [22] uses a unified graph-based framework to study both single-document and multi-document summarizations.

E_{coh} [19], as well as T_{coh} [18], use a bipartite graph to represent a document and a different algorithm, Hyperlink-Induced Topic Search (HITS) [11], is used to score sentences. They both treat the summarization problem as an ILP problem, which maximizes the sentence importance, non-redundancy, and coherence at the same time. However, since ILP is an NP-hard problem, obtaining an exact solution to an ILP problem is intractable.

Submodularity optimization [13] and Latent Semantic Analysis [8] are two other widely used unsupervised techniques for extractive summarizations.

Supervised Methods. Traditional supervised machine learning methods often need handcrafted features. These methods include Support Vector Machine and Naive Bayesian Classifier [24]. CP3 [17], which has the same underlying idea as E_{coh} and T_{coh} , mines coherence patterns in a corpus of abstracts, and extracts sentences by solving an ILP problem.

Deep learning methods, able to learn sentence or document representations automatically, have recently been used to score sentences. For example, R2N2 [4] uses a recursive neural network for both word level and sentence level scoring, followed by an ILP optimization strategy for selecting sentences. CNN-W2V [25] is another example, which modifies a convolutional-neural-network (CNN) model of sentence classification [9] to rank sentences. SummaRuNNer [16], on the other hand, treats summarizations as sequence classifications and uses a two-layer bi-directional recurrent neural network (RNN) model to extract sentences, where the first layer RNN is for words and the second layer is for sentences. Unlike unsupervised methods, the state-of-the-art deep learning approaches require a larger dataset and a significantly longer time to train a model, yet with a much lower ROUGE-1 score when evaluating on DUC dataset.

3 Semantic WordRank

We model a single-document extractive summarization problem as a 0-1 multi-objective knapsack problem. Let D be a document consisting of n sentences indexed as S_1, S_2, \dots, S_n in the order they appear, each with a length l_i and a score s_i , along with a maximum length capacity L , where l_i is the number of characters contained in S_i . Let $F_d(D)$ denote a diversity coverage measure and x_i a 0-1 variable such that $x_i = 1$ if sentence S_i is selected, and 0 otherwise. We model the summarization problem as the following multi-objective optimization problem:

$$\text{maximize } \sum_{i=1}^n s_i x_i \text{ and } F_d(D), \text{ such that } \sum_{i=1}^n l_i x_i \leq L \text{ and } x_i \in \{0, 1\}. \quad (1)$$

In this section, we describe Semantic WordRank in details. In particular, we first describe how we score sentences, followed by a spectral subtopic clustering method to facilitate diversity. Finally, instead of solving the 0-1 knapsack problem to obtain an optimal solution, which is NP-hard, we use a greedy strategy to obtain an approximation.

3.1 Semantic Word Graphs

Let $G = (V, E)$ denote a weighted, undirected multiple-edge graph to represent a document D , where V is a subset of words contained in D that pass a part-of-speech filter, a stop-word filter, and a stemmer for reducing inflected words to the word stem. Two nodes in G are connected if they co-occur within a window of N successive words in the document, or the cosine similarity of their word-embedding representations exceeds a threshold value Δ . Let u and v be two

adjacent nodes. Initially, they may have two types of connections, one for co-occurrence and one for semantic similarity. We assign the co-occurrence count of u and v as the initial weight to the co-occurrence connection, and the cosine similarity value as the initial weight to the semantic connection. Let $w_{u,v}^c$ and $w_{u,v}^s$ denote, respectively, the normalized weight for the co-occurrence connection and the semantic connection of u and v . Finally, we assign $w_{u,v} = w_{u,v}^c + w_{u,v}^s$ as the weight to the incident edge of u and v .

Note that the original TextRank algorithm [14] only considers co-occurrence between words. We add semantic similarity to represent that a pair of words express the same or similar meanings.

On a separate note, adding semantic similarity is vital for processing analytic languages, such as Chinese, which seldom use inflections. When stemming is not applicable, semantic similarity serves as an alternative to represent the relations between words with similar meanings, allowing them to share the importance when computing the PageRank scores for these nodes.

3.2 Article-Structure-Biased PageRank

Article structures define the order how to present information. For example, the typical structure of news articles is an inverted pyramid, presenting critical information at the beginning, followed by additional information with less important details. In academic writing, the structure would look like an hourglass, which includes an additional conclusion piece at the end of the article. To include article structures in our model, we use a position-biased PageRank algorithm [7].

Let G denote the word graph constructed in the previous section. The original TextRank algorithm computes score $W(v_i)$ of a node v_i by iterating the following PageRank equation until converging, where $A(v_i)$ denotes the set of nodes adjacent to v_i :

$$W(v_i) = \alpha \times \sum_{v_j \in A(v_i)} \frac{w_{ji}}{\sum_{v_k \in A(v_j)} w_{jk}} W(v_j) + (1 - \alpha), \quad (2)$$

where $\alpha \in (0, 1)$ is a damping factor, often set to 0.85 [3], and w_{ji} is the weight of the edge between node j and node i . The intuition behind this equation is that the importance of a node v_i is related to the scores of its adjacent nodes and the probability $\beta = 1 - \alpha$ of jumping from a random node to node v_i . In unbiased PageRank, each word is treated equally likely. In article-structure-biased PageRank, each word v_i is biased with a probability $P(v_i)$ according to the underlying article structure. For example, in the inverted pyramid structure, we assign a higher probability to a word that appears closer to the beginning of the article.

We rank the importance of sentences from the most important to the least important based on the underlying article structure. In the inverted pyramid structure, we assign score s_i to sentence S_i using the reciprocal of indexing (or negative indexing). The probability for node v_i can now be computed as follows:

$$P(v_i) = \frac{C(v_i)}{\sum_{v_j \in V} C(v_j)}, \quad C(v_i) = \sum_{v_i \in S_k} s_k.$$

Note that the above computation is at the sentence level, which can be easily adapted to the word level by ranking words instead of sentences.

We compute the revised TextRank score $W'(v_i)$ for node v_i as follows:

$$W'(v_i) = \alpha \times \sum_{v_j \in A(v_i)} \frac{w_{ji}}{\sum_{v_k \in A(v_j)} w_{jk}} W'(v_j) + (1 - \alpha) \times P(v_i). \quad (3)$$

We first assign an arbitrary value to each node and iterate the computation until it converges. The score associated with each node represents the word importance, and we refer it to as *salient score*.

3.3 Sentence Scoring with Softplus Adjustment

We use $W'(v_i)$ to compute a salient score for each sentence, where v_i is a node in the word graph. At the first glance, one would sum up $W'(v_i)$ for words contained in sentence S to be the salient score of S . This method has the following drawback. Suppose that two sentences S_1 and S_2 have similar salient scores with different W' score distributions, where S_1 has a bipolar word score distribution where several keywords have very high scores and the rest have very low scores, while S_2 has roughly a uniform word score distribution. In this case, we would consider S_1 more critical than S_2 , but using this method to compute a salient score of a sentence, we may end up with an opposite result.

To overcome this drawback, we use a Softplus Adjustment. The Softplus function $sp = \ln(1 + e^x)$, commonly used as an activation function in neural networks, offers a significant enhancement when the input x is a small positive number. When x is large, we have $sp(x) \approx x$. We apply the Softplus Adjustment to each keyword, and then sum them up to get a salient score by

$$Salience_{sp}(S) = \sum_{i=1}^k sp(salience_{w_i}). \quad (4)$$

3.4 Spectral Clustering for Topic Diversity

Spectral clustering [21] uses eigenvalues of an affinity matrix to reduce dimensions before clustering. This model offers better performance over K-means with fewer requirements. The affinity matrix is an $n \times n$ square matrix, where each element corresponds to a similarity between two sentences. Since we use WMD metric, where smaller value between two sentences means more similar and larger value means less similar, we can use a RBF kernel to transfer WMD scores to similarity as follows: $sim(S_i, S_j) = e^{-\gamma * WMD(S_i, S_j)^2}$, where γ can be set to 1.

Note that the time complexity for computing WMD is cubical in terms of the number of unique words in the sentences, making it difficult to scale. Linear-Complexity Relaxed WMD [1] provides a faster approximation to WMD and we will use it to reduce time complexity.

We set the number of clusters c_{num} based on the number of sentences n . According to our experiments, we find that 30% of the original text size would be the best size for a summary to contain almost all significant points. To ensure 30% summary size with no redundancy, we set $c_{num} = \min\{[0.3n], 8\}$.

3.5 Greedy Sentence Selections

Each sentence S_i now associates with four values: (1) sentence index i , (2) salient score s_i , (3) sentence length l_i , and (4) cluster index c_j it belongs to. We select sentences in a round robin fashion as follows:

1. For each sentence S_i , compute the value per unit length using $s'_i = s_i/l_i$.
2. For each cluster, sort the sentences contained in it by their unit scores s'_i .
3. Group sentences with the highest unit scores, one for each cluster, to form a sentence sequence $S_c = \langle S_x, S_y, \dots, S_{c_{num}} \rangle$. Select one sentence at a time from S_c to the summary. Repeat this step while the size of the summary is less than L .

4 Experiments

The DUC-02 [5] dataset contains a total of 567 news articles with an average of 25 sentences per document. Each article has at least two abstractive summaries written by human evaluators, where each summary is at most 100 words long. It has been a standard practice to use the DUC-02 benchmarks to evaluate the effect of summarization algorithms, including extractive summaries, even though DUC-02 benchmarks are abstractive summaries.

The SummBank [20] dataset is a better dataset for evaluations, for it provides extractive summary benchmarks produced by human evaluators. SummBank offers, for each article, four extractive summaries, with a wide range of summary length from 5% up to 90% of, respectively, words and sentences. Three human judges annotate 200 news articles with an average of 20 sentences per document. In particular, given a document, each judge assigns an importance score to each sentence, resulting in, for a given length constraint, three extractive summaries, one for each judge, by selecting sentences with scores as high as possible and as relevant as possible. Moreover, summing up the three judges' scores for each sentence, we get a combined score and new summaries based on the new scores. We refer to the new summaries as a combined performance of all judges.

ROUGE is the most used metric to evaluate the effect of summarizations. ROUGE-n is an n-gram recall between the automatic summary and a set of references, where ROUGE-SU4 evaluates an automatic summarization using skip-bigram and unigram co-occurrence statistics, allowing at most four intervening unigrams when forming skip-bigram.

4.1 Setup

We use a pre-trained word embedding representations with subword information [2] which can help solve the out-of-vocabulary problem and generate better word

embedding for rare words. We run SWR (Semantic WordRank) to evaluate 30 articles selected uniformly at random from the DUC-01 dataset, and use the similarity value denoted by Δ that maximizes the ROUGE-1 score as the threshold value for evaluations on DUC-02 and SumBank. We set $N = 2$ as the window size for co-occurrences. Since both datasets contain only short news articles, it is reasonable to use inverted pyramid as the article structure.

4.2 Evaluations

On the DUC-02 dataset, we use SWR to extract sentences with a length of 100 words and use ROUGE to evaluate the results against the benchmarks. We can see from Table 1 that SWR outperforms the state-of-the-art algorithms under ROUGE-1 (R-1), ROUGE-2 (R-2), and ROUGE-SU4 (R-SU4), where UA means that the value is unavailable in the corresponding publications.

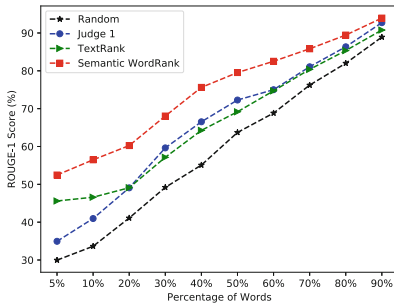
Table 1. Comparison results (%) on DUC-02

Methods	ULink	TextRank	$T_{Coh.}$	URank	$E_{Coh.}$	CNN-W2V	CP3	SWR
R-1	47.1	47.1	48.1	48.5	48.5	48.6	49.0	49.2
R-2	20.1	19.5	24.3	21.5	23.0	22.0	24.7	24.7
R-SU4	UA	21.7	24.2	UA	25.3	UA	25.8	26.1

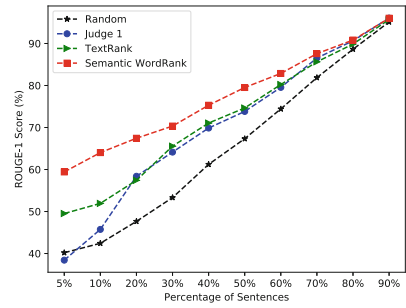
We can see that SWR produces better R-1 and R-SU4 scores over those of CP3 and produces the same R-2 score as that of CP3. But more importantly, SWR offers higher adaptability across languages and lower time complexity. CP3 is a supervised method that requires annotated data to train coherence patterns, where entity recognition and co-reference resolution are needed. Moreover, CP3 generates summarization by solving a time-consuming ILP problem. On the contrary, SWR is an unsupervised model using only word embedding features that can be obtained easily for any language.

Comparisons with Individual Judges on SumBank. To compare with each judge’s summaries, for each judge i ($i = 1, 2, 3$), we use the other two judges’ summaries as golden standards. Comparison with each judge are given in Fig. 1, where the words limitation or sentences limitation is from 5% to 90%. Table 2 depicts the ROUGE scores of different methods on summaries with 30% of sentences and words, where S represents that the percentage is on the number of sentences and W on the number of words. We can see that while TextRank is comparable with individual judges, SWR significantly outperforms each judge.

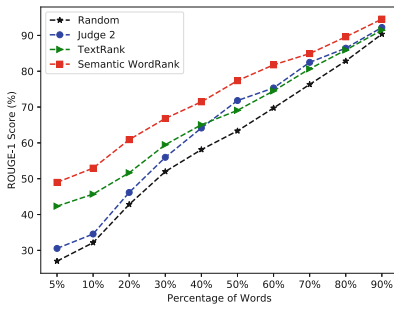
Comparisons with the Combined Performances of All Judges. To compare SWR with the combined performance of all judges, we use the summaries of individual judges as golden standards. Figure 2 depicts the comparison results.



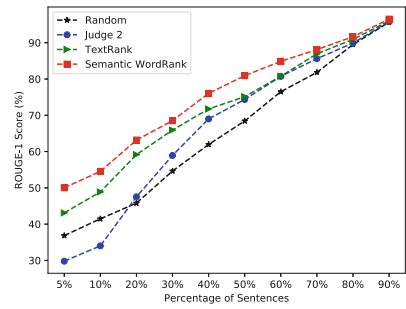
(a) Judge 1 on words



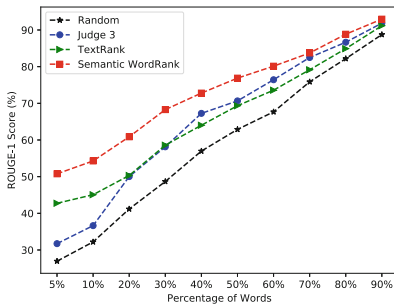
(b) Judge 1 on sentences



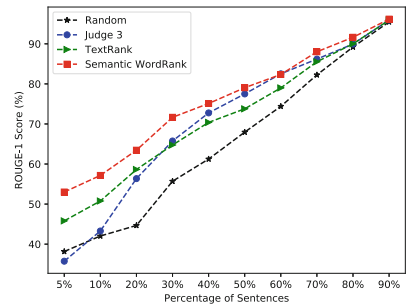
(c) Judge 2 on words



(d) Judge 2 on sentences



(e) Judge 3 on words



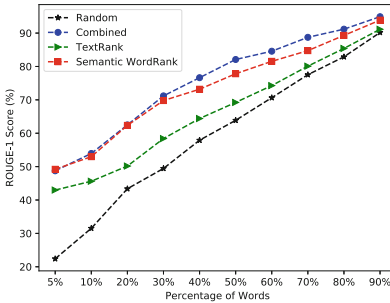
(f) Judge 3 on sentences

Fig. 1. ROUGE-1 (%) comparisons of SWR with individual judges, TextRank, and Random methods on the SummBank benchmarks

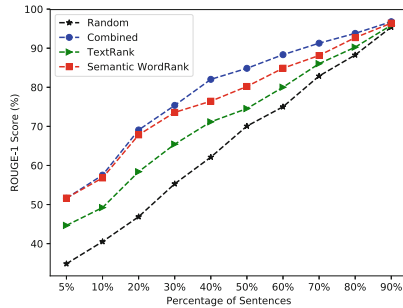
We can see that SWR is compatible with the combined performances of all judges for summaries of length on and below 30%, and close to the combined performance of all judges for more extended summaries.

Table 2. ROUGE (%) comparisons of SWR individual judge, TextRank, and Random methods on SummBank with 30% length constraint

Methods	R-1 (S)	R-2 (S)	R-SU4 (S)	R-1 (W)	R-2 (W)	R-SU4 (W)
Random	55.70	45.44	43.69	49.19	37.53	36.44
Judge 1	64.14	57.11	55.81	59.63	51.18	49.95
TextRank	65.56	58.28	56.79	57.14	47.70	46.69
SWR	70.35	64.47	63.04	68.15	63.18	61.92
Random	54.64	44.13	42.91	51.99	40.67	39.42
Judge 2	58.94	51.50	49.95	55.97	47.57	46.27
TextRank	65.92	59.09	57.82	59.48	51.06	50.06
SWR	67.53	61.45	60.14	66.11	59.08	58.06
Random	55.70	45.44	43.69	48.67	37.07	36.08
Judge 3	65.75	58.44	57.28	58.14	49.34	48.16
TextRank	64.75	57.32	55.70	58.58	49.72	48.53
SWR	72.69	67.65	66.67	69.32	63.39	62.14



(a) Different word constraints



(b) Different sentence constraints

Fig. 2. ROUGE-1 (%) comparisons of SWR with the combined performance of all judges, TextRank, and Random methods on the SummBank benchmarks

Other Comparisons. We investigate if semantic edge, Softplus function adjustment, article-structure-biased PageRank, and subtopic clustering work as expected. In particular, we remove each part one at a time to form a new method and evaluate it on the full SummBank dataset.

Table 3 depicts the evaluation results on 10%, 40%, and 70% sentence constraints, where SWR_NSE, SWR_NAS, SWR_NSC, and SWR_NSP denote, respectively, SWR using no semantic edges, no article-structure information, no subtopic clustering, and no softplus adjustment. TextRank is added as a baseline model. We can see that when we remove a feature, the ROUGE scores drop, indicating that each part is essential. The numbers in bold are the biggest drops. We also see that when the summaries are short, removing article structure results

Table 3. ROUGE (%) comparison with different features removed

Methods	R-1	R-2	R-SU4	R-1	R-2	R-SU4	R-1	R-2	R-SU4
	10%			40%			70%		
SWR	56.82	48.18	47.03	76.48	71.74	70.39	88.21	86.17	85.19
SWR_NSE	55.02	46.08	45.15	71.98	66.76	65.42	85.94	83.18	81.83
SWR_NAS	51.74	40.87	40.13	73.52	68.26	67.12	87.82	84.95	84.12
SWR_NSC	56.36	47.34	46.44	75.68	70.35	69.03	87.54	84.49	83.54
SWR_NSP	56.77	48.12	46.97	76.39	71.59	70.25	88.14	86.04	85.07
TextRank	49.22	36.07	35.29	71.16	65.47	63.94	86.04	83.52	82.29

in much larger drops, indicating that article structures are a more significant feature than the other two features. As summaries become longer, including semantic edges would become increasingly more critical. Our results indicate that using the Softplus function adjustment does improve the ROUGE scores, but not as significant as using the other three features.

5 Conclusion and Future Work

We present Semantic WordRank (SWR), an unsupervised method for generating an extractive summary of a single document. SWR incorporates semantic similarity to summarization while maintaining a high adaptability across languages. In particular, SWR generates an extractive summary using word embedding, Softplus function adjustment, article-structure-biased PageRank, and WMD spectral subtopic clustering. We evaluate SWR on DUC-02 and SummBank under three common ROUGE measures. Our experimental results show that, on DUC-02, our approach produces better results than the state-of-the-art methods under ROUGE-1, ROUGE-2, and ROUGE-SU4. We then show that, under the same ROUGE measures over SummBank, SWR outperforms each of the three individual human judges and compares favorably with the combined performance of all judges.

For a future project, we plan to adopt this approach to multi-document summarizations. We would also like to explore unsupervised sentence representations such as skip-thought vectors [10] for summarization.

Acknowledgments. We thank Liqun Shao for interesting conversations on using the Softplus function adjustment.

References

1. Atasu, K., et al.: Linear-complexity relaxed word mover's distance with GPU acceleration. In: 2017 IEEE International Conference on Big Data (Big Data), pp. 889–896. IEEE (2017)
2. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. arXiv preprint [arXiv:1607.04606](https://arxiv.org/abs/1607.04606) (2016)
3. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.* **30**(1–7), 107–117 (1998)
4. Cao, Z., Wei, F., Dong, L., Li, S., Zhou, M.: Ranking with recursive neural networks and its application to multi-document summarization. In: AAAI, pp. 2153–2159 (2015)
5. DUC: Document understanding conference 2002 (2002)
6. Erkan, G., Radev, D.R.: LexRank: graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.* **22**, 457–479 (2004)
7. Florescu, C., Caragea, C.: A position-biased pagerank algorithm for keyphrase extraction. In: AAAI, pp. 4923–4924 (2017)
8. Gong, Y., Liu, X.: Generic text summarization using relevance measure and latent semantic analysis. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 19–25. ACM (2001)
9. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint [arXiv:1408.5882](https://arxiv.org/abs/1408.5882) (2014)
10. Kiros, R., et al.: Skip-thought vectors. In: Advances in Neural Information Processing Systems, pp. 3294–3302 (2015)
11. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. In: Proceedings of the ACM-SIAM Symposium on Discrete Algorithms. Citeseer (1998)
12. Kusner, M., Sun, Y., Kolkin, N., Weinberger, K.: From word embeddings to document distances. In: International Conference on Machine Learning, pp. 957–966 (2015)
13. Lin, H., Bilmes, J.: A class of submodular functions for document summarization. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, pp. 510–520. Association for Computational Linguistics (2011)
14. Mihalcea, R., Tarau, P.: TextRank: bringing order into text. In: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (2004)
15. Mikolov, T., Yih, W.T., Zweig, G.: Linguistic regularities in continuous space word representations. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 746–751 (2013)
16. Nallapati, R., Zhai, F., Zhou, B.: Summarunner: a recurrent neural network based sequence model for extractive summarization of documents. In: AAAI, pp. 3075–3081 (2017)
17. Parveen, D., Mesgar, M., Strube, M.: Generating coherent summaries of scientific articles using coherence patterns. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 772–783 (2016)
18. Parveen, D., Ramsil, H.M., Strube, M.: Topical coherence for graph-based extractive summarization. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 1949–1954 (2015)

19. Parveen, D., Strube, M.: Integrating importance, non-redundancy and coherence in graph-based extractive summarization. In: IJCAI, pp. 1298–1304 (2015)
20. Radev, D., et al.: Summbank 1.0 ldc2003t16. web download. Linguistic Data Consortium, Philadelphia (2003)
21. Von Luxburg, U.: A tutorial on spectral clustering. *Stat. Comput.* **17**(4), 395–416 (2007)
22. Wan, X.: Towards a unified approach to simultaneous single-document and multi-document summarizations. In: Proceedings of the 23rd International Conference on Computational Linguistics, pp. 1137–1145. Association for Computational Linguistics (2010)
23. Wan, X., Xiao, J.: Exploiting neighborhood knowledge for single document summarization and keyphrase extraction. *ACM Trans. Inf. Syst. (TOIS)* **28**(2), 8 (2010)
24. Wong, K.F., Wu, M., Li, W.: Extractive summarization using supervised and semi-supervised learning. In: Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1, pp. 985–992. Association for Computational Linguistics (2008)
25. Zhang, Y., Er, M.J., Pratama, M.: Extractive document summarization based on convolutional neural networks. In: IECON 2016–42nd Annual Conference of the IEEE Industrial Electronics Society, pp. 918–922. IEEE (2016)



Exploratory Study of the Effects of Cardiac Murmurs on Electrocardiographic-Signal-Based Biometric Systems

M. A. Becerra^{1,2(✉)}, C. Duque-Mejía¹, C. Zapata-Hernández¹,
D. H. Peluffo-Ordóñez³, L. Serna-Guarín⁴, Edilson Delgado-Trejos⁴,
E. J. Revelo-Fuelagán⁵, and X. P. Blanco Valencia³

¹ Institución Universitaria Pascual Bravo, Medellín, Colombia
migb2b@gmail.com

² Universidad de Medellín, Medellín, Colombia

³ SDAS Research Group, Yachay Tech, Urcuquí, Ecuador

⁴ Instituto Tecnológico Metropolitano, Medellín, Colombia

⁵ Universidad de Nariño, Pasto, Colombia

<http://www.sdas-group.com>

Abstract. The process of distinguishing among human beings through the inspection of acquired data from physical or behavioral traits is known as biometric identification. Mostly, fingerprint- and iris-based biometric techniques are used. Nowadays, since such techniques are highly susceptible to be counterfeited, new biometric alternatives are explored mainly based on physiological signals and behavioral traits -which are useful not only for biometric identification purposes, but may also play a role as a vital signal indicator. In this connection, the electrocardiographic (ECG) signals have shown to be a suitable approach. Nonetheless, their informative components (morphology, rhythm, polarization, and among others) can be affected by the presence of a cardiac pathology. Even more, some other cardiac diseases cannot directly be detected by the ECG signal inspection but still have an effect on their waveform, that is the case of cardiac murmurs. Therefore, for biometric purposes, such signals should be analyzed submitted to the effects of pathologies. This paper presents a exploratory study aimed at assessing the influence of the presence of a pathology when analyzing ECG signals for implementing a biometric system. For experiments, a data base holding 20 healthy subjects and 20 pathological subjects (diagnosed with different types of cardiac murmurs) are considered. The proposed signal analysis consists of preprocessing, characterization (using wavelet features), feature selection and classification (five classifiers as well as a mixture of them are tested). As a result, through the performed comparison of the classification rates when testing pathological and normal ECG signals, the cardiac murmurs' undesired effect on the identification mechanism performance is clearly unveiled.

Keywords: Biometric identification · Cardiac murmur
Electrocardiographic signal · Signal processing

1 Introduction

The biometric identification is the verification of the identity of the person based on characteristics of his body, either by features or signs. Biometric identification has been a subject of great interest, and in recent years, these systems have been seen in many of the daily tasks using fingerprint or iris. The applications of biometric identification are multiple, ranging from the identification of diseases for diagnostic purposes to the identification of individuals for security purposes, which has improved in many aspects, the privacy of people and the improvement in the diagnosis of the diseases avoiding mistakes due to identity confusion.

One of the most used identification techniques is the fingerprint, which is still a very efficient method since each has different morphological characteristics and is a trait of easy acquisition [15]. Nevertheless, it is very vulnerable to counterfeiting [12, 14]; therefore it is applied supervised way, but at the same time, different techniques have been developed to avoid these identification systems, which are falsifies using different materials that allow acquiring and use this fingerprint illegally [20].

To improve the shortcomings of conventional biometrics [22] such as fingerprint [11] and iris [5], the use of the electrocardiogram (ECG) as a method of biometric identification has been proposed [15, 16, 19]. ECG signals are reliable in comparison with conventional biometrics because they generate greater security and manage to meet many of the requirements necessary to become an ideal biometric system [17]. ECG has important characteristics such as the size of the heart and spatial location, different subjacent electrical and mechanical dynamics that become unique, among others. This signal has uniqueness, permanence, universality, and evasion. In comparison with other biometrics, the ECG has the inherent capacity of life detection for identification [4]. Moreover, the ECG signals can be acquired by three different acquisition methods defined by [3] as follows: (i) in the person (use invasive equipment is used, designed to be used within the human body), (ii) on the person (signals acquired by electrodes attached to the skin), and (iii) off the person (minimal contact on the skin or without contact), an example is to acquire the signal through fingers [8].

Currently, there have been multiple studies of biometric identification based on ECG signal achieving accuracies of 94.9% [18] and 97.6% [23]. The most common methods for ECG identification have been wavelet transform for feature extraction and Support Vector Machine for classification [6, 10, 17]. However, there are no studies that define the effects of diseases or conditions in biometric identification. The ECG signals have high power for the diagnosis of multiple cardiac alterations. Therefore, some studies of biometric systems based on ECG have considered alterations to improve its performance and reliability [21]. However, some heart diseases that cannot be detected throughout ECG signals affect their morphology such as cardiac murmurs, which are considered as one of the

most common failures of the heart and they have been widely studied from phonocardiograms and echocardiograms for detecting different pathologies [2,9].

Therefore, the primary objective of this work is to analyze the effects of cardiac murmurs on biometric identification based on ECG signals which still have not been studied (taking into account our review) The second objective is to achieve an effective mechanism based on signal processing, and pattern recognition techniques. In this study are studied 40 subjects without distinguishing gender and age for human identification from ECG signals. 20 subject without heart pathologies and 20 with cardiac murmurs. A database was collected with the help of specialists in cardiology 8 continuous registers were acquired by subject. Then, the signals were filtered, standardized, and manually segmented by beat in the preprocessing stage. In the feature extraction step, the signals were decompose using discrete wavelet transform, and maximal overlap discrete wavelet transform, and different features were calculated using linear and non-linear statistical measures alongside Mel frequency cepstral coefficients extracted from the ECG signals. A relevant analysis was carried out using the Relief F algorithm. Finally, five classifiers and a mixture of them were tested. The best global result 91.19% was achieved using the LDC classifier.

2 Experimental Setup

This study was carried out in 5 main stages as it is shown in Fig. 1. First, was collected a database of ECG signals, then, segmentation and standardization were applied in pre-processing step. In third stage feature extraction was carried out using multiple techniques based on Mel Frequency cepstral coefficients (MFCC), linear and non-linear measures of ECG signals, DWT, and MODWT. To reduce the dimensionality of the feature space was performed relief F algorithm. Finally, in the stage 5 was applied multiple classifiers as follows: Support Vector Machine (SVM), Quadratic Bayes Normal Classifier (QDC), Optimisation of the Parzen classifier (PZ), k-Nearest Neighbor (K-NN), and Linear Bayes Normal Classifier (LDC) alongside a mixture of them [13].

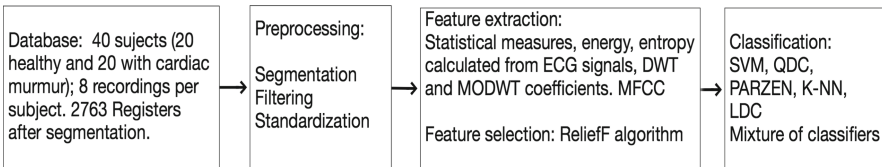


Fig. 1. Experimental procedure

2.1 Database

320 ECG recordings correspond to 40 subjects (20 healthy subjects and 20 subjects with heart disease) were selected from a database of ECG recordings collected from 143 adult subjects of whom 55 patients were labeled as normal, and 88 had evidence of cardiac murmurs (aortic stenosis, mitral regurgitation, among others). The subjects gave their formal consent and underwent a medical examination with the approval of the ethical committee. Cardiologists evaluated the valve lesion severity according to a clinical routine. Eight recordings of lasts 8 s were recorded from each subject in the phase of post-expiratory and post-inspiratory apnea. The signals were acquired at 44.1 kHz with 16-bits per sample. For this study were selected 40 subjects, 20 healthy and 20 with cardiac murmurs. In Fig. 2 are shown three normal ECG signals, and in Fig. 3 are shown three pathological signals, which evidence morphological differences generated by cardiac murmurs.

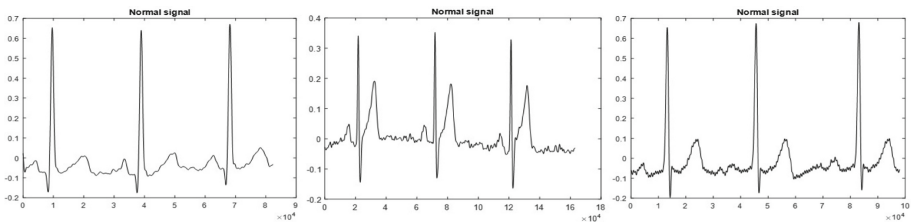


Fig. 2. Normal ECG signals

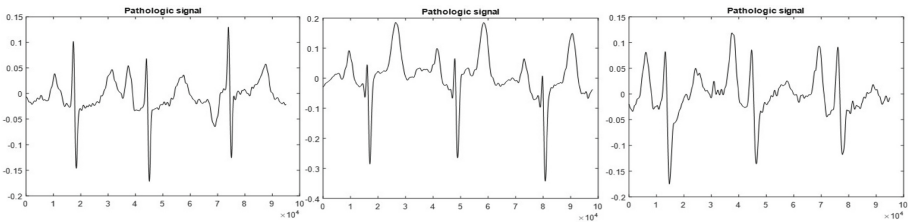


Fig. 3. Pathological ECG signals

2.2 Preprocessing

The ECG signals were manually segmented per beat. Then a standardization was applied to the amplitude of the signals between $[-1 \ 1]$. Finally, to eliminate baseline noise a Butterworth bandpass filter of second order was used with a cutoff frequency of 0.5 Hz and cutoff 150 Hz.

2.3 Feature Extraction

The following techniques were applied to ECG segmented signals: (i) 39 MFCC were calculated from ECG signals, including delta coefficients, delta-delta coefficients, and log energy, using 24 Hamming shaped filters and sliding hamming windows (50% overlap). (ii) 65 features obtained of measures shown in Table 1 applied to coefficients of DWT db10 of 4 levels. (iii) 65 features obtained of measures shown in Table 1 applied to coefficients of MODWT [1] db10 of 4 levels. (iv) 10 features obtained of measures shown in Table 1 applied to ECG signals. In total 179 features were extracted for each signal.

Table 1. Measures calculated from ECG signals - MODWT and DWT

Renyi entropy	Root mean square	Standard deviation	Entropy	Variance	Power	Energy
Shannon entropy	Log energy entropy	Covariance	Kurtosis	Min/max	Mean	

2.4 Feature Selection

The ReliefF algorithm is a filter method that ponders each feature according to its relevance to each class. The weights of them are updated iteratively [24]. This method was applied to 3 datasets for reducing the dimensionality of the following feature space: (i) Features obtained from segmented pathological signals, (ii) Features obtained from segmented normal signals, and (iii) Features obtained from segmented normal and pathological signals. The selection criteria was a ranked cumulative above of 95%.

2.5 Classification

Support Vector Machine, QDC Bayesian classification, PARZEN, k-NN and LDC classifiers, and a mixture among them were tested. Six methods of the mixture were applied as follows: Product, Mean combiner, Median combiner, Maximum combiner, Minimum combiner, and majority vote (Vm). These techniques are explained in [13]. These classifiers were validated using cross-validation with 10-fold.

3 Results and Discussion

In Table 2 are shown the results obtained by the different classifiers (SVM, QDC, K-NN, LDC) and mixture of all them using the product, mean, median, max, min and Vm operations. These classifiers systems were tested with the normal signals (NS) dataset and pathological signals (PS) dataset and All (PS and NS datasets). The best individual results 95.31% were obtained by PZ and K-NN classifiers for PS. LDQ classifier achieved the best performance 97.74% of accuracy for Normal

Table 2. Accuracy of classifiers systems

Signals	SVM	QDC	PZ	K-NN	LDC	Prod	Mean	Median	Max	Min	Vm
PS	71.09	81.25	95.31	95.31	92.97	85.94	90.62	89.84	90.62	79.69	94.53
NS	45.86	84.21	87.22	86.47	97.74	87.22	86.47	87.22	86.47	54.89	93.23
ALL	29.12	78.54	77.39	79.31	91.19	80.84	81.23	75.48	81.23	47.51	87.36

signals and 91.19 for all signals (PS and NS). The best results using a mixture of classifier were achieved using Vm technique for mixturing.

This work is compared in Table 3 with other studies of the biometric systems based on ECG signals using the accuracy. The best global result presented in this work is not comparable with the different approaches due to that the ECG biometric studies do not analyze the effects of the cardiac murmurs, despite some of them included into their studies normal and pathological ECG signals. The best result 97.74% was achieved in this work using normal signals. Nonetheless, the results obtained when were included ECG signals affected by cardiac murmurs were decreased 6.55%, resulting in performance lower than the showed by SVM and ANN approaches.

Table 3. Comparison with other approaches

Approach	SVM [23]	ANN [23]	SVM-OAA [7]	LDC (this work-ALL)	LDC (this work-NS)
Accuracy	96.6%	97.6%	88.41%	91.19%	97.74%

4 Conclusions

In this paper, a study of cardiac murmur effects on biometric identification based on ECG signals was presented. Five classifiers and six mixture of them were tested using normal and pathological signals individually and together. LDQ classifier shown the best global performance 91.19% and separately for normal signals 97.74%. K-NN and PZ classifiers shown the best results 95.31% for pathological signals. The majority vote technique for the mixture of classifiers shows the best results, but these were lower compared with the performance of individual classifiers.

Based on the results achieved with the pathological ECG signals vs. normal ECG signals for human identification was demonstrated that the cardiac murmurs affect the biometric identification based on ECG signals and generate opposite effects on identification performance between PS and NS. This effect can be explained, taking into account that cardiac murmurs are caused when the blood flow becomes turbulent near damaged cardiac valves that can have low generalization and they generate electrical changes in the cardiac muscle which elicits changes in the ECG signals. In conclusion, the cardiac murmur effects on the biometric system based on ECG signals is a characteristic that affects the

performance of the system, but no helps to human identification although the accuracy of the system can be upper for some classifiers using only pathological signals. Nevertheless, the global accuracy of identification including pathological and normal signals decrease the performance of the system. As future work, we proposed to analyze effects of post-expiratory and post-inspiratory apnea phases when acquiring ECG signals for biometric systems, and include another type of cardiac diseases into the study to achieve a significant generality of biometric systems based on ECG signals.

Acknowledgment. The authors acknowledge to the research project “Desarrollo de una metodología de visualización interactiva y eficaz de información en Big Data” supported by Agreement No. 180 November 1st, 2016 by VIPRI from Universidad de Nariño. As well, authors thank the valuable support given by the SDAS Research Group (www.sdas-group.com).

References

1. Barzegar, R., Asghari Moghaddam, A., Adamowski, J., Ozga-Zielinski, B.: Multi-step water quality forecasting using a boosting ensemble multi-wavelet extreme learning machine model. *Stoch. Environ. Res. Risk Assess.* **32**(3), 799–813 (2018). <https://doi.org/10.1007/s00477-017-1394-z>
2. Becerra, M.A., Orrego, D.A., Mejia, C., Delgado-Trejos, E.: Stochastic analysis and classification of 4-area cardiac auscultation signals using Empirical Mode Decomposition and acoustic features. *Comput. Cardiol. (CinC)* **2012**(September), 529–532 (2012)
3. Da Silva Luz, E.J., Moreira, G.J., Oliveira, L.S., Schwartz, W.R., Menotti, D.: Learning deep off-the-person heart biometrics representations. *IEEE Trans. Inf. Forensics Secur.* **13**(5), 1258–1270 (2018). <https://doi.org/10.1109/TIFS.2017.2784362>
4. Dar, M.N., Akram, M.U., Shaukat, A., Khan, M.A.: ECG based biometric identification for population with normal and cardiac anomalies using hybrid HRV and DWT features. In: 2015 5th International Conference on IT Convergence and Security, ICITCS 2015 - Proceedings (2015). <https://doi.org/10.1109/ICITCS.2015.7292977>
5. Elhoseny, M., Essa, E., Elkhatib, A., Hassanien, A.E., Hamad, A.: Cascade multimodal biometric system using fingerprint and iris patterns. In: Hassanien, A.E., Shaalan, K., Gaber, T., Tolba, M.F. (eds.) AISI 2017. AISC, vol. 639, pp. 590–599. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-64861-3_55
6. Hejazi, M., Al-Haddad, S.A., Hashim, S.J., Aziz, A.F.A., Singh, Y.P.: Feature level fusion for biometric verification with two-lead ECG signals. In: Proceeding - 2016 IEEE 12th International Colloquium on Signal Processing and its Applications, CSPA 2016, March, pp. 54–59 (2016). <https://doi.org/10.1109/CSPA.2016.7515803>
7. Hejazi, M., Al-Haddad, S.A., Singh, Y.P., Hashim, S.J., Abdul Aziz, A.F.: ECG biometric authentication based on non-fiducial approach using kernel methods. *Digit. Signal Process.: Rev. J.* **52**, 72–86 (2016). <https://doi.org/10.1016/j.dsp.2016.02.008>

8. da Silva, H.P., Carreiras, C., Lourenco, A., Fred, A., das Neves, R.C., Ferreira, R.: Off-the-person electrocardiography: performance assessment and clinical correlation. *Health Technol.* **4**, 309–318 (2015)
9. Jimenez, J., Becerra, M., Delgado-Trejos, E.: Heart murmur detection using ensemble empirical mode decomposition and derivations of the mel-frequency cepstral coefficients on 4-area phonocardiographic signals. In: *Computing in Cardiology*, vol. 41 (2014)
10. Jung, W.H., Lee, S.G.: ECG identification based on non-fiducial feature extraction using window removal method. *Appl. Sci.* **7**(12), 1205 (2017). <https://doi.org/10.3390/app7111205>. <http://www.mdpi.com/2076-3417/7/11/1205>
11. Kanchan, T., Krishan, K.: Loss of fingerprints: forensic implications. *Egypt. J. Forensic Sci.* **8**(1), 19 (2018). <https://doi.org/10.1186/s41935-018-0051-0>
12. Martinez-Diaz, M., Fierrez, J., Galbally, J., Ortega-Garcia, J.: An evaluation of indirect attacks and countermeasures in fingerprint verification systems. *Pattern Recognit. Lett.* **32**(12), 1643–1651 (2011). <https://doi.org/10.1016/J.PATREC.2011.04.005>
13. Moreno-Revelo, M., Ortega-Adarme, M., Peluffo-Ordoñez, D.H., Alvarez-Uribe, K.C., Becerra, M.A.: Comparison among physiological signals for biometric identification. In: Yin, H., et al. (eds.) *IDEAL 2017*. LNCS, vol. 10585, pp. 436–443. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68935-7_47
14. Murillo-Escobar, M., Cruz-Hernández, C., Abundiz-Pérez, F., López-Gutiérrez, R.: A robust embedded biometric authentication system based on fingerprint and chaotic encryption. *Expert. Syst. Appl.* **42**(21), 8198–8211 (2015). <https://doi.org/10.1016/j.eswa.2015.06.035>
15. Odinaka, I., Lai, P.H., Kaplan, A.D., O’Sullivan, J.A., Sirevaag, E.J., Rohrbaugh, J.W.: ECG biometric recognition: a comparative analysis (2012). <https://doi.org/10.1109/TIFS.2012.2215324>
16. Pal, S., Mitra, M.: Increasing the accuracy of ECG based biometric analysis by data modelling. *Measurement* **45**(7), 1927–1932 (2012). <https://doi.org/10.1016/J.MEASUREMENT.2012.03.005>
17. Patro, K., Kumar, P.: Machine learning classification approaches for biometric recognition system using ECG signals. *J. Eng. Sci. Technol. Rev.* **10**(6), 1–8 (2017). <https://doi.org/10.25103/jestr.106.01>
18. Pinto, J., Cardoso, J., Lourenço, A., Carreiras, C.: Towards a continuous biometric system based on ECG signals acquired on the steering wheel. *Sensors* **17**(10), 2228 (2017). <https://doi.org/10.3390/s17102228>. <http://www.mdpi.com/1424-8220/17/10/2228>
19. Sidek, K.A., Khalil, I., Jelinek, H.F.: ECG biometric with abnormal cardiac conditions in remote monitoring system. *IEEE Trans. Syst., Man, Cybern.: Syst.* **44**(11), 1498–1509 (2014). <https://doi.org/10.1109/TSMC.2014.2336842>
20. Song, W., Kim, T., Kim, H.C., Choi, J.H., Kong, H.J., Lee, S.R.: A finger-vein verification system using mean curvature. *Pattern Recognit. Lett.* **32**(11), 1541–1547 (2011)
21. Tan, R., Perkowski, M.: Toward improving electrocardiogram (ECG) biometric verification using mobile sensors: a two-stage classifier approach. *Sensors* **17**(2), 410 (2017)
22. Zapata, J.C., Duque, C.M., Rojas-Idarraga, Y., Gonzalez, M.E., Guzmán, J.A., Becerra Botero, M.A.: Data fusion applied to biometric identification – a review. In: Solano, A., Ordoñez, H. (eds.) *CCC 2017*. CCIS, vol. 735, pp. 721–733. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66562-7_51

23. Zhang, Y., Wu, J.: Practical human authentication method based on piecewise corrected Electrocardiogram. In: Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS (61571268), pp. 300–303 (2017). <https://doi.org/10.1109/ICSESS.2016.7883071>
24. Zhang, Z., Li, Y., Jin, S., Zhang, Z., Wang, H., Qi, L., Zhou, R.: Modulation signal recognition based on information entropy and ensemble learning. *Entropy* **20**(3), 198 (2018)



Improving the Decision Support in Diagnostic Systems Using Classifier Probability Calibration

Xiaowei Kortum¹(✉), Lorenz Grigull², Urs Muecke², Werner Lechner³,
and Frank Klawonn^{1,4}

¹ Department of Computer Science, Ostfalia University of Applied Sciences,
Salzdahlumer Str. 46/48, 38302 Wolfenbuettel, Germany
{x.kortum,f.klawonn}@ostfalia.de

² Department of Paediatric Haematology and Oncology, Medical University Hanover,
Carl-Neuberg Str.1, 30625 Hannover, Germany
{grigull.lorenz,urs.muecke}@mh-hannover.de

³ Improved Medical Diagnostics IMD GmbH,
Ostfeldstr. 25, 30559 Hannover, Germany
werner.lechner@improvedmedicaldiagnostics.com

⁴ Helmholtz Center for Infection Research,
Inhoffenstrasse 7, 38124 Braunschweig, Germany
Frank.Klawonn@helmholtz-hzi.de

Abstract. In modern medical diagnoses, classifying a patient's disease is often realized with the help of a system-aided symptoms interpreter. Most of these systems rely on supervised learning algorithms, which can statistically extend the doctor's logic capabilities for interpreting and examining symptoms, thus supporting the doctor to find the correct diagnosis. Besides, these algorithms compute classifier scores and class labels that are used to statistically characterize the system's confidence level on a patient's type of disease. Unfortunately, most classifier scores are based on an arbitrary scale but not uniformed, thus the interpretations often lack of clinical significance and evaluation criterion. Especially combining multiple classifier scores within a diagnostic system, it is essential to apply a calibration process to make the different scores comparable.

As a frequently used calibration technique, we adapted isotonic regression for our medical diagnostic support system, to provide a flexible and effective scaling process that consequently calibrates the arbitrary scales of classifiers' scores. In a comparative evaluation, we show that our disease diagnostic system with isotonic regression can actively improve the diagnostic result based on an ensemble of classifiers, also effectively remove outliers from data, thus optimize the decision support system to obtain better diagnostic results.

Keywords: Classifier calibration
Isotonic regression · Pool adjacent violators
Multiple-classifier system · Statistical computing

1 Introduction

Compared with the common disease appearances, diagnosing rare types based on a patient's symptoms is hard to achieve. In particular, when multiple influencing factors need to be taken into account, overlooked or inaccurately interpreted symptoms of a rare disease are common. Even when the preliminary examination for a patient's physical condition and laboratory tests have been completed for delineating the range of possible diseases, potential lack of awareness and data resources make rare diseases still hard to be identified for many medical doctors. De facto, classifying rare diseases involves a sophisticated process which can take years from early symptom appearance to a final diagnosis. Solutions like a system-assisted diagnosis for improved interpretation of signs through collective patient records are a frequently requested strategy by medical institutions.

A collaborative research project between scientific researchers and medical experts from Hannover Medical School (MHH) has been initiated. The original idea of this cooperation is based on realizing a computer-aided diagnostic system that supports medical doctors' classification capabilities on new patients with symptoms for specific types of rare diseases. For effectively collecting patient data, MHH designed questionnaires for particular disease groups through interviews, investigations, and observations of patients that have already been diagnosed. Such questionnaires cover several fields with the focus of significant symptoms and binary classification as doctor mentioned result. The combination of question & answer pattern provides strong evidence for a particular disease.

In previous publishings of our study, we could manifest a fusion classifier diagnostic system [7] that unites and channelizes the qualitative benefits of single classifier, i.e. support vector machines (SVM) [2], linear discriminant analysis (LDA) [12], logistic regression (LR) [12] and random forests (RF) [8]. The core of this system relies on methods that focus on assisting medical diagnostics [9], especially the previous established fusion method takes advantages of each classifier through evaluating their compatibility and accuracy of each derived diagnosis. Experimental approaches revealed that the fusion recognition, as a combination of various classifiers, presents a definite performance improvement rather than any single classifier [13].

Each classifier provides probabilities for the considered diseases. Some of the classifiers predict well-calibrated probabilities because they do not have biases. Some maximum margin algorithms such as SVM tend to push predicted probabilities away from 0 and 1, which brings characteristic sigmoid shaped distortion [3]. Other models such as Naive Bayes rely on unrealistic independence assumptions and tend to push probabilities closer to 0 and 1. Based on different classifiers have inequable properties, it is essential to narrow the gap of classifiers' scales and consider well-calibrated probabilities while ensemble multiple classifiers [11].

Isotonic regression is a powerful calibration method that can help in correcting monotonic distortions and rescale classifier probabilities into the same range, as the example shows in Fig. 1. The calibration module allows us to obtain better comparable probabilities of a given model, therefore enable a more meaningful combination of classifier outputs. Which makes it easier for the decision

support system to rank examples in order of class-membership likelihood and find a more accurate probability of a new patient that belongs to a particular class, thus reach a higher accuracy of decision making [15]. Another reason to do a calibration on classifier scores is that classes are often unbalanced. It is helpful to harmonize the data by introducing bias to underrepresented classes [11].

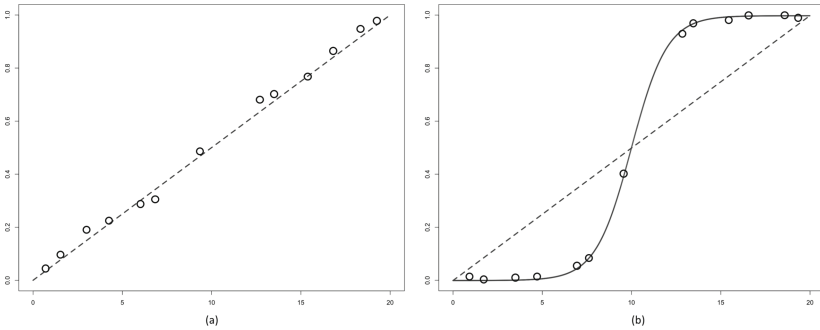


Fig. 1. Original probability (a) vs. Calibrated probability (b)

2 Related Work

This work relies on previous achievements and related studies with particular focus on multiple classifier systems, and best practice for the system-based decision improvements using probability scaling. The following concepts were methodologically adapted for the targeted medical diagnostic environment.

Zadrozny et al. [15] introduced the Pool Adjacent Violators Algorithm (PAVA) that enables the calibration of multi-class probability estimates in applying a simple ranked mapping methodology on classifier scores. The probability is an essential factor which represents the confidence level on the predicted outcome.

Niculescu-Mizil et al. [11] examine the probabilities predicted by ten supervised learning algorithms and the effectiveness of isotonic regression for calibrating the predictions made by different learning methods. It is shown that after calibration, most classifier models can predict better probability estimates.

Kortum et al. [7] investigated the benefits of using the classifier fusion method for diagnosing rare diseases in practice. Experimental results show that this strategy dramatically improves the accuracy of the system compared to any single classifier. An auxiliary tool for physicians was implemented that could derive computer-aided diagnoses in comparing a new patient's symptoms with classified records from a shared patient database.

Chen et al. [3] further investigate the characteristic problem of classifier scores in medical diagnostic approaches. The authors proved that classifier scores on an arbitrary scale could be converted to the probability scale for a target population prevalence value without affecting discrimination performance. This result takes

an essential role in this paper since the dedicated probability problem addresses an interpretability gap, which solves a potential risk in classification.

3 Diagnostic Process for New Patient Data

In this chapter, we describe the classification improvement process for our computer aided diagnosis system that was originally revealed in our previous study stages [6, 7]. Figure 2 provides a compact structure overview about the decision architecture when classifying a new patient’s symptoms while considering probability calibration methods, e.g. the isotonic regression.

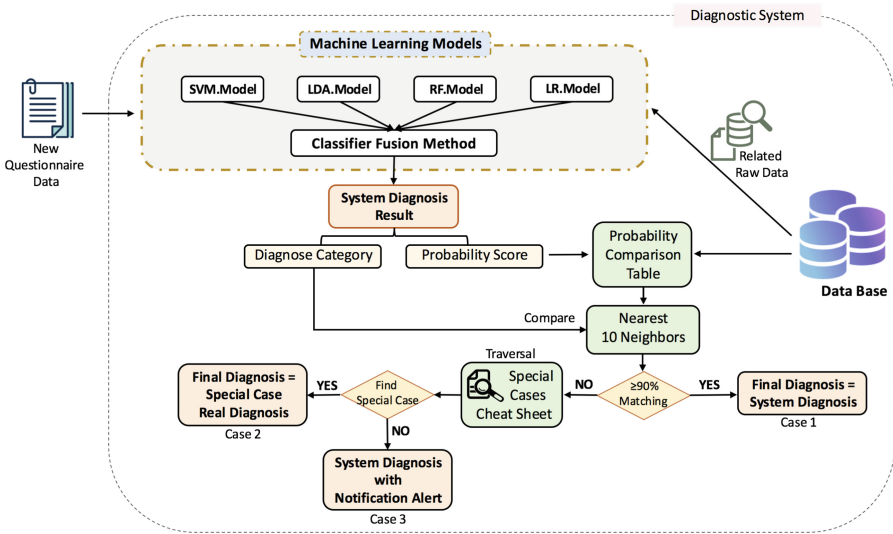


Fig. 2. System diagnosis process for a new patient

When a filled in questionnaire of a new patient is available for diagnosis, the system’s first step is to extract the questionnaire related raw data from the database as the foundation of classification models. The classifier fusion method involves four algorithms (Support Vector Machines; Linear Discriminate Analysis; Random Forest and Logistic Regression) as a combination of multiple classifiers with robust prediction capabilities on medical data records in supervised modeling approaches [7]. After training each classification model by raw data, the system proceeds in predicting the probability of new coming data from each classifier. The fusion method is applied to obtain the diagnostic result due to the highest corresponding probability score, denoted as P .

Then the acquired P and its associated classification result will be compared within the range of its closest K -neighbors in the Probability Comparison Table

(described in Sect. 3.3), K depends on the dataset size and the degree of dispersion of data within this interval. The comparison derives how many real doctors' decisions in a similar case are matching the obtained system diagnostic result. If the prediction presents a strong diagnosis probability ($\geq 90\%$ matched), the system decision will be selected as the final result. In case of a weak prediction value ($< 90\%$), the new patient symptoms data will be sequentially compared with the special case cheat sheet to check if there are any similar answer patterns in the database records with a firm diagnosis. A decisive matching (Case 2) will lead to the final diagnose according to the real doctors' diagnosis of the matched cases. If it does not find any matches in the cheat sheet (Case 3), a prompt message will be given along with the system diagnosis, indicating which variable affects the system to make the correct diagnosis because of insufficient evidence, thus helps doctors to examine the related symptoms tententiously. Once the patient' diagnostic has been confirmed, his significant rare data will be collected and backing-up for documentation and discussion purposes. Compared with the initial diagnosis system, the subsequent screening process presents an additional improvement in filtering different cases between strong statistical predictions and poor predictions. In this way, it is clearly indicated how reliable the diagnostic proposal of the system is.

3.1 Classifier Fusion Method

The concept of applying multiple classifiers relies on the idea that the fusion method takes advantages of each classifier's outcome through evaluating their compatibility and accuracy, to derive the optimal computer-assisted diagnosis. Our previous studies proved that classifiers ensemble for symptoms interpretation, especially the combination of different supervised learning algorithms as shown in Fig. 2, present strong performance improvements compared with any single classifier [7, 11].

The diagnostic process assigns a new dataset of a patient's disease symptoms to the trained classification patterns. The classifier fusion method applied in this study derives the average score from four supervised classifier scores $P(average)$. Each represents the likelihood for a particular type of disease d within the continuous range from 0 to 1. Further, each system-aided diagnosis d obtains the average probability value $P(d)$ as a representative indicator of a particular class. The diagnostic class from the highest $P(average)$ becomes selected as the diagnosis outcome. By evaluating the compatibility and accuracy of individual classifiers, the fusion method takes advantages of each single classifier [9]. In case that one classifier is more particular about a specific diagnosis, it will occupy a more significant proportion within the fusion method. This allows a medical practitioner to derive more accurate diagnosis in comparing a new patients symptoms with the exact diagnostic records from the related dataset.

3.2 Isotonic Regression for Calibrating Classifier Scores

The classifier fusion model performs its system-based diagnoses through statistical classifiers, thus the outcome can generally be found within a limited scale. Toward that, the medical diagnostic meaning of classified decision scores is not always easy to interpret [15]. Modeling that involves the concept of calibration provides advantages when transferring derived classifier scores to a more meaningful probability space. The diagnosis of a disease is more reliable when the doctor can directly interpret the prediction score.

Therefore, the unnormalized scores produced from classifiers need to be calibrated to score-conditional probabilities, to supply reliable performance measures for generalization in the medical field [3]. For example, a patient's classified diagnosis score of 0.80 presents a high likely indication for a particular disease – according to the average of all previously experienced patients. However, a patient with a classifier score of 0.45, on the other hand, does not clearly distinguish between types of diseases without straight comparison to other classified patient records. This leads to the concern about the discrimination and calibration of probabilities within multi-classifier systems [5].

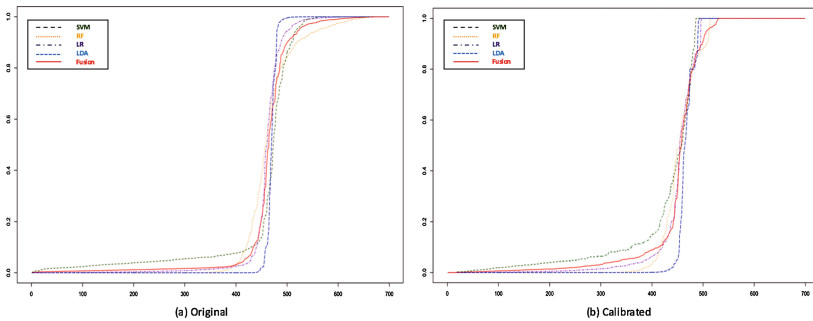


Fig. 3. Original classifier scores vs. calibrated scores - dataset: BC

Fig. 3 shows the difference between original classifier scores (a) and calibrated classifier scores (b), by using the Wisconsin breast cancer dataset (*BC*) from the UCI repository [4]. The calibrated classifier scores (b) tend to interpret the probability more closely as binary scoring without manipulating the models' posterior probabilities. This can be improved with the help of an optimal threshold selection for binning the scores. The initial maximum margin of methods such as SVM push the posterior probability away from 0 and 1 while techniques such as LDA tend to push the probability towards 0 and 1. Methods like RF and SVM perform much better after applying isotonic regression for calibration, which helps to transform classifier scores into meaningful probabilities. Isotonic regression is a non-parametric technique that does not make any assumptions such as linearity among variables and constant error variance [14].

3.3 Integrated Probability Comparison Table

The Probability Comparison Table can be obtained by applying 10-Fold-Cross-Validation to traverse the entire raw data. As shown in Fig. 4, the operation process is divided into three steps: Classifier Model Training; Calibration Model Training and Fuse Prediction Scores.

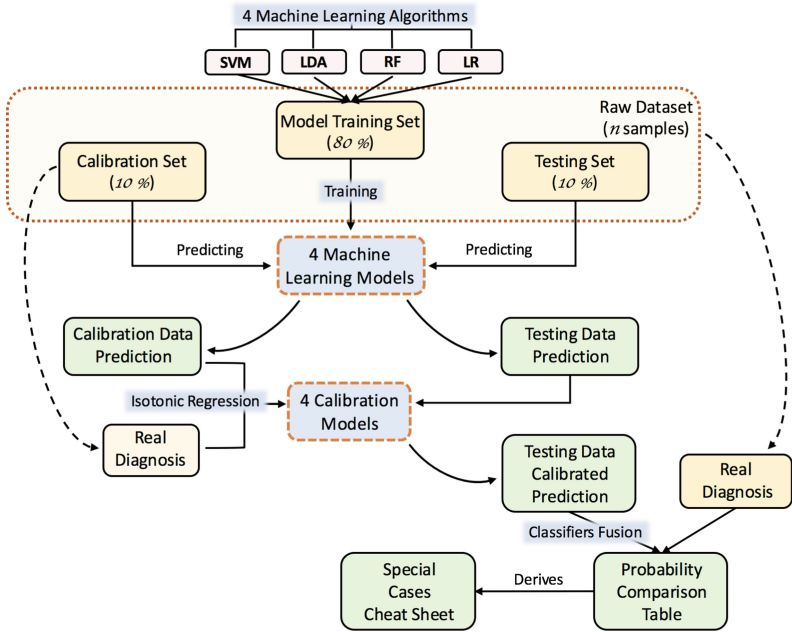


Fig. 4. Fusion classifier calibration for minimized classification errors

In each cross-validation iteration, the applied dataset with one column's binary classification will be partitioned into three parts: Model Training Set (80%); Calibration Set(10%) and Testing Set (10%). As the first step, 80% of the data records are used for training the classification models. The calibration model needs two elements to train: a prediction score and a corresponding real diagnosis. Thence before training the calibration model, the Calibration Set without diagnosis column will go through the model prediction process to obtain the corresponding predictive value. By using the isotonic regression along with the previously detached diagnosis column, the regular prediction values are used to train the calibration models. Probability calibration methods take commonly place for scaling each classification scores' range into a directly-viewed and easy to understand prevalence rate [15]. This step is analogous to the applied testing, the alternative testing data prediction, except that its probability outcome has improved statistical characteristics due to the applied isotonic regression.

As a testing set that separates the doctors' original diagnosis column beside, it needs to be applied to four trained classification models and four calibration models to get calibrated prediction probabilities finally. After applying the fusion method [6,7], the Probability Comparison Table will be generated, for later comparison and examination with the final system diagnosis. The table contains each questionnaire's original classifier scores, corresponding calibrated probability estimation, and real doctors' diagnoses of similar patient cases.

4 Evaluation of Disease Diagnostic System

We applied leave-one-out cross-validation (LOOCV) [1] to evaluate how well the improved model in Fig. 2 performs on new data, by determining the diagnostic accuracy for patients correctly interpreted disease symptoms. The improved system will be compared with the initial system by their qualitative overall accuracy. When it comes to supervised learning algorithms, LOOCV presents a common strategy to quantify the system's decisional accuracy. We evaluated the system with three different datasets:

- Breast cancer dataset (*BC*) with 699 records [4]
- Primary immunodeficiency disorders dataset (*PID*) with 126 records [10]
- Rare disease dataset (*RD*) with 1021 records [6]

All the data records consist of the same characteristic: multiple attribute columns that describe the patients' symptoms condition and a binary diagnosis column given by a real doctor. The data we use in this paper are from patients that have been individually examined, tested and diagnosed by medical doctors.

The procedure of LOOCV assesses our model quality iteratively, by removing one sample from the dataset and use it as testing data. The remaining samples are then used for training the model and predict the diagnosis category of the sample data that have been left out. After all data records have been sequentially extracted and diagnosed by the system, the model's overall accuracy can be calculated by the percentage of correct diagnoses compared with real doctors' decisions. Table 1 shows the system accuracy for the three different datasets. The second column lists the systems overall accuracy. The last column, which is the original diagnostic system accuracy for a classifier fusion without the use of isotonic regression; it can be seen that our strategy (shows in Fig. 2) can significantly enhance the overall accuracy of the decision support system.

The improvement for diagnosing breast cancer increases 1%, foremost because the initial system through the fusion classifier could already make a reliable diagnosis accuracy of 96.1%. The other two datasets achieve a better improvement with 3.6% for the primary immunodeficiency disorders recognition and 3.1% for rare disease diagnosis. Experimental result demonstrates that the designed system can improve the overall accuracy of disease diagnostics, thus provide influential decision support for doctors' diagnosis. In addition to the overall accuracy, another critical information is system verification represented

Table 1. Disease diagnostic system overall accuracy for three datasets

Data set	Improved system accuracy	Case 1 accuracy	Caes 2 accuracy	Case 3 accuracy	Original system accuracy
BC	97.1%	98.6%	100%	92.7%	96.1%
PID	88.9%	95.2%	99.8%	71.7%	85.3%
RD	86.4%	93.5%	99.1%	66.6%	83.3%

in three different cases. Case 1 ensures the high prediction accuracy by comparing the system diagnosis results with the doctors' decision within similar cases, which required to achieve at least 90% matches. In the second case, the diagnosis accuracy could reach 99%, reveals that the diagnostic result of some patients that have a relatively weak matching degree in case 1, could be adjusted by matching their record with particular response patterns in which the system cannot make an accurate decision (records into cheat sheet). Due to the similar answer patterns as in the reference records, the patient's diagnosis result will follow the doctors' suggestions, which are substantiated correct in most case. Mainly, if there is a one-to-one correspondence between a new patient's answer pattern and the special case in the cheat sheet, the system will ensure that the final diagnostic results are consistent with the doctor's decision stored in the database, regardless of the system diagnosis. Case 3 mainly collects data records that the system cannot diagnose accurately. It is possible that all classifiers point to one diagnosis category, but the reality is poles apart. In this case, the best way is to mark the variables that influence the system diagnosis and send notifications to provide constructive suggestions to doctors for examining related symptoms. Once the doctor gets confirmation of patient' diagnostic result, such significant data record will be absorbed into our cheat sheet, for more in-depth analysis and discussion purposes.

5 Conclusion and Future Perspectives

Motivated by continuously improving the disease diagnostic system, we embedded the calibration procedure into the classifiers fusion method. The conventional classification model can only interpret patients' diseases within each machine learning algorithm. But the newly integrated calibration adaption based on isotonic regression offers the possibility of cross-reference classifiers' results in the same interval, thereby derive diagnoses through three degrees of qualified outcomes (firm statistical determination, answer pattern matching and handling significant rare data record). We validated our diagnostic system by using three datasets covering 1846 records that include multiple diseases symptoms description and real doctors' diagnoses. We compared the overall accuracy of the original fusion classifier model and the calibrated classification, proved a new strategy that can improve system performance. The system can support doctors to derive

the correct diagnosis. More importantly, it can filter out the poor diagnostic outcome. We believe that the fusion classifier calibration and its case differentiation could help researchers and medical doctors to improve their patients' symptoms interpretation, also in other data analysis areas. It presents an improved measure for adequate diagnosis, especially for patients with symptom tendencies where the decidability between two types of diseases is not explicitly apparent.

References

1. Arlot, S., Celisse, A.: A survey of cross-validation procedures for model selection. *Dev. Appl. Stat.* **4**(2010). 4079, 159–177 (2003)
2. Auria, L., Moro, R.A.: Support vector machines (SVM) as a technique for solvency analysis (2008)
3. Chen, W., Sahiner, B., Samuelson, F., Pezeshk, A., Petrick, N.: Calibration of medical diagnostic classifier scores to the probability of disease. *Stat. Methods Med. Res.* **27**(5), 1394–1409 (2018)
4. Wolberg, W.H., Street, W.N., Mangasarian, O.L.: UCI machine learning repository: breast cancer wisconsin (1995). <http://archive.ics.uci.edu/ml/datasets>
5. Schmid, C.H., Griffith, J.L.: *Multivariate Classification Rules: Calibration and Discrimination*. American Cancer Society (2005)
6. Kortum, X., Grigull, L., Lechner, W., Klawonn, F.: A dynamic adaptive questionnaire for improved disease diagnostics. In: Adams, N., Tucker, A., Weston, D. (eds.) *IDA 2017. LNCS*, vol. 10584, pp. 162–172. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68765-0_14
7. Kortum, X., Grigull, L., Muecke, U., Lechner, W., Klawonn, F.: Diagnosis support for orphan diseases: a case study using a classifier fusion method. In: Yin, H., Gao, Y., Li, B., Zhang, D., Yang, M., Li, Y., Klawonn, F., Tallón-Ballesteros, A.J. (eds.) *IDEAL 2016. LNCS*, vol. 9937, pp. 379–385. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46257-8_41
8. Liaw, A., Wiener, M.: Classification and regression by random forest. *R News* **2**(3), 18–22 (2002)
9. Ma, L., Liu, X., Song, L., Zhou, C., Zhao, X., Zhao, Y.: A new classifier fusion method based on historical and on-line classification reliability for recognizing common ct imaging signs of lung diseases. *Comput. Med. Imaging Graph.* **40**, 39–48 (2015)
10. Mücke, U., et al.: Patients experience in pediatric primary immunodeficiency disorders: computerized classification of questionnaires. *Front. Immunol.* **8**, 384 (2017)
11. Niculescu-Mizil, A., Caruana, R.: Predicting good probabilities with supervised learning. In: *Proceedings of the 22nd International Conference on Machine Learning*, pp. 625–632. ACM (2005)
12. Pohar, M., Blas, M., Turk, S.: Comparison of logistic regression and linear discriminant analysis: a simulation study. *Metod. Zv.* **1**(1), 143 (2004)
13. Sboner, A., et al.: A multiple classifier system for early melanoma diagnosis. *Artif. Intell. Med.* **27**(1), 29–44 (2003)
14. Stout, Q.F.: Isotonic regression algorithms. Accessed 6 Aug 2011
15. Zadrozny, B., Elkan, C.: Transforming classifier scores into accurate multiclass probability estimates. In: *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 694–699. ACM (2002)



Applying Tree Ensemble to Detect Anomalies in Real-World Water Composition Dataset

Minh Nguyen^(✉) and Doina Logofătu

Department of Computer Science and Engineering,
Frankfurt University of Applied Sciences, 60318 Frankfurt, Germany
mhuy@stud.fra-uas.de

Abstract. Drinking water is one of fundamental human needs. During delivery in distribution network, drinking water is susceptible to contaminants. Early recognition of changes in water quality is essential in the provision of clean and safe drinking water. For this purpose, Contamination warning system (CWS) composed of sensors, central database and event detection system (EDS) has been developed. Conventionally, EDS employs time series analysis and domain knowledge for automated detection. This paper proposes a general data driven approach to construct an automated online event detention system for drinking water. Various tree ensemble models are investigated in application to real-world water quality data. In particular, gradient boosting methods are shown to overcome challenges in time series data imbalanced class and collinearity and yield satisfied predictive performance.

Keywords: Tree ensemble · Gradient boosting · Random forest
Anomaly detection · Time series · Water quality · Contamination
Class imbalance

1 Introduction

Water is vital for all known forms of life and for the growth and development of human civilization. The provision of clean and safe drinking water is a necessity and a challenge for countries around the world. Drinking water supply and its distribution network are highly sensible to any kinds of contaminations. Water supply companies need to frequently monitor water and environmental data. In the modern days, these data are collected by highly sensible sensors and analyzed to detect any kinds of anomalies. Early recognition of changes in water quality enables water supply companies to counteract in time. For this purpose, contamination warning system (CWS) has been developed. The system composes of multiple sensor station, a Supervisory Control and Data Acquisition (SCADA) central database and an event detection system (EDS) [1].

In the recent years, machine learning methods have shown remarkable performance in different time series prediction and analysis problem [2] and have driven

the operation of many automated system. This paper proposes an automated event detection system that detects the contamination event from variation in real-world water composition data collected from surrogate sensors and operational data. The system is driven by various tree ensemble methods that are tolerant to collinearity and class imbalance. The empirical performance of these ensemble models is evaluated and presented.

2 Related Work

The use of surrogate sensor to collect water quality parameter for contamination detection has been reported in [3,4]. Parameters like Turbidity, pH and Chlorine were found to be effective indicators of water contamination. Motivated by these findings, a range of anomalous contamination event detection techniques has been developed over the year. The work in [3] uses the limit of detection of 3 standard deviation interval. Time increment, linear filters (generally known as autoregressive (AR) moving average) and multivariate nearest neighbor (MV-NN) which are studied in [5], detect anomalies by taking the past and current measurements into account. Real-time event adaptive detection, identification and warning (READiw) methodology proposed in [6] recognize anomalies from the adaptively back-tracked background sensors data in a moving time window. Probabilistic approach [7] applies AR model to estimate future water quality parameters whose residuals are assigned with probabilities. The anomalous probabilities of residual are searched using Dempster-Schafer fusion. Similar approach proposed in [8] utilizes Artificial Neural Network (ANN) for parameters estimation and Bayesian sequential analysis for probabilities update. Supervised machine learning techniques like Logistic Regression, Linear Discriminant Analysis, ANN, Support Vector Machine (SVM) are applied in [9] to drinking water quality event detection. An exhaustive list of different machine learning and big data techniques suggested in various systems of water contamination detection is compared in the survey [10]. Some of the studied methods are Fast fuzzy C-mean clustering, GIS with Ant Colony algorithm, Radial Basis Network Function, ANN, Least square SVM. On a general note, tree boosting technique has been shown to give state-of-the-art results on many standard classification benchmarks [11] and has been implemented in real-world production pipeline [12]. Inspired by these studies, we think tree boosting is applicable to the present classification problem of drinking water contamination.

3 Problem Description

The dataset is provided by the Thüringer Fernwasserversorgung (TFW) and is made publicly available via GECCO industrial challenge 2018 [13]. The objective is to develop an online system that monitors water quality and operational status to detect remarkable changes in water quality, or contamination events. The system shall have reasonable response time to online input water data, output whether a remarkable change occur at a single point of time.

For the monitoring of the water quality, measurements are collected at significant points throughout the whole water distribution system, in particular at the outflow of the waterworks and the in- and outflow of the water towers. A part of the water is bypassed through a sensor system located at different stations near the outflow of a waterworks, where the most important water quality indicators and operational data are measured. For detailed description of the sensors data, refer to Table 1.

The target variable indicates the contamination caused by regular checks and replacement of sensors. Fig. 1 shows a snapshot of time series with such events marked in red. As a part of EDS Evaluation Process, simulated theoretical contamination events are also added into the data [1, 13], as shown in Fig. 2. Both kind of contamination events are marked as variable EVENT.

Table 1. Variables in dataset and their descriptions

Variable	Description
Time	Time of measurement, given in following format: yyyy-mm-dd HH:MM:SS
Tp	The temperature of the water, given in °C
Cl	Amount of chlorine dioxide in the water, given in mg/L (MS1)
pH	pH value of the water
Redox	Redox potential, given in mV
Leit	Electric conductivity of the water, given in μS/cm
Trueb	Turbidity of the water, given in NTU
CL.2	Amount of chlorine dioxide in the water, given in mg/L (MS2)
Fm	Flow rate at water line 1, given in m ³ /h
Fm.2	Flow rate at water line 2, given in m ³ /h
EVENT	Marker if this entry should be considered as a remarkable change resp. event, given in boolean

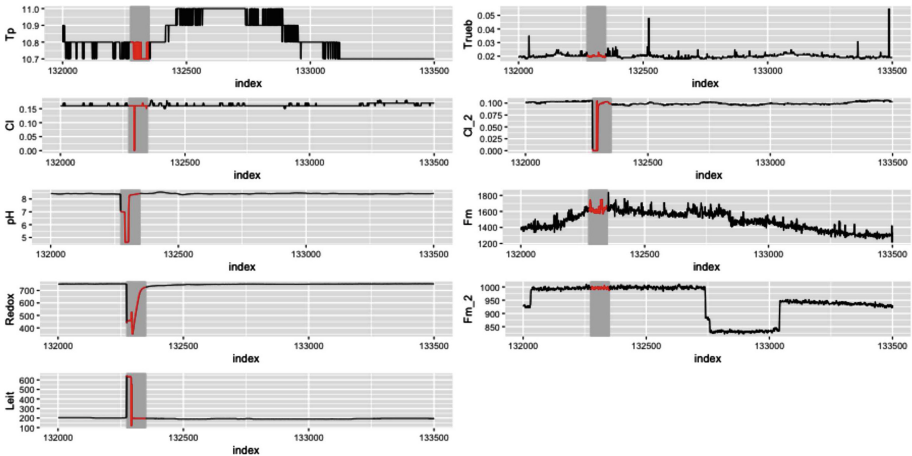


Fig. 1. Time series of sensors data of one day with original event marked in red [13].

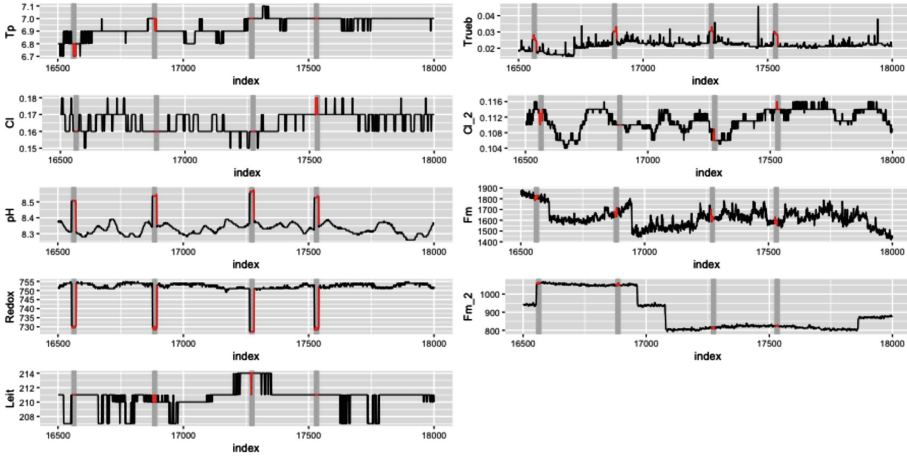


Fig. 2. Time series of sensors data of one day with simulated event marked in red [13].

4 Data Analysis

The problem is a supervised binary classification problem with 10 predictors and 1 target variable. The training data ranges from 03/08/2016 till 08/11/2016, equivalent to 3 months or 97 days. The data is recorded every 1 min, amounts up to 139,566 observations. Temporal trend might have great potential in data analysis and prediction, but the time span of 3 months is not sufficient to extract temporal trend without biases.

The number of missing values is the same for all predictors and is negligible (1045 samples or 0.75%). One can apply one of the imputation techniques studied in [14] to avoid performance degradation. All observations with missing values are labeled False and target variable contains no missing value. Considering these points, missing values are handled by forward fill, i.e. to fill missing values with its preceding values. The ratio of target variable is highly imbalanced with 98.76% False and 1.23% True (1726 samples). High class imbalance can lead to biased evaluation score like accuracy.

The target variable has medium correlated with Redox measurement and small correlation with pH and Cl₂, as shown in Fig. 3. On the other hand, predictors Tp, pH, Redox, Fm have medium to strong pairwise correlations. Collinearity caused by correlated predictors is problematic to many predictive models and deserves extra attention in modeling.

5 Proposed Approach

Our proposed predictive system contains 2 phases: Offline Training phase and Online Prediction phase. In the training phase, several predictive models are fitted and evaluated with cross validation. The model with the best cross-validated

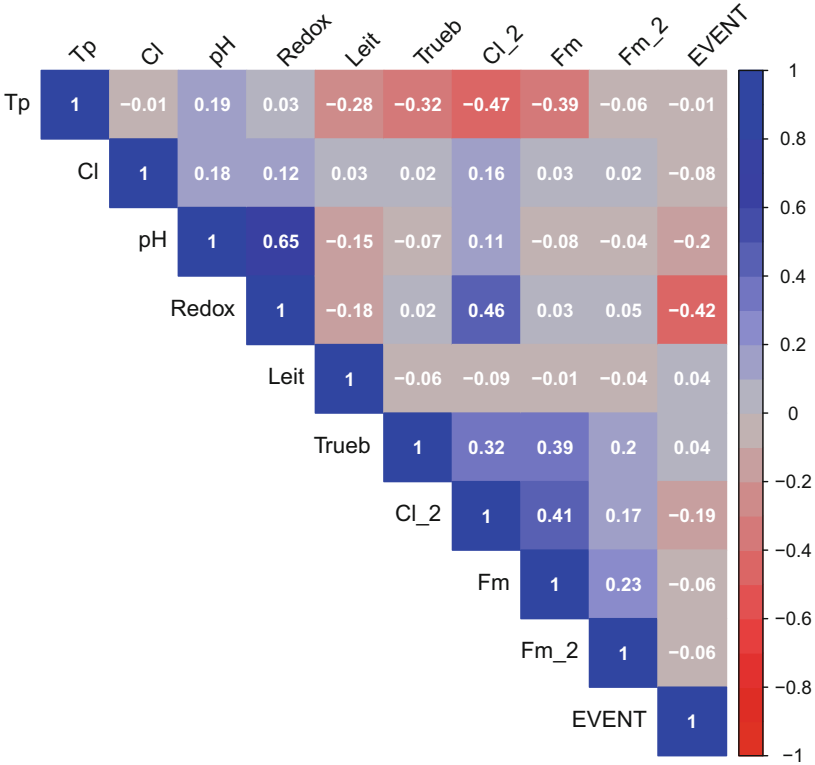


Fig. 3. Pairwise Correlation plot.

score is used in the prediction phase to generate prediction from unseen online data.

The training phase contains the pipeline of Data Preparation, Parameter Tuning and Model Evaluation that trains different tree-based models. Data preparation involves data type conversion, handling missing values and splitting data into train and test set with ratio 8:2. Cross validation is realized with stratified sampling. Parameter Tuning utilizes exhaustive search over the parameter grid, and each set of selected parameters is evaluated with 5-fold cross validation. The metric used to evaluate the learning models is F1-score. F1 is defined as the harmonic mean of recall and precision, thus is consistent in dataset with imbalanced class.

$$F_1 = \frac{2}{\frac{1}{\text{recall}} + \frac{1}{\text{precision}}} \tag{1}$$

We investigate various tree ensemble models as the core learning algorithm. The models are evaluated using cross-validation and their results are presented in Sect. 6. Due to the short time span of the training data, time series analysis

is not effective and can cause temporal bias toward the training data. Thus, to avoid adding unnecessary complexity to the online detector, we will not consider time series analysis techniques for modeling such as Moving average, Fourier Transform, ARIMA, etc.

5.1 Tree Ensemble Model

Tree-based model is essentially ensemble classifier with decision tree as base classifier. The main idea of ensemble learning is to combine many diverse classifiers to obtain a new classifier that outperforms each one of them. Decision tree is preferred as base classifier in ensemble because it is weak learner, where small changes in input data results in large changes in prediction output, thus brings diversity to ensemble and consequently reduce the variance.

There are several techniques to construct ensemble, for instance Bagging [15] and Boosting [16]. Various preprocessing technique that handle imbalanced class are embedded in state of the art ensembles [17]. In addition, ensembles can model complex relationship between predictors and are tolerant to collinearity [18]. Our experiments investigate the performance of the following tree ensembles: Random Forest [19] (RF), Regularized Random Forest [20] (RRF), Extreme Gradient Boosting [21] (xgbTree), Extreme Gradient Boosting with Dropout [22] (xgbDART). Random Forest introduces random sampling of predictors to bagged tree to reduce variance. Extreme Gradient Boosting is a scalable implementation of Gradient Boosting [23].

5.2 Gradient Boosting

Decision tree is a simple classifier, contains a set of rules that decide how to split the feature space into disjoint regions R_j , $j = 1, 2, \dots, N$. Given constant y_i assigned to each region, the prediction rule is $x \in R_j \Rightarrow f(x) = \gamma_j$. According to [24], tree can be formally defined as $T(x; \Theta) = \sum_{j=1}^J \gamma_j I(x \in R_j)$, with tree parameter $\Theta = \{R_j; \gamma_j\}_i^J$. The boosted tree model is the sum of trees $f_M(x) = \sum_{m=1}^M T(x; \Theta_m)$.

Algorithm 1 describes how a generic version of Gradient Boosting works [24], where the specific algorithm is realized by defining the loss function. For K-class classification problems, the multinomial deviance loss function is defined

$$L(y_i, p(x_i)) = - \sum_{k=1}^K I(y_i = G_k) \log p_k(x_i) \quad (2)$$

6 Experimental Results

Tree ensemble models RF, RRF, xgbDART, xgbTree are tuned and cross-validated in the training phase described above. Table 2 shows the results of cross-validation in term of average and standard deviation. Extreme Gradient Boosting

Algorithm 1. Pseudocode for Generic Gradient Tree Boosting algorithm**Input:** Training set $\{(x_i, y_i)\}_{i=1}^N$, Loss function $L(y_i, f(x_i))$, Number of iterations M **Output:** Fitted model $\hat{f}(x)$

1. Initialize model

$$f_0(x) = \arg \sum_{i=1}^N L(y_i, \gamma)$$

2. For
- $m = 1$
- to
- M
- :

- (a) For
- $i = 1, 2, \dots, N$
- compute pseudo residuals

$$r_{im} = - \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right]_{f=f_{m-1}}$$

- (b) Fit a regression tree to the target
- r_{im}
- giving terminal regions
- R_{jm}
- ,
- $j = 1, 2, \dots, N$

- (c) For
- $j = 1, 2, \dots, J_m$
- , where
- J_m
- is the size of each tree, compute

$$\gamma_{jm} = \arg \sum_{x_i \in R_{jm}}^N L(y_i, f_{m-1}(x_i) + \gamma)$$

- (d) Update the model

$$f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$$

3. Output
- $\hat{f}(x) = f_M(x)$

and its variant yield the most optimal cross-validated score and model stability. Model `xgbDART` yields the best F1-score of 0.9238 and standard deviation of 0.06. Model `xgbTree` yields a slightly smaller F1-score of 0.9164 and standard deviation of 0.05.

The recall metric, indicating the true positive rate, i.e. the rate of detecting anomaly correctly, is more prioritized than the precision, because high recall means less anomaly is missed. In Table 3, `xgbDART` achieves highest recall of 0.8852 and standard deviation of 0.07. RF performs slightly better than `xgbTree`, 0.8694 compared to 0.8650. On the other hand, RF has greater variance than `xgbTree`, approximately 0.01 in standard deviation.

Regarding the training time, `xgbTree` is remarkably faster than `xgbDART` (approximately 14 times faster). `xgbTree` implements scalable distributed architecture and is also the fastest among all investigated model. Regularization in Extreme Gradient Boosting and Random Forest slow down the training time. The training time in the offline phase is independent with the execution of the online prediction system, and can be neglected from the model selection criteria.

Feature importance is a metric used in tree-based model to indicate the contribution of each feature to the improvement of the prediction. Random Forest

Table 2. Cross validation results of tree-based models

Model	Train time (s)	F1	F1 SD	Accuracy	Accuracy SD
xgbDART	2616.05	0.9238	0.0656	0.9284	0.0609
xgbTree	186.14	0.9164	0.0536	0.9240	0.0440
RF	659.18	0.8660	0.0444	0.8665	0.0480
RRF	14201.53	0.8245	0.0877	0.8393	0.0749

Table 3. Cross validation results of tree-based models (cont.)

Model	Train time (s)	AUC	AUC SD	Recall	Recall SD
xgbDART	2616.05	0.9279	0.0609	0.8852	0.0721
xgbTree	186.14	0.9233	0.0442	0.8650	0.0895
RF	659.18	0.8662	0.0473	0.8694	0.1022
RRF	14201.53	0.8384	0.0744	0.7758	0.1302

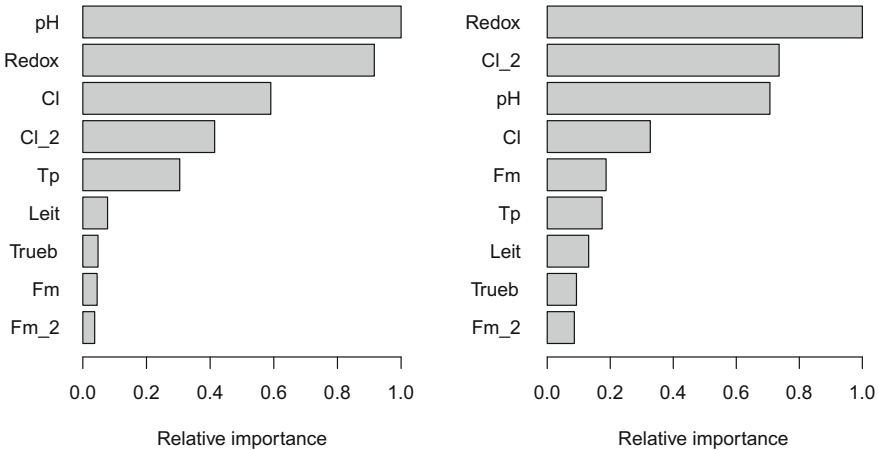


Fig. 4. Relative feature importance of Extreme Gradient Boosting (xgbTree) on the left and Random Forest (RF) on the right

computes feature importance from the decrease in Gini index at each split while Extreme Gradient Boosting feature importance measures the improvement in accuracy when splitting. Feature importance of the fitted xgbTree and RF model are illustrated in Fig. 4, relatively to the most important feature. Top three highest importances are similar for both models, namely pH of water, Redox potential and chlorine dioxide (pH, Redox and Cl for xgbTree and Redox, Cl₂ and pH for RF).

7 Conclusion

The present approach shows that the minimal machine learning workflow consisting of data analysis, learning model selection and proper cross-validation pipeline could yield satisfactory prediction results. Tree ensembles are able to perform well in scenario with imbalanced class and collinearity in the dataset. Extreme Gradient Boosting excels in performance as well as training time. However, the fact that the dataset is limited in size and time range must be taken into consideration when interpreting the results. In case the training data is not exemplary, it is likely that the fitted model is biased toward the training data, thus make it challenging to investigate the generalization capability of the predictive system.

During our work, we found that techniques such as dimensional reduction, unsupervised learning and time series analysis exhibit potential on future improvement of the present tree ensemble approach. In addition, it worth further investigation on data augmentation and the method of adding simulated theoretical values into the present dataset. On top of that, the gist of ensemble methods can be applied to other learning models that was studied in related literature, which is referred to as meta ensemble or model stacking. Meta ensemble leverages the diversity of different base learner to attain better final results. Nevertheless, the more effective solution lies in data collection and data quality assurance. Given a larger dataset, the same workflow with time series analysis integrated could extract valuable temporal feature, decorrelate sensor values and consequently achieve significantly improved performance.

References

1. McKenna, S.A., Hart, D.B., Murray, R., Haxton, T.: Testing and evaluation of water quality event detection algorithms. In: Clark, R.M., Hakim, S., Ostfeld, A. (eds.) *Handbook of Waterand Wastewater Systems Protection*, pp. 369–396. Springer, New York (2011). https://doi.org/10.1007/978-1-4614-0189-6_19
2. Hamilton, J.D.: *Time Series Analysis*. Princeton University Press, Princeton (1994)
3. Byer, D., Carlson, K.H.: Real-time detection of intentional chemical contamination in the distribution system. *J.- Am. Water Work. Assoc.* **97**(7), 130–133 (2005)
4. Hall, J., Szabo, J.: *WaterSentinel Online Water Quality Monitoring as an Indicator of Drinking Water Contamination*. Environmental Protection Agency, Washington, DC, USA (2005)
5. Klise, K.A., McKenna, S.A.: Multivariate applications for detecting anomalous water quality. In: *Water Distribution Systems Analysis Symposium 2006*, Cincinnati, Ohio, United States, pp. 1–11. American Society of Civil Engineers, March 2008
6. Jeffrey Yang, Y., Haught, R.C., Goodrich, J.A.: Real-time contaminant detection and classification in a drinking water pipe using conventional water quality sensors: techniques and experimental results. *J. Environ. Manag.* **90**(8), 2494–2506 (2009)
7. Hou, D., He, H., Huang, P., Zhang, G., Loaiciga, H.: Detection of water-quality contamination events based on multi-sensor fusion using an extended Dempster-Shafer method. *Meas. Sci. Technol.* **24**(5), 055801 (2013)

8. Perelman, L., Arad, J., Housh, M., Ostfeld, A.: Event detection in water distribution systems from multivariate water quality time series. *Environ. Sci. Technol.* **46**(15), 8212–8219 (2012)
9. Muharemi, F., Logofătu, D., Andersson, C., Leon, F.: Approaches to building a detection model for water quality: a case study. In: Sieminski, A., Kozierekiewicz, A., Nunez, M., Ha, Q.T. (eds.) *Modern Approaches for Intelligent Information and Database Systems. SCI*, vol. 769, pp. 173–183. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76081-0_15
10. Kang, G., Gao, J.Z., Xie, G.: Data-driven water quality analysis and prediction: a survey. In: *2017 IEEE Third International Conference on Big Data Computing Service and Applications (BigDataService)*, pp. 224–232, April 2017
11. Li, P.: Robust logitboost and adaptive base class (ABC) logitboost. In: *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence, UAI 2010, Arlington, Virginia, United States*, pp. 302–311. AUAI Press (2010)
12. He, X., et al.: Practical lessons from predicting clicks on ads at Facebook. In: *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising, ADKDD 2014, New York, NY, USA*, pp. 5:1–5:9. ACM (2014)
13. Rehbach, F., Moritz, S., Chandrasekaran, S., Rebolledo, M., Friese, M., Bartz-Beielstein, T.: GECCO 2018 Industrial Challenge, Monitoring of drinking-water quality (2018)
14. Muharemi, F., Logofătu, D., Leon, F.: Review on general techniques and packages for data imputation in R on a real world dataset. In: Nguyen, N.T., Pimenidis, E., Khan, Z., Trawiński, B. (eds.) *ICCCI 2018. LNCS (LNAI)*, vol. 11056, pp. 386–395. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98446-9_36
15. Breiman, L.: Bagging predictors. *Mach. Learn.* **24**(2), 123–140 (1996)
16. Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: a new explanation for the effectiveness of voting methods. *Ann. Stat.* **26**(5), 1651–1686 (1998)
17. Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., Herrera, F.: A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Trans. Syst., Man, Cybern. Part C (Appl. Rev.)* **42**(4), 463–484 (2012)
18. Dormann, C.F., et al.: Collinearity: a review of methods to deal with it and a simulation study evaluating their performance. *Ecography* **36**(1), 27–46 (2013)
19. Breiman, L.: Random forests. *Mach. Learn.* **45**(1), 5–32 (2001)
20. Deng, H., Runger, G.: Gene selection with guided regularized random forest. *Pattern Recognit.* **46**(12), 3483–3489 (2013)
21. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2016, New York, NY, USA*, pp. 785–794. ACM (2016)
22. Rashmi, K., Gilad-Bachrach, R.: Dart: dropouts meet multiple additive regression trees. In: *International Conference on Artificial Intelligence and Statistics*, pp. 489–497 (2015)
23. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. *Ann. Stat.* **29**(5), 1189–1232 (2001)
24. Hastie, T., Tibshirani, R., Friedman, J.: *The Elements of Statistical Learning*. SSS. Springer, New York (2009). <https://doi.org/10.1007/978-0-387-84858-7>



A First Approach to Face Dimensionality Reduction Through Denoising Autoencoders

Francisco J. Pulgar^(✉), Francisco Charte, Antonio J. Rivera,
and María J. del Jesus

Andalusian Research Institute on Data Science and Computational Intelligence (DaSCI), Computer Science Department, University of Jaén, 23071 Jaén, Spain
{fpulgar, fcharte, arivera, mjjesus}@ujaen.es

Abstract. The problem of high dimensionality is a challenge when facing machine learning tasks. A high dimensional space has a negative effect on the predictive performance of many methods, specifically, classification algorithms. There are different proposals that arise to mitigate the effects of this phenomenon. In this sense, models based on deep learning have emerged.

In this work, denoising autoencoders (DAEs) are used to reduce dimensionality. To verify its performance, an experimentation is carried out where the improvement obtained with different types of classifiers is verified. The classification method used are: kNN, SVM, C4.5 and MLP. The test for kNN and SVM show a better predictive performance for all datasets. The executions for C4.5 and MLP reflect improvements only in some cases. The execution time is lower for all tests. In addition, a comparison between DAEs and PCA, a classical method of dimensionality reduction, is performed, obtaining better results with DAEs in most cases. The conclusions reached open up new lines of future work.

Keywords: Classification · Deep learning · Autoencoders
Denoising autoencoders · Dimensionality reduction
High dimensionality

1 Introduction

Since the 20th century, different machine learning techniques have been developed. In particular, the classification task is one of the most well-known problems within automatic learning. A classifier aims to correctly categorize new data patterns. For this, a model is usually built from correctly labelled data. Normally, each instance has a single label associated with it [13].

These methods work with datasets with very different characteristics. Therefore, algorithms are affected in several ways by these features. One of them is the high dimensionality of the data [7]. Throughout history, many proposals that mitigate the effects of this fact have emerged. In this context, new proposals have

appeared due to the rise of Deep Learning (DL) models. This type of algorithms have obtained very good results in different fields of application, such as automatic speech processing or computer vision [6,11]. In particular, autoencoders (AEs) are deep networks that offer good results in the selection of features due to their architecture and operation [8,18].

There are different models of AEs suitable to perform the feature fusion task. Fundamentally, AEs reproduce the input of the network at the exit, through a series of hidden layers. In this way, a coding of the input is achieved in the intermediate layers. One of the types of AEs are denoising AEs (DAEs). The objective of them is learning to generate robust characteristics from the partially altered input data. In this way, the features originated by the AE are much more resistant to corrupted inputs [8,26].

The objective of this study is to verify the improvement of the predictive performance of several classifiers after applying a DAE to reduce the dimensionality of the data. To see the effects on classifiers of different types, the following algorithms are going to be used: kNN [10], SVM [17], C4.5 [24] and MLP [19]. The experimentation will consist of a first phase where the reduction of dimensionality will be applied on 5 datasets and a second phase where classification will be carried out using the previous algorithms. Next, classification results from the data with reduced dimensionality and the original data will be compared. Finally, a comparison between the use of DAEs for dimensionality reduction with respect to other classical method, such as PCA [20,22].

2 Background

Classification is a contextualized task within machine learning. Fundamentally, it is a phase of data mining whose purpose is to predict or categorize a new instance, based on a series of correctly labeled data. Normally, the process is done using supervised learning methods.

Different methodologies have been developed in order to face this work: Instance-based learning (IBL) that does not build a model, but uses the information provided by the training instances directly [1]; Artificial Neural Networks (ANNs) that are inspired by the way the human brain works. ANNs learn to detect relationships and patterns in the data through their own experience [25]; Support Vector Machines (SVMs) that are supervised learning models used for classification and regression. These algorithms map the different instances as points in space, so that the examples of different categories are separated from each other [17]; Decision tree learning using predictive models based on trees. The different features of the data are represented in the branches of the tree while the leaves contain the objective values [23].

The methodologies indicated above are some of the most used to face the task of classification. Therefore, this work includes an algorithm of each type to validate the reduction of dimensionality through DAEs. In particular, the algorithms used are: kNN [10], MLP [19], SVM [17] and C4.5 [24].

The different proposals used to perform classification must take into account the features of the data. Among these factors is the dimensionality of the data.

A dataset with high dimensionality negatively affects the predictive performance of a large part of the classifiers. This is due to the curse of dimensionality [4, 5].

This factor affects many of the classification models and different proposals have emerged to mitigate its effects [3, 14, 28]. The first solutions were manual, where experts decided which variables were the best for the process. However, this task was soon automated and the first methods of feature selection emerged. Some traditional approaches that face this problem are: LDA [27] and PCA [22]. Recently, new models have appeared based on DL that deal with this fact. In particular, AEs offer good results due to their operation and architecture [8, 18].

An AE is an ANN whose goal is to reproduce the input to the output through unsupervised learning. These models present an adequate symmetrical architecture to achieve this purpose. The operation consists of learning to represent the input data in the output layer, but fulfilling a number of requirements to avoid copying the information throughout the network. This type of algorithms have recently gained momentum due to the rise of DL models.

The basic architecture of an AE is a feed-forward neural network where the information always goes in the same direction. These models are formed by a sequence of layers: an input layer, an output layer and a series of hidden layers. The elements of each layer are connected to all the units of the previous layers. A restriction associated with the operation of the AEs is that the number of neurons of the input layer must be the same as that of the output layer. In this way, the AE can learn to represent the input to the output of the network.

As has been said, the original purpose of the AEs is to look for useful representations of the data. For this, the model learns non-linear ways to combine the data features. In this sense, there are different variations of the most basic AE model that allows discovering new characteristics of the data. Thus, the different variants generate models that allow certain restrictions to be satisfied. In this regard, a method is DAE. This type of algorithm introduces noise into the input before beginning the training process. In this way, the generated characteristics are more robust since they are able to tolerate the introduced noise and correctly reconstruct the input [8, 26].

In this study, DAEs are used to address the problem of dimensionality reduction. In Sect. 3, four classification algorithms are used with original high-dimensional data and low-dimensional data after applying DAE to fusion features. In this way, the performance to select characteristics of DAEs is analysed.

3 Experimentation

Once presented the main concepts involved in this study, the experimentation that is carried out is outlined. The fundamental objective of this stage is to analyse the behaviour of different classifiers with data sets where feature fusion is applied through DAEs.

Therefore, the tests performed have two fundamental phases. On the one hand, the fusion of features from the different datasets is done using DAE. On the other hand, the data is classified using 4 different classifiers: C4.5, kNN,

MLP, SVM. For each dataset, two classifications are made: one with the original data set and another the data with low dimensionality.

3.1 Experimental Framework

The conducted experimentation aims to determine the performance of DAEs for the task of dimensionality reduction. Therefore, the use of different types of datasets is necessary. Their traits are in Table 1. The source is shown in the last column. A 2×5 folds cross validation scheme is applied in all executions.

Table 1. Characteristics of the datasets used in the experimentation.

Dataset	Number of Samples	Number of Features	Number of Classes	Type	Ref.
Image	2310	19	7	Real	[2]
isolet	7797	617	26	Real	[9]
madelon	2000	500	2	Real	[16]
mfeat	2000	649	10	Real	[2]
nomao	1970	118	2	Real	[2]
semeion	1593	256	10	Integer	[2]

The parametrization of the different classification algorithms is the one given by default in each method. In the case of kNN, the value of k will be 5, since it is a usual value in the literature [12, 15, 21]. In addition, to evaluate the quality of the different models it is necessary to compute several measures. In this case, Accuracy, F-Score and runtime will be used.

3.2 AE Architecture

In this section, DAE architecture used to perform the fusion of features is presented. As indicated above, these models have a symmetric architecture where the number of neurons of the input layer is the same as that of the output layer. In this study, the objective is to reduce the dimensionality of the input data, therefore, the intermediate layer (or layers) must have a smaller number of units than the input and output.

To perform the evaluation of the model, a 50% reduction in the number of original attributes is proposed. Therefore, the model will have the following layers:

- Input layer: This part will have as many units as features has the data set.
- Hidden layer: The number of elements will be half of units of the input layer.
- Output layer: Due to the operation of the AE, it must have the same number of neurons as the input layer.

The architecture indicated above has a single hidden layer. The main reason for this is to conduct a baseline study on the most basic model of DAE, with the aim of obtaining a first approximation to the reduction of dimensionality through this type of AEs. Likewise, the number of elements in the hidden layer is established at 50% of the number of characteristics, since it is considered a significant reduction for observing the performance of DAEs in this task.

3.3 Results Analysis

The experimentation carried out consists of several executions where data sets are classified by 4 different algorithms: C4.5, kNN, MLP and SVM. Each method classifies the original data set and the reduced data set by DAE.

As described in Sect. 3.1, there are 6 data sets with different traits. Thus, Tables 2, 3, 4 show the results for Accuracy, F-Score and Runtime metrics, respectively.

Table 2. Accuracy classification results for test data.

Dataset	C4.5		kNN		MLP		SVM	
	Base	DAE	Base	DAE	Base	DAE	Base	DAE
Image	0.840	0.861	0.928	0.951	0.548	0.865	0.860	0.879
isolet	0.741	0.748	0.872	0.899	0.371	0.359	0.935	0.949
madelon	0.521	0.538	0.531	0.598	0.501	0.618	0.578	0.588
mfeat	0.890	0.896	0.438	0.975	0.086	0.860	0.976	0.982
nomao	0.877	0.872	0.891	0.902	0.901	0.563	0.914	0.919
semeion	0.612	0.608	0.908	0.917	0.781	0.453	0.890	0.950

Table 3. F-Score classification results for test data.

Dataset	C4.5		kNN		MLP		SVM	
	Base	DAE	Base	DAE	Base	DAE	Base	DAE
Image	0.833	0.864	0.927	0.951	0.494	0.870	0.850	0.879
isolet	0.743	0.749	0.874	0.900	0.350	0.329	0.935	0.950
madelon	0.521	0.566	0.551	0.590	0.512	0.583	0.571	0.579
mfeat	0.890	0.898	0.447	0.975	0.016	0.849	0.976	0.982
nomao	0.877	0.872	0.892	0.902	0.897	0.280	0.914	0.919
semeion	0.614	0.608	0.910	0.917	0.784	0.413	0.891	0.952

Table 4. Time classification results (in seconds) for test data.

Dataset	C4.5		kNN		MLP		SVM	
	Base	DAE	Base	DAE	Base	DAE	Base	DAE
Image	30.002	20.424	0.311	0.173	1451.797	931.370	1.248	0.945
isolet	8645.028	5903.823	64.465	33.781	71800.729	38842.212	410.516	201.704
madelon	250.329	112.155	3.967	1.716	6518.699	3220.424	18.843	7.592
mfeat	282.250	183.543	4.885	2.221	8618.690	4287.535	12.995	6.746
nomao	27.973	15.148	0.384	0.223	1347.476	750.514	1.396	0.762
semeion	91.609	50.778	1.023	0.436	2402.432	1363.049	5.417	2.305

C4.5 Analysis: The results of the C4.5 algorithm shown in Tables 2, 3, 4 reflect that the reduction of dimensionality with DAE improves the predictive performance in certain datasets. In particular, the executions with the reduced datasets obtain better results in 4 out of 6 tests for the Accuracy and F-Score metrics, while in the 2 remaining datasets better results are obtained with the original dataset. In terms of execution time, the results with the reduced dataset are better in all cases. The reason is obvious, since the time will be less when classifying a set of data with fewer features.

kNN Analysis: The tests show better predictive performance for the kNN algorithm by reducing the number of input characteristics through DAE, since all datasets are improved. On the one hand, the tests performed after performing the fusion of characteristics obtain better results in all cases for the Accuracy and F-Score metrics. On the other hand, the results in terms of time are similar to the algorithm C4.5, since the classification with fewer features is faster.

MLP Analysis: The executions carried out indicate that the reduction of dimensionality does not have clear effects for MLP, since there are datasets with better performance and others with worse. In detail, there are 3 datasets whose results improve considering Accuracy and F-Score, while the performance decreases in the other 3. With respect to time, the results are similar to the two algorithms previously reported. Therefore, the performance applying dimensionality reduction is comparable to that obtained by not doing it. However, the time is less when the input space is less.

SVM Analysis: In the case of the SVM method, the analysis is very similar to the kNN algorithm, since the improvement of the predictive performance is clear, obtaining superior results for all the datasets. Tables show: on the one hand, the Accuracy and F-Score metrics show better results for the 6 datasets used; on the other hand, the execution time is less when reducing the dimensionality, in the same way as in the previous algorithms.

Results Discussion: The data presented in this study show improvements in predictive performance in the four families of algorithms analyzed. The use of DAE to reduce the dimensionality of the input space allows to achieve better results. The reason is different for each of the methodologies.

In particular, the reduction of dimensionality for kNN implies that the distances between individuals are more significant, therefore, the performance is improved. As for SVM, a smaller input space implies that the spatial representation is smaller and groupings of elements can be performed more precisely. In these two cases the improvements are significant, since these are observed for all datasets.

For the C4.5 and MLP methods the improvements are less significant, there are cases in which they are improved and others in which they are not. This gives a first sample of which the characteristics of this type of algorithms do that the improvements obtained when reducing the dimensionality by means of DAE are not as significant as for kNN and SVM. However, the execution time improves for all the methods analysed.

3.4 DAE vs PCA

In this Subsection, the objective is to assess the competitiveness of DAEs against traditional dimensionality reduction algorithm, such a PCA [20,22]. A 50% reduction in the number of original attributes is performed with these methods, since it is the same one that has been done with DAEs.

Table 5. Accuracy and F-Score classification results of DAE and PCA for test data

Dataset	Accuracy								F-Score							
	C4.5		kNN		MLP		SVM		C4.5		kNN		MLP		SVM	
	DAE	PCA	DAE	PCA	DAE	PCA	DAE	PCA	DAE	PCA	DAE	PCA	DAE	PCA	DAE	PCA
Image	0.861	0.835	0.951	0.862	0.865	0.801	0.879	0.843	0.864	0.836	0.951	0.862	0.870	0.805	0.879	0.844
isolet	0.748	0.701	0.899	0.589	0.359	0.333	0.949	0.943	0.749	0.703	0.900	0.638	0.329	0.301	0.950	0.944
madelon	0.538	0.513	0.598	0.511	0.618	0.590	0.588	0.562	0.536	0.515	0.590	0.503	0.583	0.582	0.585	0.569
mfeat	0.896	0.853	0.975	0.703	0.860	0.821	0.982	0.951	0.898	0.853	0.975	0.771	0.849	0.823	0.982	0.955
nomao	0.872	0.853	0.902	0.869	0.563	0.934	0.919	0.902	0.872	0.851	0.902	0.883	0.280	0.853	0.919	0.898
semeion	0.608	0.594	0.917	0.706	0.453	0.403	0.950	0.936	0.608	0.596	0.917	0.718	0.453	0.406	0.952	0.937

Table 5 shows the results obtained with different classification algorithms after applying dimensionality reduction with DAE and PCA. The best predictive performance is obtained with the dataset generated by DAEs for most of the dataset and classification algorithms.

4 Concluding Remarks

One of the problems that affect many classification algorithms is the high dimensionality of the data. This feature is present in many real datasets. Therefore, it

is important to propose solutions that mitigate the negative effects of this factor. In this work, DAE are used to reduce the dimensionality and a series of tests are performed to see their effects in classification algorithms corresponding to different methodologies.

On the one hand, the experimentation carried out has shown that the predictive performance for the kNN and SVM algorithms is significantly improved. On the other hand, the results for the C4.5 and MLP algorithms are improved for some of the datasets used. This indicates that the use of DAE to reduce the dimensionality offers better performance for the IBL and SVM algorithms. In addition, the comparison between DAEs and PCA shows that DAEs offer better results in most of the cases analysed.

The results derived from this empirical verification open new lines of future work. The experimentation can be extended with new datasets. Likewise, other types of AEs can be used to face the task of dimensionality reduction. Another line of work is the creation of models that combine dimensionality reduction techniques based on AEs and traditional classification algorithms.

In conclusion, this study allows us to take the first step to address the problem of dimensionality reduction using DL-based models and generate hybrid models that improve predictive performance when classifying high-dimensional data.

Acknowledgment. The work of F. Pulgar was supported by the Spanish Ministry of Education under the FPU National Program (Ref. FPU16/00324). This work was partially supported by the Spanish Ministry of Science and Technology under project TIN2015-68454-R.

References

1. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based learning algorithms. *Mach. Learn.* **6**(1), 37–66 (1991)
2. Bache, K., Lichman, M.: UCI Machine Learning Repository (2013)
3. Batista, G.E., Prati, R.C., Monard, M.C.: A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explor. Newsl.* **6**(1), 20–29 (2004)
4. Bellman, R.: *Dynamic Programming*. Princeton University Press, Princeton (1957)
5. Bellman, R.: *Adaptive Control Processes: A Guided Tour*. Princeton University Press, Princeton (1961)
6. Bengio, Y.: Deep learning of representations: looking forward. In: Dediu, A.-H., Martín-Vide, C., Mitkov, R., Truthe, B. (eds.) *SLSP 2013. LNCS (LNAI)*, vol. 7978, pp. 1–37. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39593-2_1
7. Beyer, K., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is “Nearest Neighbor” meaningful? In: Beeri, C., Buneman, P. (eds.) *ICDT 1999. LNCS*, vol. 1540, pp. 217–235. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-49257-7_15
8. Charte, D., Charte, F., García, S., del Jesus, M.J., Herrera, F.: A practical tutorial on autoencoders for nonlinear feature fusion: taxonomy, models, software and guidelines. *Inf. Fusion* **44**, 78–96 (2018)
9. Cole, R., Fandy, M.: Spoken letter recognition. In: *Proceedings of the Workshop on Speech and Natural Language*, pp. 385–390 (1990)

10. Cover, T., Hart, P.: Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory* **13**(1), 21–27 (1967)
11. Deng, L.: Deep learning: methods and applications. *Found. Trends Signal Process.* **7**(3–4), 197–387 (2014)
12. Derrac, J., Chiclana, F., García, S., Herrera, F.: Evolutionary fuzzy k-nearest neighbors algorithm using interval-valued fuzzy sets. *Inf. Sci.* **329**, 144–163 (2016)
13. Duda, R.O., Hart, P.E., Stork, D.G.: *Pattern Classification*. Wiley, New York (1973)
14. Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., Herrera, F.: A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **42**(4), 463–484 (2012)
15. Ghosh, A.K.: On optimum choice of k in nearest neighbor classification. *Comput. Stat. Data Anal.* **50**(11), 3113–3123 (2006)
16. Guyon, I., Gunn, S., Ben-Hur, A., Dror, G.: Result analysis of the NIPS 2003 feature selection challenge. In: *Proceedings of Neural Information Processing Systems*, vol. 4, pp. 545–552 (2004)
17. Hearst, M.A., Dumais, S.T., Osuna, E., Platt, J., Scholkopf, B.: Support vector machines. *IEEE Intell. Syst. Their Appl.* **13**(4), 18–28 (1998)
18. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. *Science* **313**(5786), 504–507 (2006)
19. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. *Neural Netw.* **2**(5), 359–366 (1989)
20. Hotelling, H.: Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol.* **24**(6), 417–441 (1933)
21. Keller, J.M., Gray, M.R., Givens, J.A.: A fuzzy k-nearest neighbor algorithm. *IEEE Trans. Syst. Man Cybern. SMC* **15**(4), 580–585 (1985)
22. Pearson, K.: LIII. On lines and planes of closest fit to systems of points in space. *Lond. Edinb. Dublin Philos. Mag. J. Sci.* **2**(11), 559–572 (1901)
23. Quinlan, J.R.: Induction of decision trees. *Mach. Learn.* **1**(1), 81–106 (1986)
24. Quinlan, J.R.: *C4. 5: Programs for Machine Learning*. Elsevier, Amsterdam (2014)
25. Schalkoff, R.J.: *Artificial Neural Networks*, vol. 1. McGraw-Hill, New York (1997)
26. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: *Proceedings of the 25th International Conference on Machine Learning*, pp. 1096–1103. ACM (2008)
27. Yu, H., Yang, J.: A direct LDA algorithm for high-dimensional data-with application to face recognition. *Pattern Recognit.* **34**(10), 2067–2070 (2001)
28. Zadrozny, B., Elkan, C.: Learning and making decisions when costs and probabilities are both unknown. In: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 204–213. ACM (2001)



An Approximation to Deep Learning Touristic-Related Time Series Forecasting

Daniel Trujillo Viedma^(✉), Antonio Jesús Rivera Rivas,
Francisco Charte Ojeda, and María José del Jesus Díaz

Andalusian Research Institute on Data Science and Computational Intelligence
(DaSCI), Computer Science Department, University of Jaén, 23071 Jaén, Spain
dtviedma@ujaen.es
<http://wwwdi.ujaen.es/>

Abstract. Tourism is one of the biggest economic activities around the world. This means that an adequate planning of existing resources becomes crucial. Precise demand-related forecasting greatly improves this planning. *Deep Learning* models are showing a greatly improvement on time-series forecasting, particularly the LSTM, which is designed for this kind of tasks. This article introduces the touristic time-series forecasting using LSTM, and compares its accuracy against well known models RandomForest and ARIMA.

Our results shows that new LSTM models achieve the best accuracy.

Keywords: LSTM · ARIMA · Time series forecasting

1 Introduction

World tourism is considered one of the most important activities from an economic point of view, only exceeded by oil and derivative products [1]. In the case of Spain, 12.1% of the total employments in year 2013 were linked to touristic services, starting from a 9.8% in 2001 [3]. In 2016, it represents 13% of total employment and 11,2% of Spanish GDP, approximately 125.529 millions of euros [4]. Such a big economic activity requires an accurate demand forecasting, in order to adapt the resources (beds, services, etc.) to successfully attend a highly variable demand.

By its nature, time series are important to this kind of economic activity, given the high number of relevant variables around it that are modeled in a time-dependent way such as the number of customers, available beds, or average customer spending, among others. These variables can only be analyzed if their observations are taken into account in a time-ordered fashion, and this let's us to make predictions based on the same order. They are, therefore, time series.

Typically, times series have been modeled using the ARIMA (*AutoRegressive Integrated Moving Average*) [8] technique, which integrates a moving averages analysis with an autoregressive one, known as ARMA model, and extend it to

work with differentiated time series. ARIMA, together with heuristic analysis in order to find the best values for its parameters, has a strong statistical foundation and is considered state of the art in time series forecasting.

On the other hand, new deep learning models have greatly improved previous machine learning algorithms, thanks to better available training methods and faster computers on which run those methods. These models have improved tasks such as handwriting recognition [9] or statistical machine translation [2].

One of these models, LSTM (*Long Short-Term Memory*) [6], is specially designed for time series forecasting. Its structure allows it to store relevant information to be consumed on the next input, providing an useful extended context when analyzing an input time step, greatly improving the accuracy.

The main purpose of this study is to introduce touristic time-series forecasting with LSTM, while comparing its accuracy against the well known models RandomForest and ARIMA.

This article is structured as follows: The next section, the second one, briefly describes the methods involved in this study. The third section details the experiments made to test all the models. Finally, the results of those experiments are shown and discussed with a brief, final conclusion.

2 Methods

This section gives an introductory description of the methods involved in this study, LSTM, RandomForest and ARIMA. As providing a complete description of these methods is not a goal of this article, most important references are cited.

2.1 LSTM

LSTM is an artificial recurrent neuron architecture. Its structure allows it to maintain an internal state, made from past observations of the neuron, that drives future outputs. This internal state constitutes, in fact, a memorized knowledge, stored in a component called CEC (*Constant Error Carrousel*).

LSTM is designed to solve the exploding and vanishing gradient problems which are extensively discussed in [7], being vanishing gradient the most frequently found in neural networks. This problem references the decreasing effect of a network parameter (like an actual weight) over the network output, as far as the number of layers between the parameter and the output increases, affecting very deep feedforward networks and recurrent networks with big time lags. This model boosted recurrent networks research, by providing practical advantages over traditional feedforward networks due to solving the vanishing gradient problem [7].

In its original formulation, [10], this processing block, called *cell*, defines a data pipeline through which input values are transformed by means of multiplicative operations and activation functions. Additions to this structure were later added, setting the de facto standard configuration of the cell. These additions can be summed up in the following:

- *Forget gate*: Was introduced by [5]. Previously, the output of the CEC was simply copied to its input, and the input of the CEC was transformed by another activation function. With the addition of the product operation, this activation function no longer makes sense.
- *Peephole connections*: Bypass values from the CEC’s output to the (input, forget and output) gates.

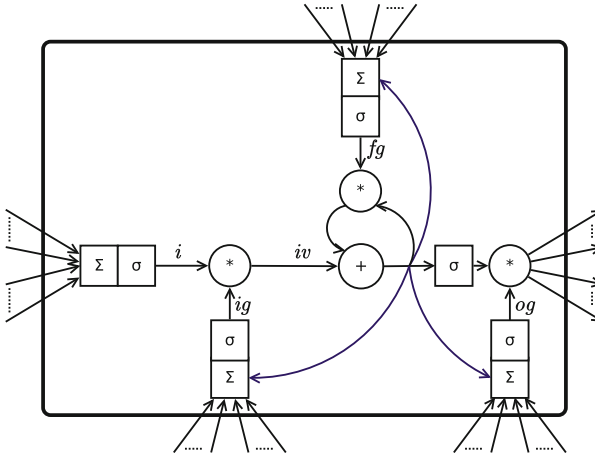


Fig. 1. Schematic of an LSTM cell.

Figure 1 details the LSTM cell components. The data pipeline can be quickly seen, starting from the left side, where the input values are provided to the cell, until the right side, where the output values are returned from the cell; while the data is transformed by each gated unit. Also, the CEC can be appreciated.

This data pipeline can be described as follows:

1. Input values are provided to the *input*, *input gate* y *forget gate*, computing values i , ig y fg , respectively.
2. i and ig are combined by a pointwise product operation, obtaining iv .
3. The pointwise product between fg and the value from the last CEC evaluation is computed. This value will feed the CEC the next time step, added up in a pointwise fashion with iv . If there are peephole connections present, this CEC’s output value is copied to the destination of these connections.
4. The CEC’s output value feed an activation function, which is multiplied, also in a pointwise fashion, with the *output gate* value, named og , which is computed given the recurrent input to the cell and the previously computed CEC’s output value.

This internal structure was designed to work with long input time dependencies. Along with an adequate training algorithm, such as backpropagation,

a constant error flow through the CEC is assured, while the *input gate* protects CEC from irrelevant inputs, while the *output gate* protects another cells from irrelevant CEC values.

This *modern LSTM* is described in detail in [6].

The most commonly used training method used with this kind of neural networks is BPTT (*Backpropagation Through Time*), which is a generalization of the well known Backpropagation algorithm, adapted to work with recurrent networks. It starts by unrolling the LSTM layer so that it becomes a feedforward network, given a recursion limit. Then, applies a standard Backpropagation to obtain changes in weights. Lastly, aggregates (typically, with an statistical mean) the changes corresponding to the same recursive weight. This algorithm is detailed in [11].

2.2 RandomForest

RandomForest is just an ensemble of decision trees in which every tree is specialized on predicting over a number of randomly chosen variables from the whole dataset. A new instance is evaluated by retrieving the output of each tree separately and then deciding the final output of the model. Usually, the most predicted class, or the mean of predicted values is given as a final output.

The number of variables each tree sees is set as a parameter, and then the optimal number of trees is derived by means of minimal out-of-bag error.

In this scenario, a RandomForest model can be trained to predict the next value of a sequence given the past ones, and also including another variables to improve accuracy.

2.3 ARIMA

ARIMA [8] is a widely used method for time series forecasting. It works by finding a function that approximates the input time series and that can be evaluated for further time steps. That means that ARIMA is powerful for predictive tasks, but also useful in descriptive tasks.

ARIMA builds on top of the ARMA model, extending it to work with differentiated time series (in essence, a data sequence composed by the differences between an element and the following one from the original sequence). ARMA stands for *AutoRegressive and Moving Averages* and consists on finding an autoregressive adjust over the differences of the input values, and then model the error with a moving averages model.

3 Experiments

In order to establish a comparison between the distinct models considered, a real world touristic dataset has been used to train several models of each kind (LSTM, RandomForest and ARIMA), validating them by means of two error metrics.

From all the models trained, a selection of the best of each kind was made by training accuracy criteria. Later, these models were validated with new data, which doesn't got involved in training at all. This way, we can simulate a realistic scenario where the future values are not know at the present, but will be known later.

3.1 Error Metrics

We have considered 2 distinct error functions: RMSE (*Root Mean Squared Error*) and MAPE (*Mean Absolute Percentage Error*).

These error metrics quantifies the deviation of the predictions made with the models from the real data.

Given E the expected values and P the predicted values (both of them, with the same number of observations), these metrics are defined as follows:

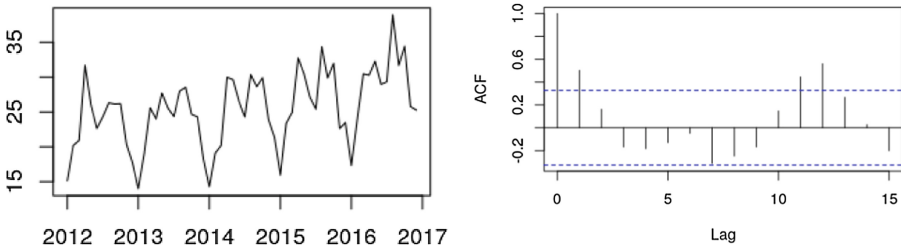
$$RMSE(E, P) = \sqrt{MSE(E, P)} \tag{1}$$

$$MSE(E, P) = \frac{\sum_{i=1}^n (E_i - P_i)^2}{n} \tag{2}$$

$$MAPE(E, P) = \frac{100}{n} \sum_{i=1}^n \frac{E_i - P_i}{E_i} \tag{3}$$

3.2 Dataset

The dataset used in this study was retrieved from the Spain's Instituto Nacional de Estadística. More precisely, the touristic occupation survey, occupation grade per place series in the province of Jaén. The data has been plotted in Fig. 2a. Data is offered in a monthly resolution, and only values between January 2012 and December 2016 are considered.



(a) Visualization of the data set. (b) Autocorrelation function.

Fig. 2. Raw data and ACF graphs.

At first appearance, some patterns can be easily recognised, as well as occupation peaks which corresponds to the summer holidays and the Easter holidays. As expected, there's a clear correlation between years.

From this dataset, values of year were extracted to be used only in the final validation, being the rest (January 2012 - December 2015) used for training.

In order to improve accuracy, additional variables whose values are known a priori have been merged into the dataset, namely:

- Year.
- Month of the year.
- Easter: Boolean stating if the instance belongs to a month where a Easter occurs.
- Sort weeks: Number of holidays in Tuesday or Thursday.
- Long weekends: Number of holidays in Monday or Friday.

The time lags included in LSTM and RandomForest instances were chosen following a autocorrelation function criteria. This function applied over the training input data is shown in Fig. 2b, in where the most relevant time lags can be seen easily.

Though this is the general structure of the dataset, some model's training algorithms implementations forces us to adapt this structure:

LSTM and RandomForest: Each instance has the following form:

$(V_n, V_{n-1}, V_{n-11}, V_{n-12}, HW, SW, LW)$.

ARIMA w/xregs: Each instance has the following form:

$(V_n, V_{n-12}, HW, SW, LW)$.

Being:

n The month the instance data refers to.

V_i The value of the time series at month i .

HW If the month contains an Easter (or part of it).

SW Number of short weeks within the month.

LW Number of long weekends within the month.

3.3 Methods

Experiments with LSTM neural networks were made with a *Python* script, using *Keras* library with *TensorFlow*. Also, *Pandas* library was used for data structures, *Numpy* for numerical computing, *Matplotlib* for data visualization and *Scikit-learn* for error metric computing and data normalization.

The parameter settings taken into account have been: 5 and 10 LSTM cells per network; 800 and 1000 iterations; and 1, 2, and 4 batch sizes.

Given the probabilistic nature of the training algorithm (Backpropagation Through Time), which randomly initializes the networks weights, 20 repetitions of each parameter set have been made, in order to reduce the impact of the initial weights on the computed errors.

The ARIMA and RandomForest experimentation was made in R, justified by the quality of the implementations of these two kind of models in *caret*, *randomForest* and *forecast* packages. More precisely, we have used *auto.arima*

function to train the ARIMA model, and caret’s *train* function, with *method* parameter set to “*rf*”, and *mtry* parameter varying from 1 to 6. In the ARIMA case, we trained 2 kind of models: with and without external regressors, given the important accuracy disparity both models exhibit.

3.4 Results

Table 1 shows the error made by the best accurated LSTM models trained, in ascending error order. For each one of the parameter configuration, mean and standard deviation of RMSE from 20 repetitions are listed. The first (less erratic) model is selected to compete with ARIMA and RandomForest. In a similar way, Table 2 shows the result for RandomForest experiments. Also, like in the LSTM case, 20 repetitions were made for each specific parameter setting.

Table 1. LSTM - Training dataset - RMSE

Num LSTM	Epochs	batch size	RMSE	
			Average	Std. Dev.
5	1000	4	2.1291	0.0260
5	1000	1	2.1323	0.0283
5	800	1	2.1331	0.0332
5	800	2	2.1361	0.0334
10	800	1	2.1389	0.0314
5	800	4	2.1389	0.0320
10	800	4	2.1419	0.0216
5	1000	2	2.1446	0.0331
10	800	2	2.1455	0.0292
10	1000	4	2.1481	0.0271
10	1000	1	2.1496	0.0501
10	1000	2	2.1698	0.0195

Table 2. RandomForest - Training dataset - RMSE

mtry	RMSE	
	Average	Std. Dev.
6	2.4680	0.0636
5	2.5092	0.0463
4	2.5300	0.0608
3	2.5370	0.0352
2	2.6612	0.0898
1	3.4334	0.0639

Table 3 shows the result of ARIMA experiment. Given the fact that ARIMA is not probabilistic, it doesn't make sense to run it several times, since all of it will have exactly the same error. It's easy to see that, when external regressors are added into the ARIMA model training, it greatly improves the accuracy of the resulting model.

Table 3. ARIMA

xregs	Training	Testing	
	RMSE	RMSE	MAPE
No	2.4056	1.8901	5.0111
Yes	3.6616	6.7481	18.7578

Table 4. Testing dataset accuracy

	RMSE	MAPE
LSTM	1.7216	4.8653
ARIMA w/xregs	1.8901	5.0111
ARIMA	6.7481	18.7578
RandomForest	5.7435	16.2852

Lastly, Table 4 shows the accuracy, this time on the testing dataset, of the best parameter setting for each kind of model (LSTM, ARIMA and RandomForest). This data provides evidence that support our thesis, that LSTM models have best accuracy on testing data than classical (non deep learning), state of the art, ones.

4 Conclusion

In this work, we established a comparison between 3 different kind of models, including a novel, deep learning one, and another coming from Statistics field, considered the state of the art in time series forecasting, to improve a real world touristic occupation forecasting, which justifies by the economic importance of tourism activities around the world.

The results of this comparison, which can be seen in Table 4, shows that the deep learning model, LSTM, can achieve best accuracy than the other models taken into account.

Acknowledgements. This work is partially supported by the Spanish Ministry of Science and Technology under project TIN2015-68454-R.

References

1. Altés, C.: Marketing y turismo. Editorial Síntesis, Madrid (1993)
2. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint [arXiv:1406.1078](https://arxiv.org/abs/1406.1078) (2014)
3. Cuadrado Roura, J.R., López Morales, J.M., et al.: El turismo, motor del crecimiento y de la recuperación de la economía española (2015)
4. de Estadística, I.N.: Aportación del turismo a la economía española (2016). <http://www.ine.es/>
5. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: continual prediction with LSTM (1999)
6. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **18**(5–6), 602–610 (2005)
7. Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J., et al.: Gradient flow in recurrent nets: the difficulty of learning long-term dependencies (2001)
8. Hyndman, R.J., Khandakar, Y., et al.: Automatic time series for forecasting: the forecast package for R. No. 6/07. Monash University, Department of Econometrics and Business Statistics (2007)
9. Romanjuk, V.V.: Training data expansion and boosting of convolutional neural networks for reducing the mnist dataset error rate. *Naukovi Visti NTUU KPI* **6**, 29–34 (2016)
10. Schmidhuber, J., Hochreiter, S.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
11. Werbos, P.J.: Generalization of backpropagation with application to a recurrent gas market model. *Neural Netw.* **1**(4), 339–356 (1988)



CCTV Image Sequence Generation and Modeling Method for Video Anomaly Detection Using Generative Adversarial Network

Wonsup Shin and Sung-Bae Cho^(✉)

Department of Computer Science, Yonsei University, Seoul, Republic of Korea
{wsshin2013, sbcho}@yonsei.ac.kr

Abstract. Video anomaly detection is one of the most attractive problem in various fields likes computer vision. In this paper, we propose a VAD classifier modeling method that learns in a supervised learning manner. The basic idea is to solve the problem of labeled data shortage through transfer learning. The key idea is to create an underlying model of transfer learning through the GAN of discriminator. We solved this problem by proposing a GAN model consisting of a generator that generates video sequences and a discriminator that follows LRCN structure. As a result of the experiment, The VAD classifier learned through GAN-based transfer learning obtained higher accuracy and recall than the pure LRCN classifier and other machine learning methods. Additionally, we demonstrated that the generator be able to stably generate the image similar to the actual data as the learning progressed. To the best of our knowledge, this paper is the first case to solve the VAD problem using the GAN model and the supervised learning manner.

Keywords: Autonomous CCTV surveillance · Video anomaly detection
Transfer learning · Supervised learning · Generative adversarial network

1 Introduction

Due to the widespread usage of surveillance cameras and the limitations of surveillance system through manpower, the need for an automatic video surveillance system is rapidly emerging [1]. A fundamental challenge of autonomous video surveillance systems is to automatically detect anomaly, defined as unusual, uncommon or irregular event, in complex and crowded scenes [2, 3]. This issue is one of the major issues in the field of computer vision and various researchers have been working to solve the problem [4–7].

The methods for solving the Video Anomaly Detection (VAD) problems should be able to extract spatial-temporal features and to classify anomaly through features. Deep Neural Network (DNN) is a one of the key technologies for modeling these methods. Especially, models combining the Convolutional Neural Network (CNN), which can extract spatial information well and the Long Short-Term Memory (LSTM), which can

learn temporal information well, are making notable achievements in the domains using video [8, 9].

The deep VAD classifiers have been modeled almost in an absolutely unsupervised learning manner [10–12]. The reasons why unsupervised based approaches are preferred are: first, it is difficult to collect sufficient amounts of labeled anomaly data required to learn the internal variance of anomalies and second, anomalies that are not included in the training data may occur. However, labeled data is very valuable and important information that can guide the learning direction of the classifier. Therefore, when there is even a little bit of labeled abnormal data, it is hard to think that the unsupervised based approaches which does not consider any labeled data are the best way to solve the problem. Even semi-supervised learning-based approaches in VAD problem exclude consideration of labeled abnormal data. These approaches are only interested in configuring a training set with normal data [6]. In this context, research on supervised learning-based methods is important in terms of using existing data as efficiently as possible.

In order to solve this problem with the supervised learning approach, we must solve the two problems of the labeled data mentioned above. Among the two, this study suggests a way to overcome the lack of labeled data. Transfer learning is the key to solving the problem of small amount of labeled data. It transfers core parts of other model learned with vast amounts of data to the target model. The target model, which has been transferred, just fine-tuned with a small amount of data [13]. But, how to get an enormous amount of labeled dataset? Our approach to dealing with this issue is to use Generative Adversarial Network (GAN) [14]. GAN, which has been evaluated as the most innovative deep-running model in recent years, is a key to generate and learn various kinds of normal and abnormal data. The discriminator of GAN is learned through actual data and extensive fake data. The main idea of this study is to transfer the learned parts of the discriminator to the VAD classifier.

In this paper, we propose a method to generate a deep VAD classifier that learns in a supervised learning manner with small amount of labeled data through transfer learning. The proposed method uses GAN which uses Long-term Recurrent Convolutional Network (LRCN) [8] as discriminator. The procedure of the proposed method has two phases. First, learns GAN with existing dataset. Second, transfers the feature extractor of the discriminator to the VAD classifier. Finally, fine-tunes the classifier with the labeled dataset. Since the variance of the data generated by the GAN is large, the feature extractor has the effect of learning through a large dataset with various classes of normal or abnormal data. The main contributions of this paper are summarized as follows.

- Proposed the method of generating deep VAD classifier in a supervised learning manner that can learn with a small amount of labeled data.
- Proposed a transfer learning method using GAN in VAD domain.

The rest of this paper is organized as follows. The related works for VAD problem and some guides of transfer learning are introduced in Sect. 2. The proposed model is introduced in Sect. 3. Section 4 presents the experiments and performance comparison with other machine learning methods. We conclude this paper with remarks on future work in Sect. 5.

2 Related Work

2.1 Previous Works of Image or Video Anomaly Detection

This section briefly introduces various previous studies for comparison with the proposed method. To illustrate the motivation of the proposed method, we introduce some previous studies that use image data as domain. As mentioned in the previous section, methods based on unsupervised learning or semi-supervised are dominant.

From the late 2000s until the early 2010s, methods of directly embedding information of objects existing in images have been attempted. For example, [15] proposed a social force model that detects anomaly by embedding the correlation between objects in a bag of word method. [16] proposed a method to detect anomaly in real time by embedding data between steps through dynamic sparse coding method.

In the mid-2010s, studies combining RNN and feature extraction methods have been actively conducted. [17] and [18] proposed models combining recurrent model and existing direct feature embedding methods. [17] proposed Social-LSTM that combines social force model and LSTM. And [18] proposed a model that combines the sparse coding method with the stacked RNN. [19] and [8] proposed models combining LSTM and convolutional layer. [19] proposed the Convolutional-LSTM in which the input and output of the LSTM were changed to three dimensions and the vector operation between them was changed to a convolution operation. [8] proposed an LRCN that uses an image sequence, which passes through its own convolutional layer, as a sequential input of the LSTM. This method is intuitively easy to understand and has achieved state-of-the-art performance at the proposed moment.

In addition, [20] and [21] proposed an anomaly detection method using GAN, which was motivated by our research. However, they processed only image data and were unsupervised learning-based approaches. On the other hand, the proposed method is a supervised learning based approach that can perform anomaly detection on video data using LRCN as discriminator and using transfer learning. Table 1 summarizes the related studies that have been introduced.

Table 1. Related works on video or image anomaly detection

Category	Author	Year	Method
Direct feature embedding	R. Mehran	2009	Social force model
	B. Zhao	2011	Dynamic sparse coding
RNN + Direct feature embedding	A. Alahi	2016	Social-LSTM
	W. Luo	2017	TSC
LSTM + Convolutional layer	J. Donahue	2015	LRCN
	J. R. Medel	2016	ConvLSTM
GAN	T. Schlegl	2017	AnoGAN
	H. Zenati	2018	BiGAN

2.2 Transfer Learning

Transfer Learning is a way to accelerate learning speed of model (that is, the model can be learned with a small amount of learning data) and it improve predictive performance of new model by using existing well-formed models. In particular, this technique is more appropriate as the origin data are similar to the new training data which we have. [22] guides the transfer learning method of the DNN model consisting of the feature extractor and the classifier, such as ConvNet, for the four major scenarios as shown in Table 2. The discriminator of GAN is good underlying model of transition learning for VAD classifier. Because, it is a well-formed model using rich fake data which generated from generator and some real data. Even the datasets used for learning discriminators and VAD classifiers are very similar. Since the amount of labeled data in the VAD problem is generally small, we fine-tune the VAD classifier according to the left-most scenario in Table 2.

Table 2. A guide to transition learning methods for four major scenarios

Category	Text	Scenarios			
Checkout	Is there a lot of new training data?	X	O	X	O
	Is the new data similar to the existing data?	O	O	X	X
Diagnosis	Learn only the classifier	+++	-	+	-
	Learn the classifier and some other layers.	-	-	++	-
	Learn full-network	-	+++	-	++

3 The Proposed Method

3.1 Overview of the Proposed Model Structure

This paper proposes a classification model on video anomaly detection problem that learn with supervised learning manner. The overall structure of the proposed model is shown on Fig. 1. The proposed method consists of two learning phases. In the first phase, the constructor and the discriminator of the GAN are trained. The real data block consists of a part of the whole video data, and the generator also outputs a partial video of the same length as the real data block. The second is the transfer learning phase. In this step, only the feature extractor of discriminator is transferred to the our VAD classifier. Note that discriminator and classifier use very similar datasets, but the purpose of both classification models is obviously different. After classifier receives the feature extractor, the classifier is fine-tuned in a supervised learning manner through cross entropy loss. After all phases are completed, the performance of classifier is verified using the test dataset.

3.2 Generative Adversarial Network for Video Generation

This section describes the structure of the GAN model used in this paper. In this study, we divide a dataset into several 3-dimensional blocks using windowing method and

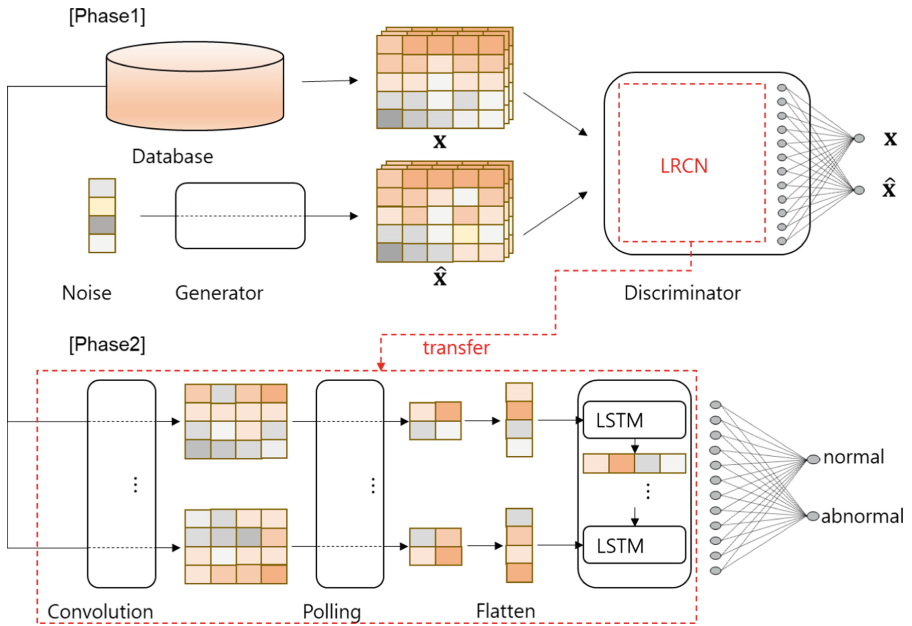


Fig. 1. Proposed model structure

define each block as a data x . The constructor G was designed based on the method presented in [22]. G receives randomly sampled noise vector z along the normal distribution. Next, through one fully connected layer and up-sampling layer, z is amplified to the same size as the real data block except for the number of filters. The amplified block is adjusted so that the number of filters is equal to the real data block through the two of convolutional layer. In addition, batch normalization was applied to all layers except the last convolution layer. As a result, the constructor creates a fake image sequence $\hat{x} = G(z)$. Table 3 shows the generator architecture used in this paper. The applied window size is 10.

Table 3. The proposed generator architecture

Layer	Output	Kerner size	Filters	Pad	Activation
Dense	1092000	–	–	–	ReLu
UpSampling	$200 \times 120 \times 182$	$2 \times 2 \times 1$	200	–	–
Conv2D	$100 \times 120 \times 182$	$3 \times 3 \times 1$	100	Same	ReLu
Conv2D	$50 \times 120 \times 182$	$3 \times 3 \times 1$	50	Same	ReLu
Conv2D	$10 \times 120 \times 182$	$1 \times 1 \times 1$	10	Same	Sigmoid

The discriminator D receives the real image sequences x and \hat{x} as input, and is learned to determine whether the input is actual data or fake data. The objective function $V(D, G)$ of the GAN is given by Eq. 1.

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))] \quad (1)$$

Where $D(\mathbf{x})$ represents the probability of classifying \mathbf{x} as the real data and p represents the probability that the sample is to be obtained. That is, G is learned so that D do not distinguish data generated by itself, and D is learned so that real data can be accurately distinguished from fake data.

The discriminator D extracts the spatial features of each image sequence block by placing a three convolution-pooling layer. The Eq. 2 defines value c_i obtained by convolving from i 'th pixels to $(i + M)$ 'th pixels by j 'th filter in each layer.

$$c_i^j = \sigma\left(b_j + \sum_{m=1}^M w_m^j x_{i+m-1}^j\right) \quad (2)$$

Where σ is the activation function, b is the bias term, and w represents the kernel value. And the Eq. 3 shows the operation of a pooling layer. Where r is a pooling size. As shown in Eq. 3, We get the abstracted data block \dot{x}_i^j through the max pooling operation. The max pooling is a type of pooling that selects the largest number in the subarea. It is generally known to perform better than average pooling or L2-norm pooling methods.

$$\dot{x}_i^j = \max_{r \in R} c_{i+r}^j \quad (3)$$

Next, spatial-temporal features of the image sequence block are extracted using the features of each image obtained by three convolution-polling layers as an input to the LSTM layer. In the LSTM, we use memory cells rather than simple recurrent units to store and output temporal features of image sequence data. It makes network to easily understand the relationship of image sequence of large time scale.

$$i_t = \sigma(W_{xi}\dot{x}_t + W_{hi}h_{t-1} + W_{ci} \circ c_{t-1} + b_i) \quad (4)$$

$$f_t = \sigma(W_{xf}\dot{x}_t + W_{hf}h_{t-1} + W_{cf} \circ c_{t-1} + b_f) \quad (5)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tanh(W_{xc}\dot{x}_t + W_{hc}h_{t-1} + b_c) \quad (6)$$

The Eq. 4–6 use notation i_t , f_t , and o_t , which represent input gate, forget gate and cell state at time step t , respectively. The notation \circ denotes elemental-wise product. The Eqs. 7 and 8 use notation o_t and h_t , which represent output gate and hidden value, respectively (Table 4).

$$o_t = \sigma(W_{xo}\dot{x}_t + W_{ho}h_{t-1} + W_{co} \circ c_t + b_o) \quad (7)$$

$$h_t = o_t \circ \tanh(c_t) \quad (8)$$

Finally, we connect two fully connected layers to the network that receive the output value of the LSTM layer as input. These fully connected layers play the role of a

Table 4. The proposed discriminator architecture

Layer	Output	Kerner size	Filters	Pad	Activation
TD (Conv2D)	$10 \times 114 \times 176 \times 16$	$7 \times 7 \times 1$	16	Valid	ReLU
TD (Pooling)	$10 \times 57 \times 88 \times 16$	$2 \times 2 \times 1$	–	–	–
TD (Conv2D)	$10 \times 53 \times 84 \times 16$	$5 \times 5 \times 1$	16	Valid	ReLU
TD (Pooling)	$10 \times 26 \times 42 \times 16$	$2 \times 2 \times 1$	–	–	–
TD (Conv2D)	$10 \times 24 \times 40 \times 16$	$3 \times 3 \times 1$	16	valid	ReLU
TD (Pooling)	$10 \times 12 \times 20 \times 16$	$2 \times 2 \times 1$	–	–	–
Dropout (50%)	–	–	–	–	–
LSTM	1024	–	–	–	–
Dropout (50%)	–	–	–	–	–
Dense	256	–	–	–	ReLU
Dense	2	–	–	–	Softmax

classifier to discriminate whether the features of the image sequence extracted from the previous layers are real data or fake data.

Since the proposed neural network is deep, Rectified Linear unit(ReLU) is used as an activation function of all convolutions and fully connected layers. And a 50% dropout layer is placed between the LSTM layer and the fully connected layers to prevent overfitting. Table 3 shows the LRCN discriminator architecture used in this paper. Where notation TD represents the time distributed rapper. The architecture of proposed VAD classifier is same.

4 Experimental Result

4.1 UCSD-Pedestrian1 Dataset

In this paper, we used UCSD-pedestrian1 dataset for experiments [23]. This dataset is a collection of surveillance camera image sequence data recording sidewalk. The ordinary pedestrians are classified as normal event and five types of abnormal events -biker, skateboarder, person in wheelchair, car, and people who do not walk along the road-are classified as anomaly. The dataset consists of 70 videos where each of the length is 200 frames, of which 34 images contain anomaly. Also, only 29% of the total frames are labeled as anomaly. This means that the data imbalance problem is quite severe. Therefore, in order to solve the data imbalance problem, we preprocessed data through sliding window method and under sampling method [24]. For the same purpose, data blocks with anomaly frame equal to or more than half of the window size are labeled as anomaly. As a result of preprocessing, 3885 anomaly data blocks were generated, and through under-sampling process a total of 7770 data blocks were defined as datasets to be used in the experiment. Additionally, due to limitations in learning speed and memory storage capacity, we adjust the size of each image to 182×120 (Fig. 2).

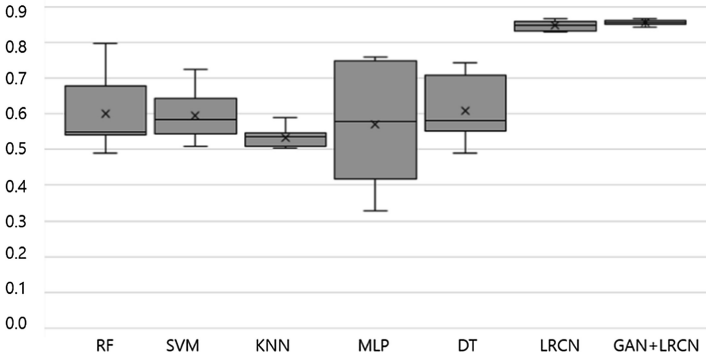


Fig. 2. Comparison of accuracy by 10-fold cross validation

Table 5. Comparison of LRCN model and proposed model

Method	Recall	Precision	Accuracy	F1-score
LRCN	0.95343	0.78647	0.84645	0.86111
GAN+LRCN	0.96963	0.78890	0.85502	0.86992

4.2 Results and Analysis

To demonstrate the usefulness of the proposed method, we conducted a 10-fold cross validation experiment with other machine learning methods. Since the number of features is still extremely large in other machine learning algorithms that do not use the convolution network, we reduced the size of each input image sequence to 90×60 size. Network architecture of LRCN and Proposed are same. Experimental results show that the existing machine learning methods have a performance that is not significantly different from that of random selection, whereas the proposed method with LRCN has more than 80% accuracy. However, the proposed method has more robust learning performance than the existing LRCN method because it has higher average accuracy and less variance.

Additionally, we compare and analyze additional indicators for existing LRCN methods and proposed methods. Additional indicators used in this experiment are recall, precision, and F1 score. The most important indicator of additional indicators is recall. This is because it means that model can react with the highest probability to an actual anomaly situation. Equation 9 is an equation for recall. TP means true positive and FP means false positive.

$$recall = \frac{TP}{TP + FP} \tag{9}$$

Table 5 shows the result of the comparison. Experimental results show that recall of the proposed method is 1.7% higher. Precision was also slightly higher than the conventional method. This means that there is a lower probability that an alarm will

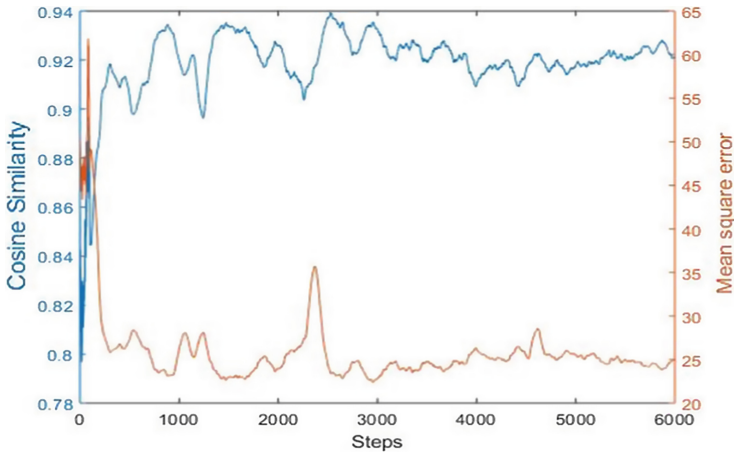


Fig. 3. Comparison of similarity between real and fake image sequences



Fig. 4. Real data (left) and fake data (right)

sound wrong in a normal situation. Accuracy, the most intuitive indicator, also rose by about 1%.

Finally, we verify the similarity between real and fake image sequences to show that the GAN can actually create a fake image sequence similar to a real image sequence. In this experiment, we compare the cosine similarity and L2-norm distance between real and fake data according to learning progress of GAN model. Experiments were performed by randomly sampling 30 fake image sequences and real image sequences at each step and averaging the similarities of one-to-one comparison. Figure 3 shows the result of similarity experiment. As learning continues, the cosine similarity increases and the L2-norm distance decreases. Also, convergence appeared after about 3000 steps. This means that the GAN generator stably generates fake data similar to real data. Figure 4 indicates that one of the image sequences generated by the GAN is compared with the actual data. It can be seen that the generated data (right) well describes the overall characteristics of the real data (left).

5 Conclusion and Future Work

This paper proposes the VAD classifier modeling method in a supervised learning manner that can learn only with a small amount of labeled data. The basic idea is to overcome the lack of labeled data through transfer learning and to make the underlying model by learning GAN using LRCN as discriminator. As a result of comparison with existing machine learning methods of supervised learning manner through 10-fold cross validation, the proposed method showed the best performance.

However, robustness to unseen data is still not resolved. In order to determine the severity of the problem, we conducted the experiment according to the following steps. First, exclude one of the 5 types of anomaly in the USCD-ped1 from the training set. Second, check the classification performance of unseen anomaly of the learned model. As a result, only three anomaly types showed more than 50% accuracy. Even in all experiments, the model had an accuracy of less than 60%. Therefore, we will study a model that can classify unseen image sequences well in VAD problems with supervised learning manner.

Acknowledgement. This work was supported by the ICT R&D program of MSIP/IITP. [2017-0-00306, Development of Multimodal Sensor-based Intelligent Systems for Outdoor Surveillance Robots].

References

1. Fleck, S., Straßer, W.: Privacy sensitive surveillance for assisted living—a smart camera approach. In: Nakashima, H., Aghajan, H., Augusto, J.C. (eds.) *Handbook of Ambient Intelligence and Smart Environments*, pp. 985–1014. Springer, Boston (2010). https://doi.org/10.1007/978-0-387-93808-0_37
2. Cong, Y., Yuan, J., Liu, J.: Sparse reconstruction cost for abnormal event detection. In: *Computer Vision and Pattern Recognition*, pp. 3449–3456 (2011)
3. Xu, D., Yan, Y., Ricci, E., Sebe, N.: Detecting anomalous events in videos by learning deep representations of appearance and motion. *Comput. Vis. Image Underst.* **156**, 117–127 (2017)
4. Tomé, A., Salgado, L.: Anomaly detection in crowded scenarios using local and global Gaussian mixture models. In: Blanc-Talon, J., Penne, R., Philips, W., Popescu, D., Scheunders, P. (eds.) *ACIVS 2017*. LNCS, vol. 10617, pp. 363–374. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70353-4_31
5. Popoola, O.P., Wang, K.: Video-based abnormal human behavior recognition—a review. *IEEE Trans. Syst. Man Cybern. Part C* **42**(6), 865–878 (2012)
6. Kiran, B.R., Thomas, D.M., Parakkal, R.: An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. [arXiv:1801.03149](https://arxiv.org/abs/1801.03149) (2018)
7. Li, T., Chang, H., Wang, M., Ni, B., Hong, R., Yan, S.: Crowded scene analysis: a survey. *IEEE Trans. Circuits Syst. Video Technol.* **25**(3), 367–386 (2015)
8. Donahue, J., et al.: Long-term recurrent convolutional networks for visual recognition and description. In: *Computer Vision Pattern Recognition*, pp. 2625–2634 (2015)
9. Ballas, N., Yao, L., Pal, C., Courville, A.: Delving deeper into convolutional networks for learning video representations. [arXiv:1511.06432](https://arxiv.org/abs/1511.06432) (2015)

10. Hasan, M., Choi, J., Neumann, J., Roy-Chowdhury, A.K., Davis, L.S.: Learning temporal regularity in video sequences. *Computer Vision and Pattern Recognition*, pp. 733–742 (2016)
11. Chong, Y.S., Tay, Y.H.: Abnormal event detection in videos using spatiotemporal autoencoder. In: Cong, F., Leung, A., Wei, Q. (eds.) *ISNN 2017*. LNCS, vol. 10262, pp. 189–196. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59081-3_23
12. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using LSTMs. In: *International Conference on Machine Learning*, pp. 843–852 (2015)
13. Pan, S.J., Yang, Q.: A survey on transfer learning. *Knowl. Data Eng.* **22**(10), 1345–1359 (2010)
14. Goodfellow, I., et al.: Generative adversarial nets. In: *Neural Information Processing Systems*, pp. 2672–2680 (2014)
15. Mehran, R., Oyama, A., Shah, M.: Abnormal crowd behavior detection using social force model. In: *Computer vision and pattern recognition*, pp. 935–942 (2009)
16. Zhao, B., Fei-Fei, L., Xing, E.P.: Online detection of unusual events in videos via dynamic sparse coding. In: *Computer Vision and Pattern Recognition*, pp. 3313–3320 (2011)
17. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social LSTM: human trajectory prediction in crowded spaces. In: *Computer Vision and Pattern Recognition*, pp. 961–971 (2016)
18. Luo, W., Liu, W., Gao, S.: A revisit of sparse coding based anomaly detection in stacked RNN framework. In: *ICCV*, vol. 1, no. 2, pp. 3 (2017)
19. Medel, J.R., Savakis, A.: Anomaly detection in video using predictive convolutional long short-term memory networks. [arXiv:1612.00390](https://arxiv.org/abs/1612.00390) (2016)
20. Schlegl, T., Seeböck, P., Waldstein, S.M., Schmidt-Erfurth, U., Langs, G.: Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In: Niethammer, M., Styner, M., Aylward, S., Zhu, H., Oguz, I., Yap, P.-T., Shen, D. (eds.) *IPMI 2017*. LNCS, vol. 10265, pp. 146–157. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59050-9_12
21. Zenati, H., Foo, C.S., Lecouat, B., Manek, G., Chandrasekhar, V.R.: Efficient GAN-based anomaly detection. [arXiv:1802.06222](https://arxiv.org/abs/1802.06222) (2018)
22. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. [arXiv:1511.06434](https://arxiv.org/abs/1511.06434) (2015)
23. Mahadevan, V., Li, W.-X., Bhalodia, V., Vasconcelos, N.: Anomaly detection in crowded scenes. In: *Computer Vision and Pattern Recognition*, pp. 1975–1981 (2010)
24. Catania, C.A., Bromberg, F., Garino, C.G.: An autonomous labeling approach to support vector machines algorithms for network traffic anomaly detection. *Expert Syst. Appl.* **39**(2), 1822–1829 (2012)



Learning Optimal Q-Function Using Deep Boltzmann Machine for Reliable Trading of Cryptocurrency

Seok-Jun Bu and Sung-Bae Cho^(✉)

Department of Computer Science, Yonsei University, Seoul, Republic of Korea
{sjbuhun, sbcho}@yonsei.ac.kr

Abstract. The explosive price volatility from the end of 2017 to January 2018 shows that bitcoin is a high risk asset. The deep reinforcement algorithm is straightforward idea for directly outputs the market management actions to achieve higher profit instead of higher price-prediction accuracy. However, existing deep reinforcement learning algorithms including Q-learning are also limited to problems caused by enormous searching space. We propose a combination of double Q-network and unsupervised pre-training using Deep Boltzmann Machine (DBM) to generate and enhance the optimal Q-function in cryptocurrency trading. We obtained the profit of 2,686% in simulation, whereas the best conventional model had that of 2,087% for the same period of test. In addition, our model records 24% of profit while market price significantly drops by -64%.

Keywords: Deep reinforcement learning · Q-network
Deep Boltzmann Machine · Portfolio management

1 Introduction

Bitcoin is a peer-to-peer, decentralized electronic cash protocol [1]. The explosive price volatility from the end of 2017 to January 2018 shows that bitcoin is a high risk asset that is insufficient to function as a currency. As shown in Fig. 1, the price has plummeted to the \$6,000 from \$13,000 and shows the extreme volatility of crypto-currency investment. Despite some potential threats, the cyptocurrency trading has become more active and attracts more attention both from the business and academia. In particular, Massive time-series data collected every single minute and single transaction from a market valued at \$820 billion is attractive in terms of volume and volatility.

The term portfolio management is the decision making process of allocating an amount of asset into different financial investment products to maximize the profit and minimize the risk [2]. Many of the existing portfolio management methods that applying machine learning algorithm defines some rules based on domain knowledge, but some of the human-defined rules or even expert domain knowledge are not sufficient to deal with the market dynamics.

Besides, it is impossible to tune the parameters of the model with the supervised-manner in order to derive the optimal action from the posterior probability distribution



Fig. 1. The volatility of the bitcoin price in the past years

of the state of the cryptocurrency market. In this paper, we propose the combination of two methods: Double Q-network and Deep Boltzmann Machine (DBM).

On the one hand, reinforcement learning using Q-network is a well-known method of learning that acts to maximize some measure of future payoff or reward [3]. It derives the optimal solution for trading based on Q-network that outputs adequate action from specific state. Double Q-network is a method designed to cope with non-stationary problems that arise when using only one neural network [4]. On the other hand, unsupervised learning using DBM is an effective method of estimating posterior probability distribution [5]. DBM is a method for modeling the prior distribution of the hidden layer on the input values of the neural network [6]. In order to find the optimal action for the cryptocurrency market state, the parameters of the neural network are tuned in an enormous search space. We propose the encoding network that is pre-trained with market states to reduce the search space.

With the combination of double Q-network and DBM, our trading method records 2,686% of profit in simulation while existing best model records 2,087% for same test period without domain knowledge or human-generated rules. Remarkably, our model for the same test periods as the existing models, including deep-learning based models. To analysis our model, we visualized the output decisions using t-Stochastic Neighbor Embedding (t-SNE) algorithm.

The remainder of this paper is organized as follows. In Sect. 2, we review existing trading algorithms or models based on machine learning methods and clarify the contributions of this paper by discussing the differences. Section 3 explains how the market history is encoded and modeled using double Q-network and DBM pre-training algorithm. The performance of our model is evaluated in Sect. 4 through various experiments, including visualizations of the decisions the model made, measurements of performance and comparisons with existing algorithms.

2 Related Works

In this section, we introduce various works based on machine learning methods for comparison with the proposed trading agent. Almost most of the cryptocurrency trading research was done in late 2010, several studies were included to introduce the basic framework of the study.

Huang et al. used a basic machine learning algorithm to model the weekly volatility of the stock market [7]. Although the domains are different, the weekly price prediction

Table 1. Related works on financial trading using machine learning algorithm

Authors	Method	Domain
Huang [7]	SVM	Stock price prediction
Schumaker [8]	SVM	Stock price prediction
Patel [9]	NB, RF, SVM, NN	Stock price prediction
McNally [10]	ARIMA, RNN, LSTM	Cryptocurrency price prediction
Bell [13]	Wavelet, SVM	Cryptocurrency trading
Zbikowski [14]	EMA, SVM	Cryptocurrency trading
Jiang [12]	DQN(CNN)	Cryptocurrency trading

also significant in terms of feasibility. Schumaker et al. discuss about the necessity of external information to predict stock market [8]. They categorized the difficulty of price prediction into fundamental and technical aspect and modeled stock market with external text data from web source. The information from quarterly reports or breaking news stories made the price prediction more accurate. Patel et al. applied various machine learning methods to predict stock price and compared them [9]. Due to the uncertainty of the market fluctuation, they achieved under 80% of classification accuracy and showed the random forest algorithm is effective with short-term prediction.

In the late of 2010, a trading agent that includes existing prediction engine was studied in cryptocurrency domain, as well as stock market domain. McNally et al. applied wavelet transform to encode cryptocurrency history and modeled the encoded feature using deep learning algorithms including LSTM [10]. They achieved about 50% of classification accuracy, but contributed to the modeling the sequence of cryptocurrency price as encoded images using wavelet transformation. Amjad et al. proposed the bitcoin trading agent based on predicted price [11].

Among various studies suggesting a agent for modeling cryptocurrency markets and making optimal decision or action, our study using double Q-network and pre-training using DBM was inspired by the study of Jiang et al. [12]. Unlike previous approach, Jiang et al. do not include a prediction engine inside the agent. The Q-network directly map the input state into output action to deal with two problems. The first reason is to exclude the domain knowledge of the person, and the second is high accuracy in predicting price movement is usually difficult to achieve. Table 1 contains a summary of the methods discussed above.

3 Proposed Method

In this section, we present the architecture of the proposed trading agent that makes optimal decision to states by combining double Q-network and encoding network. The trading agent consists mainly of three components: the agent that in the form of two neural networks are connected in series, the unsupervised learning module and the environment.

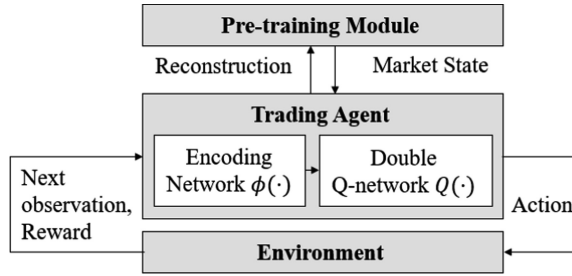


Fig. 2. Main components of the trading agent

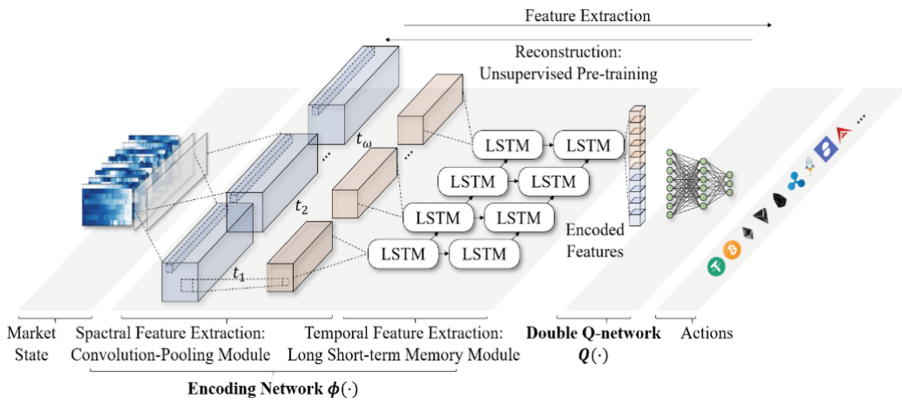


Fig. 3. Deep neural architecture of trading agent consisting of pre-trained encoding network and double Q-network

3.1 Overview

The proposed cryptocurrency trading agent consists of three main components, as shown in Fig. 2. As the first component, The purpose of the encoding network and the double Q-network is to map the input state to appropriate action directly. Figure 3 shows the architecture of the encoding network and the double Q-network in detail. Compared to conventional Q-learning algorithms use Q-tables to map states and actions [15], the generalization performance of double Q-network can reduce the computational complexity. The encoding network consists of a Convolutional Neural Network (CNN) which has a strengths in a wide range of vision fields [16], and a Long Short-Term Memory (LSTM) which has a well known recurrent neural network for time series modeling [17].

The second main component is pre-training learning module. The encoding network is designed to extract the effective information from the state space and deliver it to the double Q-network. Unlike the typical classification problem, by learning to reduce the difference between the probability distribution of the input state and the

posterior probability distribution of the hidden neurons [18], encoding network can be effectively pre-trained.

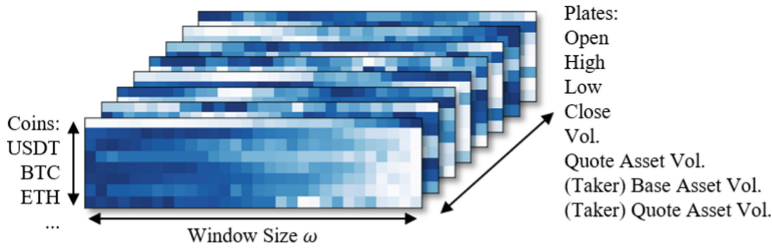


Fig. 4. Preprocessed 3D input state of market history

The last main component is the environment, which is the core of the Q-learning algorithm. For each of action from a double Q-network, the environment updates the current state and sends back to the Q-network and determines the reward for the action taken [19]. We chose eight coins based on trading volume and defined actions as 16-dimensional vectors which mean buying/selling weights for each coin. In order to model the time-series features of price, we preprocessed market state by sliding window as shown in Fig. 4. The market state is represented by 3D-blocks named as history block, with vertical axis (window size ω), horizontal axis (coin type) and depth axis (plates). The reward for the proposed agent is defined as +1 if the asset is increased, 0 if remain still and -1 if it is decreased.

3.2 Unsupervised Pretraining of Encoding Network

The encoding network consists on CNN which models the spatial feature using convolution-pooling operations $\phi_c(\cdot)$, $\phi_p(\cdot)$, and LSTM which models the temporal feature using a gate operation and a recurrent loop $\phi_s(\cdot)$ [20]. For every history block $X = \{x_1, \dots, x_n\}$, the encoding function $\phi(\cdot)$ is defined as below with step t :

$$\phi(x_t) = \phi_p(\phi_c(x_t)) \tag{1}$$

The convolution operation, which preserves the spatial relationships between features by learning filters that extract correlations, is known to reduce the translational variance between features. The hidden correlations between features in cryptocurrencies and its financial attributes are modeled as a feature-map through emphasis or distortion during the convolution operation. Given t th input state, the encoding networks performs the convolution operation $\phi_c(\cdot)$ using $m \times m \times m$ sized filter w :

$$\phi_c^l(x_t) = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \sum_{c=0}^{m-1} w_{abc} x_{(i+a)(j+b)(k+c)}^{l-1} \tag{2}$$

The summary statistic of nearby outputs is derived from $\phi_p(\cdot)$ by max-pooling operation [21]. Because the dimension of the output vectors from the convolutional

layer is increased by the number of convolutional filters, it must be carefully controlled. Pooling refers to a dimension reduction process used in CNN in order to impose a capacity bottleneck and facilitate faster computation. The max-pooling operation has effects on feature selection and dimension reduction under $k \times k \times k$ sized area with pooling stride τ :

$$\phi_p^l(x_t) = \max x_{t_{ijk \times \tau}}^{l-1} \tag{3}$$

The purpose of encoding network is to extract the effective information from the state space and deliver it to the double Q-network. Since the size of the available state space, it is difficult to find optimal parameters of encoding network being connected in series with the double Q-network. Focusing on the markov property of history block, we formalize the window size parameter ω and markov chain of sequence of price as below:

$$p(x_t|x_{t-1}, \dots, x_1) = p(x_t|x_{t-1}, \dots, x_{t-\omega}) \tag{4}$$

We define the energy $E(x_t, h)$, partition constant z and parameterize the joint probability x, h as $p(x_t, h)$ as shown below:

$$E(x_t, h) = -\phi_s(\phi_p(\phi_c(x_t))), z = \sum_{ij} e^{-E(x_t, h_j)} \tag{5}$$

$$p(x_t) = \sum_i p(x_{ti}, h) = \sum_i \frac{e^{-E(x_t, h)}}{z} \tag{6}$$

After we define the probability distribution of the hidden layer and the input state x_t above, we define the loss of the encoding network L_ϕ using the Kullback-Leibler divergence Δ_{KL} between the observed distribution:

$$L_\phi = \sum_{t=1}^N L(\theta|x_t) = \Delta_{KL}(x_t, p(x_t|\theta)) = \left(\left(-\sum x_t \ln p(x_t|\theta) \right) - \left(-\sum x_t \ln x_t \right) \right) \tag{7}$$

3.3 Double Q-Network and Market Environment

The objective of Q-network training based on reinforcement learning is to expect to output appropriate action in response to the generalization performance of the neural network. The minimizing process of the loss function L_Q using stochastic gradient descent algorithm considers the reward of next step. To cope with the non-stationary problem, Hasselt et al. proposed the double Q-network that copies the Q-network into two networks and fix the target of Q-network [4]. We formalized out objective as a loss of double Q-network L_Q , where reward r and decaying hyperparameter γ :

$$L_Q = \min_{\theta} \sum_{t=0}^n \left[\hat{Q}(\phi(x_t|\theta)) - \left(r_t + \gamma \hat{Q}(\phi(x_{t+1})|\bar{\theta}) \right) \right]^2 \tag{8}$$

The details of the training algorithm used are presented in Fig. 5. In each episode, Q-network takes a different action by the random probability ϵ even if it is the same step. Since the hyperparameter ϵ , also called exploration rate, can be the method to find a new buying/selling strategy that can raise profit but also can be the pithole of the entire method. Once the training process starts, the encoding network $\phi(\cdot)$ pre-trained by unsupervised manner. For the steps that make up each episode, Q-network repeatedly receives the state and outputs an action vector. Each element of the action vector represents the weight of the asset movement. In this paper, we have selected eight coins based on volume, so we output a 16-dimensional action vector consisting of buy/sell actions for each coin.

```

Pretrain  $\phi(\cdot)$  by unsupervised learning
Initialize replay memory  $D$  to capacity  $d$ 
Initialize double Q-network  $Q$  and target-network  $\hat{Q}$ 
Initialize exploration rate  $\epsilon$ 
for episode  $e = 1, E$  do
  Initialize score
  for step  $t = 1, T$  do
    Preprocess state  $\phi(s_t)$ 
    Set  $a_t = \begin{cases} \text{Random Sequences,} & \text{if } \epsilon_t \leq \epsilon. \\ Q(\phi(s_t)|\theta), & \text{otherwise.} \end{cases}$ 
    Execute  $a_t$  and observe reward  $r_t$ , next state  $s_{t+1}$ 
    Store  $(s_t, a_t, r_t, s_{t+1})$  in  $D$ 
    Sample random minibatch if step>memory
    Set  $y_t = \begin{cases} r_j, & \text{if episode terminates.} \\ r_j + \hat{Q}(\phi(s_j|\bar{\theta})), & \text{otherwise.} \end{cases}$ 
    Perform a gradient descent on  $L_Q$ 
    Set  $\hat{Q} = Q$  for every  $c$  steps
  end for
end for

```

Fig. 5. The training algorithm of proposed double Q-network

Putting it all together, the main contribution of our study is to propose a combination of the two existing methods that can efficiently model the history of financial or derivatives markets. First, we introduce a framework of existing research to map the vast state space and action space corresponding the cryptocurrency market. Second, we modified and adapted the existing DBM pre-training algorithm to reduce the parameter space of encoding network. In addition, we collected a large amount of cryptocurrency market historical data and preprocessed.

4 Experimental Results

In this section, we evaluated our agent through various experiments, including measurements of performance, comparisons with existing algorithms, parameter optimization and visualizations of the decisions the agent made. While short trades that are difficult for humans to understand, the final profit is the highest among the existing algorithms. To cope with the high computational complexity that is proportional to the amount of historical data of the cryptocurrency transactions, we used four NVIDIA GTX1080-Ti to learn a large number of agents.

4.1 Comparisons with Existing Methods

For quantitative comparison with existing studies, we conducted the back-test during test period 2016/05/14-2016/07/03. We selected bitcoin and seven altcoins which had the highest trading volumes during the period, as assets. In order to compare performance intuitively, the score was defined as the ratio between total value after investment and initial value.

Table 2. Score and risk measure comparison with other algorithms

Algorithm	Score	Sharpe ratio	Maximum drawdown
Uniform buy and hold	0.8760	-1.5413	0.3820
Best single asset	1.3776	1.1257	0.2883
Universal portfolio [22]	1.0484	-1.0110	0.3309
Online neutron step [23]	2.6482	1.0458	0.2787
PAMR [24]	21.8728	0.0062	0.3530
DQN (CNN) [12]	16.3053	0.0368	0.2960
DQN (Ours)	27.8684	0.0027	0.4627

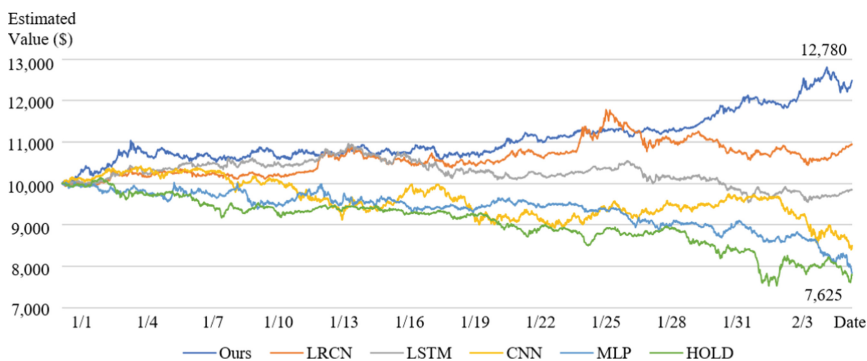


Fig. 6. The short-periodic profit comparisons with other deep models

Two financial measures, sharpe ratio and maximum drawdown, are used to evaluate the risk of strategies. Sharpe ratio are used to average return earned in excess of the risk-free rate per unit of volatility or total risk, defined as $Sharp_r = \frac{\widehat{r}_p - r_f}{\sigma_p}$ where \widehat{r}_p is expected portfolio return, r_f is risk free rate, σ_p is the standard deviation for portfolio. Generally, higher sharpe ratios are known to guarantee higher returns for the same risk level. Maximum drawdown is defined as the maximum distance from a peak to portfolio, and it can be used as an measure of the rate of change of price.

Table 2 summarizes the change in assets for the initial \$10,000 asset. Our agent outperformed among existing methods by achieving 27.86 of score. But in terms of risk, our agent has the volatility about price fluctuation. The Online Newton Step algorithm has the largest sharpe ratio and smallest maximum drawdown, indicating the algorithm is most stable.

To verify the performance of our agent against the latest cryptocurrency market and evaluate the effect of unsupervised learning, we compared our agent to other double Q-network based on deep learning methods as shown in Fig. 6. We conducted a back-test during 2018/01/01-2018/01/31. Long-term Recurrent Convolutional Network (LRCN) is the same architecture as the proposed agent but except pre-training algorithm. Remarkably, our proposed model achieved 1.27 score despite of the crash of bitcoin price while the investors who only bought bitcoin had a -23.75% of loss.

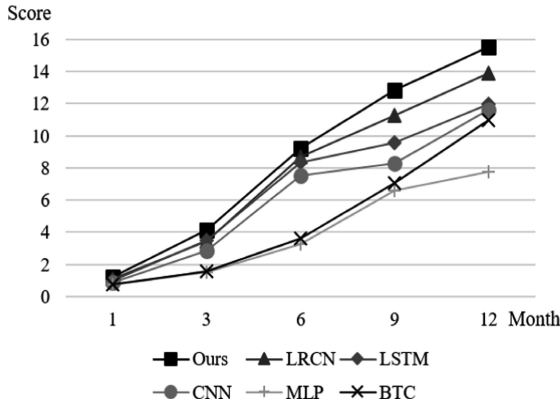


Fig. 7. The long-periodic profit comparisons with other deep models

Figure 7 represents the long-periodic profit of our agent, which is trained and tested up to 1 year. Since cryptocurrency market and price grows ten times between early in 2017 to 2018, investors who invested only in bitcoin also gained a 10 times of profit. Double Q-network with simple neural network recorded lower profit than bitcoin growth rate, which is similar behavior of individual investors suffering losses for no apparent reason.

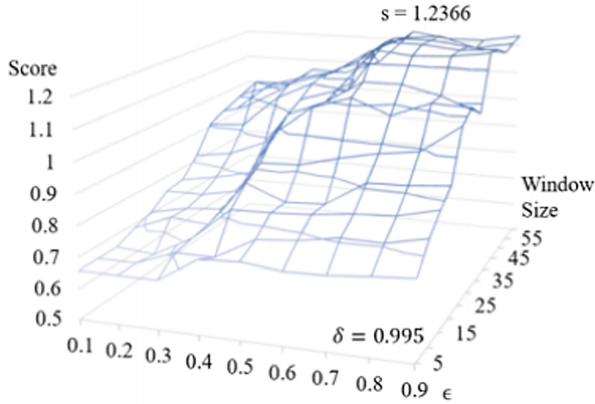


Fig. 8. The grid search to find optimal parameter for trading agent (Color figure online)

4.2 Parameter Optimization

Since the proposed trading agent combines double Q-network framework with DBM, various types of parameters can be adjusted. Typically, these parameters include the probabilistic factors that are used for exploration of Q-networks, as well as the hyperparameters for deep learning models such as the number of training iterations and the size of the convolution-pooling layer or LSTM layers. We used a traditional grid search to perform parameter optimization for trading agent (Fig. 9).

A grid search is simply an exhaustive search of a manually specified subset of the parameter space. The parameters to be optimized were set as the exploration rate, which is known to be responsible to escape the local minima in each episodes, and the window size of the market history in blocks, which is one of the major factors

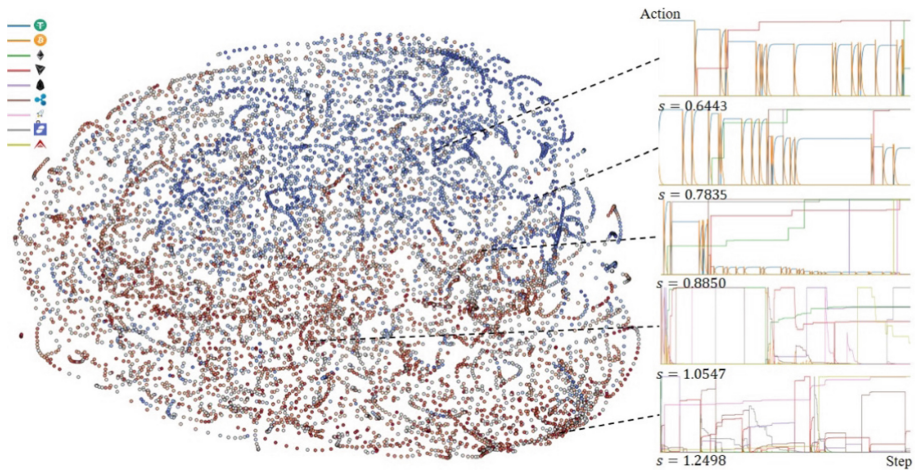


Fig. 9. The visualization of the states and actions

influencing the performance of sequence modeling. Figure 8 presents the performance of the double Q-network combined with DBM based on hyperparameters. After we fix the remaining parameters, we conducted a back-test for 2017/07/01-2017/07/31. As expected, the window contains the temporal information. The trading agent failed to map the appropriate action when the window is less than 15 min long.

4.3 Analysis of Action Vector

Figure 8 presents the entire dataset after the t-SNE algorithm is applied to the pre-processed states. t-SNE algorithm is a dimension reduction technique that is capable of retaining the local structures in data with while revealing important global structures [25]. Each of point is a single history block and the blue color represents the price drop in the block. The cluster of blue points and red points indicates the unsupervised training of encoding network was effective to reduce the search space.

On the right side of the figure, we visualized the action vector that occurred in the episode containing the state. From top to bottom, as learning progresses, the action decision of the trading agent is increasingly complex. The decision to buy and sell for the very top simply a few coins evolves into a complex decision that can not be interpreted at the very bottom.

5 Conclusion

We proposed a combination of double Q-network and DBM to generate and enhance the optimal Q-function in cryptocurrency trading. As the first component, The purpose of the encoding network and the double Q-network, in which two networks are connected in series, is to map the input state to appropriate action directly. The second main component is unsupervised learning module that pretrains the encoding network using modified DBM pre-training algorithm. We evaluated our agent through various experiments, including measurements of performance and risk, comparisons with existing algorithms, parameter optimization and visualization of the decisions. We achieved the highest profit among the existing models and deep learning models. Surprisingly, even when bitcoin price plummeted to -40% , the proposed agent achieved 20% of profit.

The main contribution of our research is to propose a combination of the two existing methods that can effectively model the history of financial or cryptocurrency markets. We introduce a framework of existing research to map the state into action corresponding the cryptocurrency market. We modified the existing DBM training algorithm and adapt to reduce the parameter space of encoding network. In addition, we collected and preprocessed 210.24 million of cryptocurrency trading records.

Since empirical results indicates that our agent is more risky than existing algorithms, Future work will include a stabilization process of volatile decision that our agent made by redesigning the Q-network. Secondly, we will enhance the performance of our agent by attaching generative model. Based on the generative network that generates and classifies virtual trade record, we will improve our Q-network more precisely.

Acknowledgements. This research was supported by Korea Electric Power Corporation. (Grant number:R18XA05).

References

1. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system (2008)
2. Agarwal, A., Hazan, E., Kale, S., Schapire, R.E.: Algorithms for portfolio management based on the Newton method. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 9–16. ACM (2006)
3. Mnih, V., et al.: Human-level control through deep reinforcement learning. *Nature* **518**, 529 (2015)
4. Van Hasselt, H., Guez, A., Silver, D.: Deep reinforcement learning with double q-learning. In: AAAI, vol. 16, pp. 2094–2100 (2016)
5. Lee, H., Grosse, R., Ranganath, R., Ng, A.Y.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 609–616 (2009)
6. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. *Neural Comput.* **18**, 1527–1554 (2006)
7. Huang, W., Nakamori, Y., Wang, S.Y.: Forecasting stock market movement direction with support vector machine. *Comput. Oper. Res.* **32**, 2513–2522 (2005)
8. Schumaker, R.P., Chen, H.: Textual analysis of stock market prediction using breaking financial news: the AZFin text system. *ACM Trans. Inf. Syst.* **27**, 12 (2009)
9. Patel, J., Shah, S., Thakkar, P., Kotecha, K.: Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Syst. Appl.* **42**, 259–268 (2015)
10. McNally, S.: Predicting the Price of Bitcoin using Machine Learning. National College of Ireland (2016)
11. Amjad, M., Shah, D.: Trading bitcoin and online time series prediction. In: NIPS 2016 Time Series Workshop, pp. 1–15 (2017)
12. Jiang, Z., Liang, J.: Cryptocurrency portfolio management with deep reinforcement learning. In: Intelligent Systems Conference, pp. 905–913 (2017)
13. Bell, T.: Bitcoin Trading Agents. University of Southampton (2016)
14. Żbikowski, K.: Application of machine learning algorithms for bitcoin automated trading. In: Ryzko, D., Gawrysiak, P., Kryszkiewicz, M., Rybiński, H. (eds.) *Machine Intelligence and Big Data in Industry*. SBD, vol. 19, pp. 161–168. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-30315-4_14
15. Tesauro, G.: Extending Q-learning to general adaptive multi-agent systems. In: *Advances in Neural Information Processing Systems*, pp. 871–878 (2004)
16. Pinheiro, P.H., Collobert, R.: Recurrent convolutional neural networks for scene labeling. In: International Conference on Machine Learning, pp. 82–90 (2014)
17. Sainath, T.N., Vinyals, O., Senior, A., Sak, H.: Convolutional, long short-term memory, fully connected deep neural networks. In: *Acoustics, Speech and Signal Processing*, pp. 4580–4584 (2015)
18. Ren, Y., Wu, Y.: Convolutional deep belief networks for feature extraction of EEG signal. In: International Joint Conference on Neural Networks, pp. 2850–2853 (2014)
19. Lample, G., Chaplot, D.S.: Playing FPS games with deep reinforcement learning. In: AAAI, pp. 2140–2146 (2017)

20. Donahue, J., et al.: Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2625–2634 (2015)
21. Bu, S.-J., Cho, S.-B.: A hybrid system of deep learning and learning classifier system for database intrusion detection. In: Martínez de Pisón, F.J., Urraca, R., Quintián, H., Corchado, E. (eds.) HAIS 2017. LNCS (LNAI), vol. 10334, pp. 615–625. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59650-1_52
22. Cover, T.M.: Universal portfolios. In: The Kelly Capital Growth Investment Criterion: Theory and Practice, pp. 181–209 (2011)
23. Das, P., Banerjee, A.: Meta optimization and its applications to portfolio selection. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1163–1171 (2011)
24. Li, B., Zhao, P., Hoi, S.C., Gopalkrishnan, V.: PAMR: passive aggressive mean reversion strategy for portfolio selection. *Mach. Learn.* **87**, 221–258 (2012)
25. Maaten, L.V.D., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**, 2579–2606 (2008)



Predicting the Household Power Consumption Using CNN-LSTM Hybrid Networks

Tae-Young Kim and Sung-Bae Cho^(✉)

Department of Computer Science, Yonsei University, Seoul, Republic of Korea
{taeyoungkim, sbcho}@yonsei.ac.kr

Abstract. Prediction of power consumption is an integral part of the operation and planning of the electricity supply company. In terms of power supply and demand, For the stable supply of electricity, the reserve power must be prepared.

However, it is necessary to predict electricity demand because electricity is difficult to store. In this paper, we propose a CNN-LSTM hybrid network that can extract spatio-temporal information to effectively predict the house power consumption. Experiments have shown that CNN-LSTM hybrid networks, which linearly combine convolutional neural network (CNN), long short-term memory (LSTM) and deep neural network (DNN), can extract irregular features of electric power consumption. The CNN layer is used to reduce the spectrum of spatial information, the LSTM layer is suitable for modeling temporal information, the DNN layer generates a predicted time series. The CNN-LSTM hybrid approach almost completely predicts power consumption. Finally, the CNN-LSTM hybrid method achieves higher root mean square error (RMSE) than traditional predictive methods for the individual household power consumption data sets provided by the UCI repository.

Keywords: Power consumption prediction · Deep learning
Convolutional neural network · Long short-term memory

1 Introduction

A power system is a sophisticated system that simultaneously handles demand and supply, balancing demand and supply. Household energy consumption is also steadily increasing as population is increasing and citizens' standard of living improved [1]. However, since electric energy cannot be stored, it is necessary to forecast electricity demand [2]. Energy consumption forecasting is a multivariate time series problem with several variables that determine power consumption [3]. The variables vary according to the user's consumption pattern and affect the power consumption. Figure 1 shows a visualization of the individual household power consumption data set provided by the UCI repository. This data has a total of nine variables and is used as a benchmark for power demand forecasting. Figure 2 shows the trend components for the individual home power consumption data sets. The trend component was extracted using time series decomposition method. This trend is very complex and irregular. Therefore, it is difficult to forecast power demand using existing statistical techniques. In this paper, we propose a CNN-LSTM hybrid neural network that linearly connects CNN and

LSTM to automatically predict the household power consumption. Power consumption, which is a multivariate time series, includes spatial and temporal information. Therefore, the CNN-LSTM hybrid neural network can extract the space-time feature of the power consumption variable to predict the household power consumption.

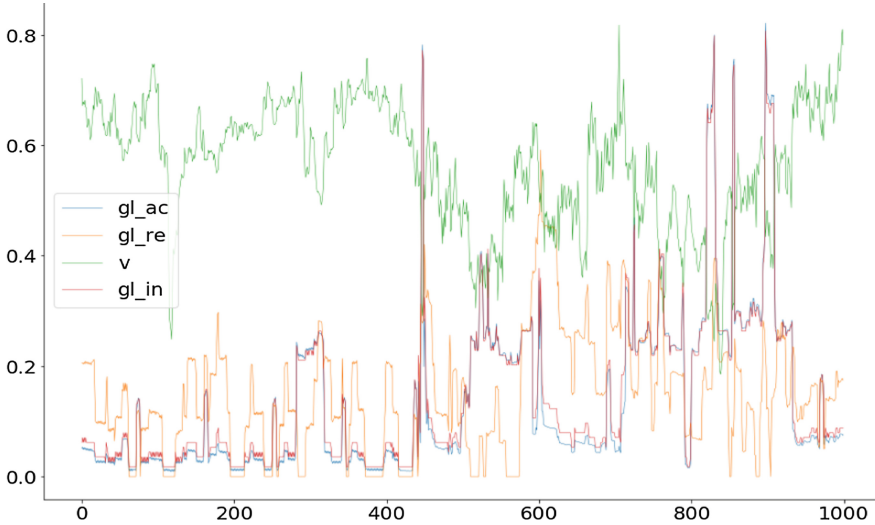


Fig. 1. Attribute of household power consumption data

The proposed CNN-LSTM method reduces the spectrum of time series data by using the convolution and the pooling layer. The output of this CNN layer is passed as input to the LSTM layer to model the temporal information. The output of the LSTM unit is used as an input to the fully connected layer to generate a time series that

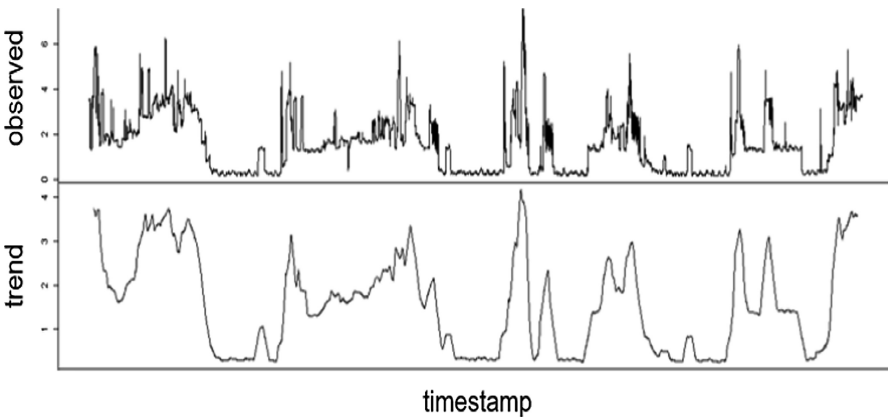


Fig. 2. Trend component for individual home power consumption dataset

predicts power demand. Finally, we tuned some parameters to see if the performance of CNN-LSTM improves. In particular, we analyze the internal output process of the proposed method of extracting spatial features from time series data using CNN. We also compare performance with the LSTM model.

The remainder of this paper is organized as follows. Section 2 discusses related work on time series data prediction. Section 3 describes the proposed CNN-LSTM hybrid network architecture. Section 4 presents experiment settings and results. Section 5 concludes this paper.

2 Related Works

Table 1 shows the related work for processing time series data. Many researchers have studied various feature extraction methods to predict time series data. Methods for extracting features from a time series can be divided into three categories: statistical modeling, spatial information modeling, and temporal information modeling.

Table 1. Related work on time series preprocessing

Category	Author	Year	Method	Description
Statistical modeling	Nychis [4]	2008	Entropy based	Entropy based correlation analysis
	Münz [5]	2007	K-means clustering	Classification using cluster
	Zhang [6]	2006	Random forest	Outlier pattern extraction
Spatial information modeling	Ince [7]	2016	1D-CNN	Sliding window-based feature extraction
	Kiranyaz [8]	2015	1D-CNN	ECG feature classification
	Souza [9]	2014	SVM	Sequence texture mapping
Temporal information modeling	Bontemps [10]	2016	LSTM	Normal signal prediction
	Taylor [11]	2016	LSTM	Prediction and exception calculation
	Malhotra [12]	2015	Stacked LSTM	Using prediction error distribution

Münz et al. use k-means clustering algorithm in time series data to predict time series of irregular patterns. They calculated the center value of the cluster and classified the time series into regular and irregular trend according to the distance [5]. Zhang and Zulkernine used a random forest algorithm to detect outliers in time series data. They attempted to predict irregular features using unsupervised learning [6]. These methods provide high performance when estimating the value of time series data. However, a value having the same distribution as the normal sequence cannot be properly predicted.

Kiranyaz et al. tried to predict the signal by extracting the feature from the patient’s electrocardiogram signal. They split the ECG signal into time fragments and extracted features using 1D CNN [8]. Souza et al. created a local feature by mapping the time series data to a texture. They created texture features using SVM [9]. This method efficiently extracts spatial information from a sequence of complex patterns with noises. Compared to previous research, they get better predictive performance. However, the time information of the time series data is lost by the convolution and the pulling operation.

Bontemps et al. attempted to predict the time-series using LSTM [10]. Malhotra et al. classify irregular signals into sensor data from various equipment using a stacked LSTM structure [12]. They learned the normal signal to the LSTM and then calculated the actual data and error distribution using the predicted signal. These methods can easily predict in time series data with periodicity. However, if the time series data does not have a period, it cannot predict the actual power consumption.

3 Proposed Method

3.1 CNN-LSTM Hybrid Neural Network

The proposed CNN-LSTM hybrid neural network consists of a linear structure of CNN and LSTM layers. Figure 3 shows the structure for predicting household power consumption using the proposed CNN-LSTM. CNN-LSTM model uses time-series data

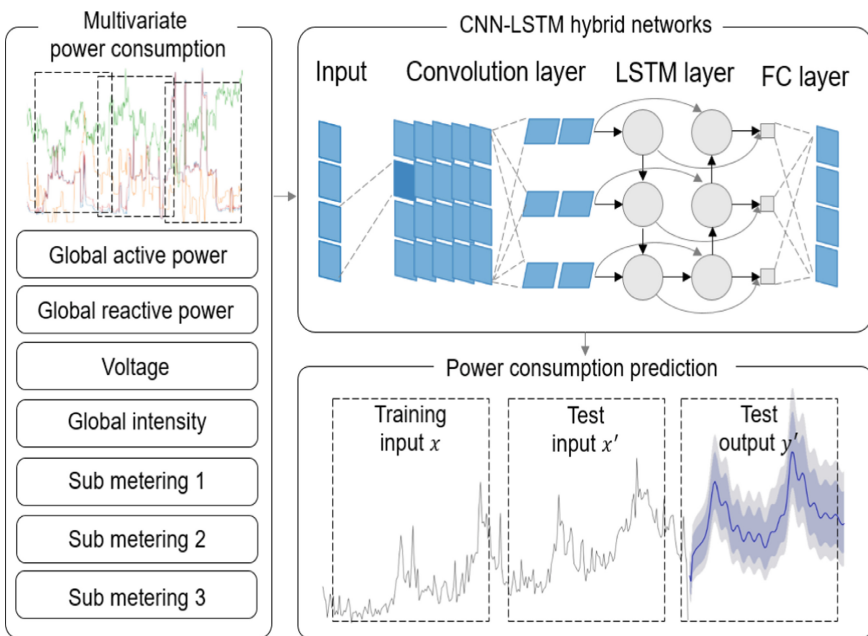


Fig. 3. The proposed household power prediction structure

preprocessed with sliding window algorithm. The preprocessed window extracts spatial features through convolution and pooling operations [13]. Then, temporal feature of the data is modeled by the LSTM. The training model performs household power consumption prediction on the test data in the DNN layer

Assume that $x_i^0 = \{x_1, x_2, \dots, x_n\}$ is a input vector preprocessed by the sliding window algorithm and n is the number of normalized values. Equation 1 passes x_i to the first convolutional layer to derive the output vector y_{ij}^l . This can be calculated using x_{ij}^l , which is the output vector of the previous layer. b_j^l is the bias for the j^{th} convolution kernel, w is the weight of the convolution kernel, m is the index value of each kernel filter, and σ is the activation function. we used ReLu as an activation feature.

$$y_{ij}^l = \sigma \left(\sum_{m=1}^M w_{m,j}^l x_{i+m-1,j}^0 + b_j^l \right) \quad (1)$$

The pooling layer uses max-pooling and is applied independently for each depth slice to reduce the spatial size. Power Consumption Reduces the number of parameters and computation by reducing the space size of the time series. Using the Eq. 2, a pooling layer operation can be performed. R is the input vector whose resolution is reduced by the convolution operation, is a pooling size smaller than the size of y , and T is a stride that determines how far to move the pooled area.

$$p_{ij}^l = \max_{r \in R} y_{i \times T + r, j}^{l-1} \quad (2)$$

In the LSTM layer, multiple memory cells are used to store historical power consumption. Using these cells allows us to remember power consumption for a long time to make it easier to understand. The output value previously calculated on the CNN layer is used as input to the LSTM layer. Each LSTM unit consists of gates and updates the cell according to the active state and is controlled to a continuous value between 0 and 1. The LSTM has three gates, which are inputs, outputs, and forgetting gates. The hidden value of the LSTM cell, h_t is updated every time interval t .

$$i_t = \sigma(W_{pi}p_t + W_{hi}h_{t-1} + W_{ci} \circ c_{t-1} + b_i) \quad (3)$$

$$f_t = \sigma(W_{pf}p_t + W_{hf}h_{t-1} + W_{cf} \circ c_{t-1} + b_f) \quad (4)$$

$$o_t = \sigma(W_{po}p_t + W_{ho}h_{t-1} + W_{co} \circ c_t + b_o) \quad (5)$$

Equations 3, 4, and 5 are denoted i , f , and o , respectively, and are used to calculate input, forget, and output gate activation values. Equations 6 and 7 show the process of calculating the cell state and the hidden state, using notation c and h , respectively. These two values are determined by the input, forget and output gate enable values. σ is calculated using the tanh activation function. The term p_t is used as an input to the memory cell and is the output generated at time t in the previous CNN layer. W is the weight matrix of the LSTM unit, b is the bias vector, and \circ represents Hamad product for cell state calculation.

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma(W_{pc}p_t + W_{hc}h_{t-1} + b_c) \tag{6}$$

$$h_t = o_t \circ \sigma(c_t) \tag{7}$$

Fully-connected layer can help you to predict your home power consumption more easily. These are the last link layer of CNN-LSTM. The output of the LSTM layer is flattened to the vector $h^l = \{h_1, h_2, \dots, h_l\}$, and transferred to the fully-connected layer. Where l is the total number of units in the output of the LSTM. Equation 8 is used in a fully-connected layer. σ is the activation function, w is the weight of the i^{th} node in layer $l - 1$ and j^{th} node in layer l , and b_i^{l-1} is the bias. We can generate predicted time series in a fully-connected layer.

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma(W_{pc}p_t + W_{hc}h_{t-1} + b_c) \tag{8}$$

3.2 CNN LSTM Hybrid Architecture

The CNN-LSTM include various structures depending on the number of convolution, pooling, lstm, and dense layers [14]. It can also change the kernel size, stride size, activation function. By appropriately adjusting the parameters, more features can be extracted from the learning data, affecting the learning speed and can have a significant impact on performance. To change the parameters and determine the CNN-LSTM architecture, we need to understand the characteristics of univariate time series data. We can generate a predicted time series from the input power consumption. The input of the CNN-LSTM is a time slice of length 60. Therefore, 2×1 kernel is used to extract the most suitable features minimizing the loss of information. The convolution and pooling layers are stacked in two layers to update and learn more weights. LSTM uses 64 units to model temporal feature. Hyperbolic tangent is used as an activation function to increase the convergence rate. Table 2 represents the proposed CNN-LSTM architecture.

Table 2. The Proposed CNN-LSTM architecture

Layer	Kernel size	Stride size	# parameter
Convolution	2×1	1	192
Activation (ReLU)	-	-	0
Pooling	2×1	2	0
Convolution	2×1	1	8,256
Activation (ReLU)	-	-	0
Pooling	2×1	2	0
TimeDistributed	-		0
LSTM (64)	-		180,480
Activation (tanh)	-		0
Dense (32)	-		2,080
Dense (60)	-		1,980

4 Experiment and Results

4.1 Individual Household Electric Power Consumption Dataset

In this paper, we use the individual household electric power consumption dataset provided by UCI [15]. This data is a set of data for one generation of power consumption with a one-minute sampling rate over the long term from 2006 to 2010. The data is composed of 9 attributes and the Date (2006/12/16 ~ 2010/11/26), Time (minute), Global_active_power (kW), Global_reactive_power (kW), Voltage (V), Global_intensity (Wh). We use the ‘‘Global Active Power’’ variable among the nine attributes for power demand forecasting. This variable is the average total active power in kWh (kWh). Raw data was not ready to construct the prediction model because some of the values were missing and the recorded time frame was inappropriate. Because of the lack of information, the prediction efficiency of the predictive model may deteriorate. Because the input value for CNN-LSTM is between 0 and 1, we had to pre-process the power station congestion. The values were normalized using Eq. 9. The data consists of a total of 2,075,259 time series data and there are 25,979 missing values. We removed all missing values in the data preprocessing process. We pre-processed multivariate time - series data by the sliding window algorithm.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{9}$$

4.2 Comparison of Loss Between LSTM and CNN-LSTM Model

To verify the usefulness of the proposed method, we compared the learning loss graphs of the LSTM and CNN-LSTM models. The proposed CNN-LSTM hybrid model showed stable learning compared to the existing LSTM method. The left side of Fig. 4 shows that the validation loss of the LSTM model is very unstable. The right side shows the loss of the CNN-LSTM model. It shows unstable in the middle of learning but can be stabilized again.

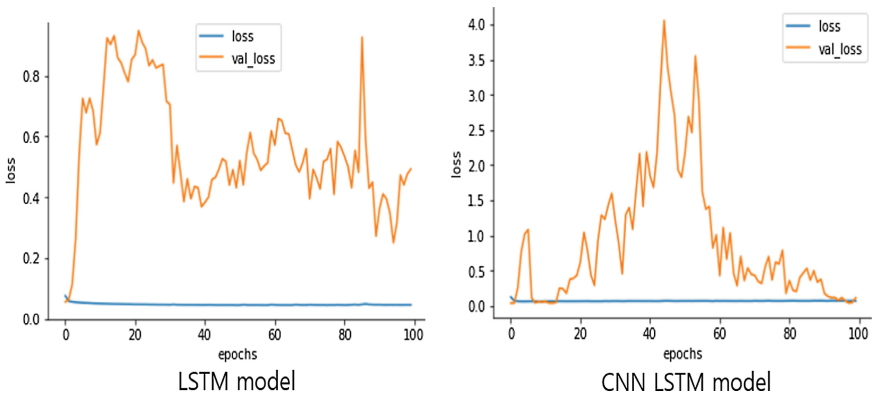


Fig. 4. Loss and validation loss per epoch

4.3 Performance Comparison of Deep Learning Model

Figure 5 shows different machine learning methods and performance comparisons. The proposed CNN-LSTM method achieves higher performance than other models. Root mean square error (RMSE) was used as an evaluation metric of the learning model.

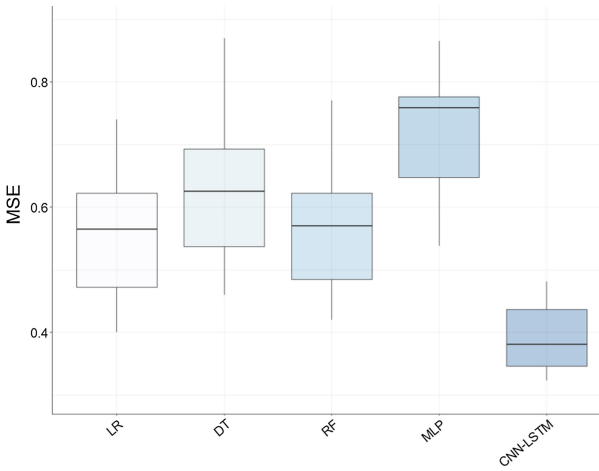


Fig. 5. The RMSE of 10-fold cross validation using other methods

4.4 C-LSTM Model Internal Analysis

We confirmed the operation of the CNN-LSTM through internal visualization. In particular, we can see how the power consumption input changes through the CNN layer. The intermediate outputs were analyzed in C-LSTM neural network using Individual household electric power consumption dataset. Figure 6 shows the outputs

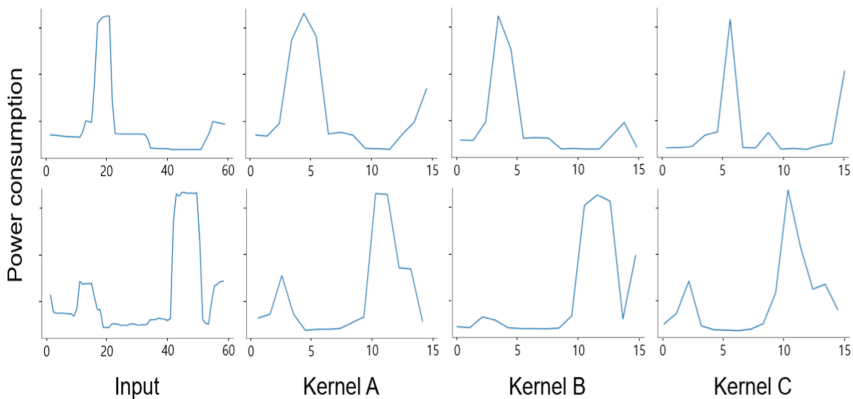


Fig. 6. The CNN-LSTM internal output according to kernel

of the CNN-LSTM layers. Each CNN layer learns to extract the most appropriate features while changing kernel weights. Each layer of CNN reduces the overall length of the input sequence by convolution and pulling operations. However, Fig. 6 confirms that the intermediate output maintains the spatial and temporal feature.

4.5 Visualize Forecasting Results Through CNN-LSTM Hybrid Model

Figure 7 shows a graph that visualizes the predicted results using the CNN-LSTM model. We have confirmed that the power consumption prediction in the individual household electric power consumption dataset has been successful. Predictive results can be visually confirmed similar to the ground truth. In addition, it can be seen that excellent prediction performance is achieved even in situations where the periodicity is not observed. We can visually confirm that the proposed CNN-LSTM model performs well in power consumption prediction. We can confirm that the proposed CNN-LSTM model predicts local features well in power consumption prediction.

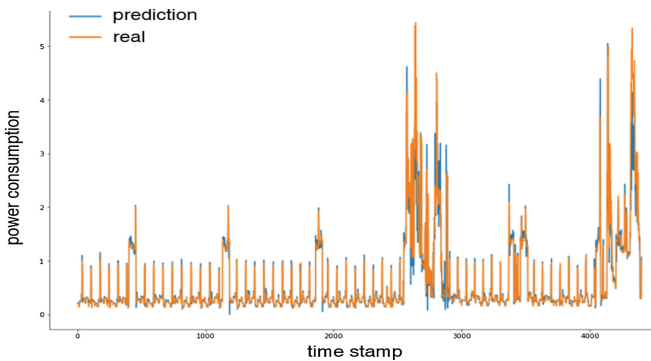


Fig. 7. Graph of predicted results using CNN-LSTM

5 Conclusion

We proposed a CNN-LSTM hybrid architecture to predict power consumption. We have demonstrated usability and excellence by comparing the proposed model with other machine learning methods. We also found an optimal CNN-LSTM prediction model through performance experiments and parameter tuning. This method can quickly and accurately predict the irregular trend of energy consumption in household power consumption dataset. We use the CNN-LSTM method to automatically extract correlations and time information from variables in multivariate time series. We also checked how time series data changes to predict power consumption through CNN-LSTM internal analysis. We also analyzed how time series data predicts power consumption through CNN-LSTM internal analysis. The proposed CNN-LSTM neural network predicts time-series characteristics that were difficult to predict with conventional machine learning methods. However, since the proposed method is pre-

processed by the sliding window algorithm, there is a delay in predicting the actual data. This problem remains a challenge.

Acknowledgement. This research was supported by Korea Electric Power Corporation. (Grant number: R18XA05).

References

1. Cho, S.-B., Yu, J.-M.: Hierarchical modular Bayesian networks for low-power context-aware smartphone. *Neurocomputing* (2017)
2. Kant, G., Sangwan, K.S.: Prediction and optimization of machining parameters for minimizing power consumption and surface roughness in machining. *J. Clean. Prod.* **83**, 151–164 (2014)
3. Hernandez, L., et al.: A survey on electric power demand forecasting: future trends in smart grids, microgrids and smart buildings. *IEEE Commun. Surv. Tutor.* **16**(3), 1460–1495 (2014)
4. Nychis, G., Sekar, V., Andersen, D.G., Kim, H., Zhang, H.: An empirical evaluation of entropy-based traffic anomaly detection. In: *Proceedings of the 8th ACM SIGCOMM Internet Measurement Conference*, pp. 151–156 (2008)
5. Münz, G., Li, S., Carle, G.: Traffic anomaly detection using k-means clustering. In: *GI/ITG Workshop MMBnet* (2007)
6. Zhang, J., Zulkernine, M.: Anomaly based network intrusion detection with unsupervised outlier detection. In: *IEEE International Conference on Communications*, vol. 5, pp. 2388–2393 (2006)
7. Ince, T., Kiranyaz, S., Eren, L., Askar, M., Gabbouj, M.: Real-time motor fault detection by 1-D convolutional neural networks. *IEEE Trans. Ind. Electron.* **63**(11), 7067–7075 (2016)
8. Kiranyaz, S., Ince, T., Gabbouj, M.: Real-time patient-specific ECG classification by 1-D convolutional neural networks. *IEEE Trans. Biomed. Eng.* **63**(3), 664–675 (2016)
9. Souza, V.M., Silva, D.F., Batista, G.E.: Extracting texture features for time series classification. In: *22nd International Conference on Pattern Recognition*, pp. 1425–1430 (2014)
10. Bontemps, L., Cao, V.L., McDermott, J., Le-Khac, N.-A.: Collective anomaly detection based on long short-term memory recurrent neural networks. In: Dang, T.K., Wagner, R., Küng, J., Thoai, N., Takizawa, M., Neuhold, E. (eds.) *FDSE 2016. LNCS*, vol. 10018, pp. 141–152. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48057-2_9
11. Taylor, A., Leblanc, S., Japkowicz, N.: Anomaly detection in automobile control network data with long short-term memory networks. In: *Analytics IEEE International Conference on Data Science and Advanced*, pp. 130–139 (2016)
12. Malhotra, P., Vig, L., Shroff, G., Agarwal, P.: Long short term memory networks for anomaly detection in time series. In: *European Symposium on Artificial Neural Networks, Computational Intelligence*, p. 89 (2015)
13. Ronaó, C.A., Cho, S.B.: Human activity recognition with smartphone sensors using deep learning neural networks. *Expert Syst. Appl.* **59**, 235–244 (2016)
14. Kim, T.Y., Cho, S.-B.: Web traffic anomaly detection using C-LSTM neural networks. *Expert Syst. Appl.* **106**, 66–76 (2018)
15. Loginov, A., Heywood, M.I., Wilson, G.: Benchmarking a coevolutionary streaming classifier under the individual household electric power consumption dataset. In: *IEEE International Joint Conference on Neural Networks*, pp. 2834–2841 (2016)



Thermal Prediction for Immersion Cooling Data Centers Based on Recurrent Neural Networks

Jaime Pérez¹(✉), Sergio Pérez¹, José M. Moya^{1,2}, and Patricia Arroba^{1,2}

¹ Integrated Systems Laboratory, Universidad Politécnica de Madrid,
ETSI Telecomunicación, Avenida Complutense 30,
Madrid 28040, Spain

jaime.perez.sanchez@alumnos.upm.es

² Center for Computational Simulation, Universidad Politécnica de Madrid,
Campus de Montegancedo, Boadilla del Monte,
Madrid 28660, Spain

Abstract. In the data center's scope, current cooling techniques are not very efficient both in terms of energy, consuming up to 40% of the total energy requirements, and in terms of occupied area. This is a critical problem for the development of new smart cities, which require the proliferation of numerous data centers in urban areas, to reduce latency and bandwidth of processing data analytics applications in real time. In this work, we propose a new disruptive solution developed to address this problem, submerging the computing infrastructure in a tank full of a dielectric liquid based on hydro-fluoro-ethers (HFE). Thus, we obtain a passive two phase-cooling system, achieving zero-energy cooling and reducing its area. However, to ensure the maximum heat transfer capacity of the HFE, it is necessary to ensure specific thermal conditions. Making a predictive model is crucial for any system that needs to work around the point of maximum efficiency. Therefore, this research focuses on the implementation of a predictive thermal model, accurate enough to keep the temperature of the cooling system within the maximum efficiency region, under real workload conditions. In this paper, we successfully obtained a predictive thermal model using a neural network architecture based on a Gated Recurrent Unit. This model makes accurate thermal predictions of a real system based on HFE immersion cooling, presenting an average error of 0.75 °C with a prediction window of 1 min.

Keywords: Predictive thermal modeling
Recurrent neural networks · Data center · Immersion cooling

1 Introduction

Due to the proliferation of IoT devices and smart cities, cloud computing will not be longer a valid alternative for managing the volume of data generated by

these applications. Today there are around 23,000 million connected IoT devices, but it is estimated that by 2030 there will be more than 100,000 million [4]. The latency and the bandwidth are now critical metrics so, novel Edge data centers have arisen to bring the computing infrastructure close to the source of the data. Edge data centers, distributed in urban areas, are able to process the data, thus reducing the amount of information that reach the cloud. Traditionally, data centers' computing and cooling consumption represent about the 50% and the 40% of the energy budget [1]. Cooling infrastructures are mainly based on Computer Room Air Conditioner (CRAC) units that are highly inefficient, not only in terms of energy consumption, but also in terms of the area. As Edge data centers will be placed in urban locations, which also present power grid limitations, both the area and the power consumption are also a critical restriction.

In this work we propose an immersion cooling solution, compatible with the Edge requirements, that will help to enable future smart cities. Our approach present a passive two phase-cooling system, achieving zero-energy cooling and reducing its area when compared with traditional CRAC-based data rooms. However, the nature of the hydro-fluoro-ether (HFE) dielectric fluid used to submerge the servers has specific thermal conditions to ensure the maximum heat transfer capacity.

Therefore, this research aims at implementing an accurate predictive thermal model to provide temperature predictions well in advance to keep the temperature of the cooling system within the maximum efficiency region, under real workload conditions. Predictive models are key for any system that needs to work around the point of maximum efficiency, but making a model for complex systems that is accurate and fast can be a difficult challenge. Analytical models help us to represent a solution in a closed form. However, they require the classification of all the parameters that have an impact on the system's performance, thus understanding the complex non-linear relationships between them, which can be a very tedious task in complex systems. On the other hand, Recurrent Artificial Neural Networks, as higher level metaheuristics, have been satisfactorily applied for modeling time series [3].

The **key contribution** of our work is to provide a predictive thermal model using a recurrent neural network architecture, for a disruptive HFE-based immersion cooling solution for Edge data centers. The remainder of this paper is organized as follows: Sect. 2 gives further information on the related work on this topic. Section 3 explains the theory on artificial neural networks for thermal modeling. Our problem description is provided in Sect. 4. Section 5 describes profusely the experimental results. Finally, in Sect. 6 the main conclusions are drawn.

2 Related Work

In this section, we analyze different state-of-the-art approaches for the implementation and use of thermal models. We will also study recent research on modeling time series that will help us to find appropriate techniques for designing predictive models. Traditionally, modeling the temperature in a data center

helps to improve the efficiency of the cooling subsystem by creating strategies that distribute the workload along the computing infrastructure.

Moore et al. [6], use simple heuristics to model the behavior of a data center. They study the temperature variation and tested different algorithms and experiments to gauge the inefficiencies of the system. Their algorithm was able to nearly halve cooling costs when compared to other approaches. Xu et al. [8] present an empirical cooling optimization system an m-block alternating direction method of multipliers (ADMM) algorithm. According to their study, by modeling the temperature and distributing the loads with this algorithm, they provide savings between 15% and 20% for the cooling energy and between 5% and 20% of the overall energy cost. Tang et al. [7] develop a thermal model to minimize temperature peaks in the servers' inlet. They propose a thermal-aware load placement strategy based on the estimations of a heat recirculation model. Their approach offers 30% energy savings for a small simulated data center when compared to previous models.

2.1 Recurrent Artificial Neural Networks for Modeling Time Series

According to Connor et al. [3] Recurrent Neural Networks (RNN) respond very satisfactorily to problems of temporal prediction. This is because, in the RNN, the connections between nodes form a graph directed along a sequence, which allows them to also learn dynamic temporal behavior. And unlike the rest of neural networks, some can use an internal state or memory to process the input sequences. Zaytar et al. [9] present a meteorological prediction model using recurrent neural networks of LSTM type. Using this modeling technique, they are able to predict temperature, humidity and wind speed with errors around 2%. Che et al. [2] propose a multivariable temporal prediction using recurrent neural networks of the GRU type with an average error of around 0.7 and a standard deviation of 0.02. This strategy is especially interesting when working with signals that are noisy and unstable. Research proposed by Kermanshahi [5] predicts the electric power demand of nine industrial facilities using RNN with errors between 0.53% and 2.76%.

We can conclude that modeling the temperature to vary the dynamic load is effective to improve the energy efficiency in conventional cooling systems in data centers. However, previous research works model the temperature in data centers with air-ventilated cooling infrastructures. In this paper we propose a thermal model for a two phase-immersion cooling data center based on an HFE dielectric fluid. For this purpose we use Recurrent Artificial Neural Networks, as they have been satisfactory applied for modeling time series in other research fields with high accuracy.

3 Recurrent Neural Networks for Thermal Modeling

Based on the state-of-the-art presented in Sect. 2, we decided to model our thermal predictions using Artificial Neural Networks (ANNs). These algorithms are

named after the neural networks of animal’s nervous systems because they try to emulate their behavior. ANNs are able to automatically learn complex patterns, correlations and behaviors, analyzing large amounts of information.

The ANNs form a system of links that interconnects the different neurons, which makes them collaborate with each other to produce exit stimuli. Each link has a numerical weight, which is recalculated during the network training to be adapted to the data input, thus being able to “learn”. Thanks to the non-linear activation functions inside the neurons, the networks are able to describe any behavior of the real world, which is precisely non-linear. Some typical examples are the sigmoidal function, the normalized exponential (SoftMax), the hyperbolic tangent (tanh) or the rectified linear (ReLU) among others.

As our models need to predict the temperature over time, we need neural networks with memory, which are called Recurrent Neural Networks (RNNs). RNNs are able to work with temporary data sequences because their neurons have an internal state (memory) to process the input sequences. In this work we use GRU architectures as they have been able to reduce gradient problems and also are better suited for small datasets like the one used in this research (our dataset present five features and 2400 time steps). GRU cells, as shown in Fig. 1, have two gates called *update gate* (z_t) and *reset gate* (r_t). z_t determines the amount of information from the previous state that will affect the current state and the r_t specifies how much information from the past state is forgotten. In our thermal modeling, x_t is the current temperature, while h_{t-1} is the information about previous temperatures, h_t is the future temperature prediction passed to following cells and \tilde{h}_t is the current memory content. W_z , W_r and W are the weights of the parameter matrices respectively as in Eqs. 1-4:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{1}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{2}$$

$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t]) \tag{3}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \tag{4}$$

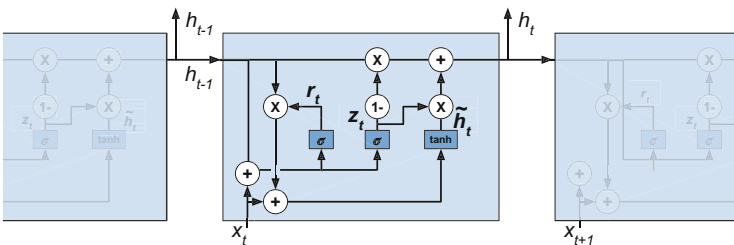


Fig. 1. Scheme of GRU neuron

4 Problem Description

Our prototype, shown in Fig. 2 is a system cooled by passive immersion that consists of a small container filled with dielectric liquid based on Hydro-Fluoro-Ethers, in which we submerge a Raspberry Pi 3 Model B+ cluster. This cluster runs a workload based on data analytics, in particular, a prediction of environmental pollution in the city of Beijing, China¹. This has been chosen to provide a real Edge data center environment. Using a Graphite-based monitoring system² we have compiled a dataset, which includes the temperature ($T_{CPU,x}$), the utilization ($U_{CPU,x}$) and the working frequency ($f_{CPU,x}$) of each Raspberry Pi's CPU (x), collecting data every ten seconds. After that, we use our dataset to design and implement the predictive thermal model.



Fig. 2. HFE-based immersion cooled prototype.

To set the value of the RNN hyperparameters, which are all the tunable settings that the network allows to change, we have first defined a basic model using state-of-the-art examples. We then conduct experiments to set the optimizer (among RMSprop, Adam, Adamax and Nadam) and the best loss function (among MSE, MAE and Binary Cross Entropy) that offer the best results. We set these hyperparameters first as they are considered to be the most independent from the rest. Afterwards, we proceed to make a specific model using the following strategy. First we optimize the neural structure (number of neurons and layers) leaving the rest of the hyperparameters fixed. We perform experiments from smaller structures (1 layer with 2 neurons) to more complex structures (3 layers with 32, 16, and 8 neurons respectively). Finally, we optimize the

¹ archive.ics.uci.edu/ml/datasets/Beijing+PM2.5+Data.

² graphiteapp.org.

hyperparameters involved in model training. We tune the batch size (from 15 to 125) and the number of epochs (from 10 to 250) in steps of 10. Also, we optimize the learning rate (from 0.001 to 0.01), the activation function (Softmax, Relu, Elu, tanh) and the learning decay (from 0.001 to 0.01).

In our problem, the most important hyperparameter is the prediction window. A bigger window means longer reaction time to optimize our system but it also means worse predictions and longer training times. We decide to set it in one minute, because we consider that it gives the system enough time to perform direct or indirect optimization actions. To examine the performance of our prediction approach, we include results with three widely used prediction error metrics: Mean Absolute Error (MAE), Root Mean Square Deviation (RMSD) and Coefficient of determination (R^2), which can be seen in Eqs. 5 to 7 respectively, where y_n is the real measurement, \bar{y} its average value, x_n is the prediction and N is the number of traces in our dataset.

$$MAE = \frac{1}{N} \sum_n |y_n - x_n|, \quad 1 \leq n \leq N \quad (5)$$

$$RMSD = \sqrt{\frac{1}{N} \cdot \sum_n (y_n - x_n)^2} \quad (6)$$

$$R^2 = 1 - \frac{\sum_n (y_n - x_n)^2}{\sum_n (y_n - \bar{y})^2} \quad (7)$$

5 Performance Evaluation

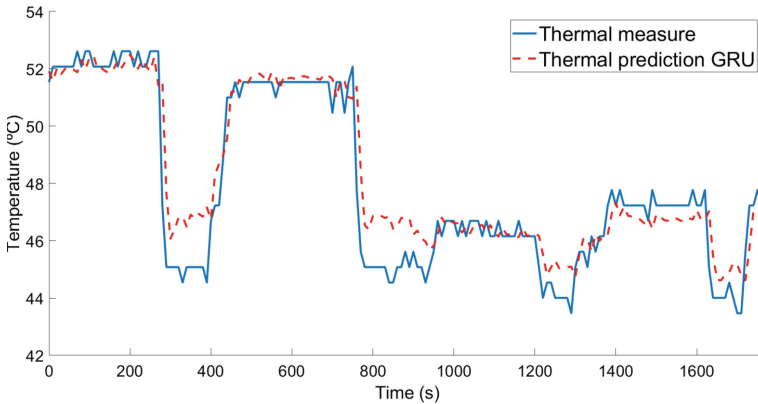
Our dataset (explained in Sect. 4) has been split into a training and a testing set. The training stage builds the thermal model according to the hyperparameter's settings. Then, the testing stage checks the model accuracy for a workload never seen by the modeling process, which consists of the 12% of our traces. Using the methodology proposed in Sect. 4 for the optimization of the hyperparameters, we obtain an optimized neural network for the GRU recurrent unit proposed in this research. Also, we apply the same methodology using other types of recurrent unit (Fully connected and LSTM) in order to compare the prediction accuracy. Table 1 presents the characteristics of each type of RNN and the prediction accuracy results in the testing dataset. For each recurrent unit type we provide results for the best solution found for 3, 2 and 1 layers respectively, when optimizing the neural structure. Additionally, we also present the best solution found when optimizing the hyperparameters involved in model training, following our modeling methodology. Our experiments are configured using the Nadam optimizer, the tanh activation function and the MAE loss function.

Based on the results it can be observed that our GRU model has an edge over LSTM and fully connected structures. It provides a $MAE \pm STD$ error of 0.75 ± 0.59 °C when evaluating the entire testing dataset, similar to the one provided by LSTM. Our GRU model also presents an RMSD of 0.957 °C and an R^2 value of 89.539% that defines how well the observed outcomes are replicated by

Table 1. Comparison of testing prediction accuracy

Recurrent unit	Neural structure	Batch size	Epochs	MAE \pm STD ($^{\circ}$ C)	RMSD ($^{\circ}$ C)	R^2 (%)
Fully Connected	(16, 32, 16)	65	80	1.11 ± 1.14	1.588	71.925
	(16, 8)	65	80	1.13 ± 1.08	1.562	72.824
	(256)	65	80	1.09 ± 0.83	1.371	79.071
	(256)	95	140	0.84 ± 0.84	1.187	83.897
LSTM	(8, 4, 2)	65	80	0.95 ± 1.13	1.475	75.782
	(1, 1)	65	80	1.02 ± 0.82	1.311	80.871
	(2)	65	80	0.91 ± 0.89	1.270	82.044
	(2)	65	210	0.71 ± 0.66	0.969	87.527
GRU	(4, 2, 1)	65	80	0.81 ± 0.83	1.157	82.211
	(4, 2)	65	80	0.85 ± 0.78	1.150	82.429
	(4)	65	80	0.74 ± 0.68	1.002	86.668
	(4)	95	120	0.75 ± 0.59	0.957	89.539

the prediction model, outperforming the results provided by the other recurrent units. Figure 3 shows the thermal fitting of our GRU model with a prediction window of 1 min and it can be seen that it offers a good fitting to the real measurement's curve on scenarios with workloads that vary significantly during runtime.

**Fig. 3.** Testing fitting for our RNN-based thermal predictive model.

6 Conclusions

In this research, we successfully provide a predictive model based on a recurrent artificial neural network architecture, with GRU type neurons. Our model

makes accurate thermal predictions in a real system based on immersion cooling, where the computing infrastructure is submerged in a tank full of HFE-based fluid running real data analytics applications. The proposed model predicts the temperature of the computing system one minute in advance, presenting an average error of 0.75°C , an RMSD of 0.957°C and an R^2 value of 89.539% when compared to the real measurements provided by our monitoring system. By using this model, predictions can be made well in advance to take proactive decisions, both direct and indirect, on the system's temperature. This enables the development of novel proactive optimization strategies that provide a cooling setpoint temperature within a specific range, thus ensuring the maximum heat transfer capacity of the HFE.

Acknowledgment. This project has been partially supported by 3M, the Centre for the Development of Industrial Technology (CDTI) under contract IDI-20171194, and the Spanish Ministry of Economy and Competitiveness, under contracts TIN-2015-65277-R and AYA2015-65973-C3-3-R.

References

1. Breen, T.J., Walsh, E.J., Punch, J., Shah, A.J., Bash, C.E.: From chip to cooling tower data center modeling: part I influence of server inlet temperature and temperature rise across cabinet. In: 2010 12th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems, pp. 1–10 (June 2010)
2. Che, Z., Purushotham, S., Cho, K., Sontag, D., Liu, Y.: Recurrent neural networks for multivariate time series with missing values. CoRR abs/1606.01865 (2016)
3. Connor, J.T., Martin, R.D., Atlas, L.E.: Recurrent neural networks and robust time series prediction. *IEEE Trans. Neural Netw.* **5**(2), 240–254 (1994)
4. Howell, J.: Number of connected IoT devices will surge to 125 billion by 2030. IHS Markit Press Release (2017)
5. Kermanshahi, B.: Recurrent neural network for forecasting next 10 years loads of nine Japanese utilities. *Neurocomputing* **23**(1), 125–133 (1998)
6. Moore, J., Chase, J., Ranganathan, P., Sharma, R.: Making scheduling “cool”: temperature-aware workload placement in data centers. In: Proceedings of the Annual Conference on USENIX Annual Technical Conference, ATEC 2005, p. 5. USENIX Association, Berkeley (2005)
7. Tang, Q., Gupta, S.K.S., Varsamopoulos, G.: Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: a cyber-physical approach. *IEEE Trans. Parallel Distrib. Syst.* **19**(11), 1458–1472 (2008)
8. Xu, H., Feng, C., Li, B.: Temperature aware workload management in geo-distributed data centers. *IEEE Trans. Parallel Distrib. Syst.* **26**(6), 1743–1753 (2015)
9. Zaytar, M.A., Amrani, C.E.: Sequence to sequence weather forecasting with long short-term memory recurrent neural networks. *Int. J. Comput. Appl.* **143**(11), 7–11 (2016)



Detecting Intrusive Malware with a Hybrid Generative Deep Learning Model

Jin-Young Kim and Sung-Bae Cho^(✉)

Department of Computer Science, Yonsei University, Seoul, South Korea
{seago0828, sbcho}@yonsei.ac.kr

Abstract. A small amount of unknown malware can be analyzed manually, but it is generated with extremely more and more so that automatic detection of them is needed. Malware is usually generated with different features from those of existing ones (e.g., code exchange, null value insertion, or reorganization of subroutines) to avoid detection of antivirus systems. To detect malware with obfuscation, this paper proposes a method called latent semantic controlling generative adversarial networks (LSC-GAN) that learns to generate malware data with *i*-feature from a specific Gaussian distribution which represents *i*-feature and distinguish it from the real. Variational autoencoder (VAE) projects data to latent space for feature extraction and is transferred to generator (G) of LSC-GAN to train it stably. G generates data from Gaussian distribution, so it produces similar data but not identical to the actual data: it includes modified features compared with the real. The detector is inherited with transfer learning in an encoder that learns various malware features using real and modified data generated by the LSC-GAN based on a LSC-VAE. We show that LSC-GAN achieves detection accuracy of 96.97% on average that is higher than those of other conventional models. We demonstrate statistical significance of the performance of the proposed model using t-test. The result of detection is analyzed with confusion matrix and F1-score.

Keywords: Malicious software · Malware detection
Generative adversarial networks · Variational autoencoder · Transfer learning

1 Introduction

Malicious software (malware), which is a generic term for all software products that adversely affect computers, is significant tool used in cyberwar. It is generally used to steal personal, financial, or business information as well as military information. It has been steadily growing in speed (rapidity of threats), number (growing threat landscape), and discrepancy (introduction of new methods) [1]. They usually intrude into a computer and destroy or steal important information. Once infected, it infects additional connected computers in the vicinity and increases the damage. Malware is generally divided into 4 types in terms of its behavior: Virus, worm, PUP, and trojan. When virus is executed, it manipulates malicious behavior that it is doing to other files, such as by inserting code in another medium. Unlike a virus, a worm replicates itself without an intermediary, and the infection proceeds through a vulnerability of the

operating system to another network. If the worm is executed, it will copy itself constantly in the PC, which will adversely affect the memory and the CPU. Potentially unwanted program (PUP) is installed with consent that is not done voluntarily. It may use an implementation that can compromise privacy or weaken the computer's security. Trojan misleads the user's original intention by performing a hidden function if this is executed after the attacker has tricked the target for the intended purpose.

There are two main ways to detect malware: one is static code analysis and the other is dynamic code analysis. Former refers to analyze software without actual execution, but latter does with it. Static code analysis detects malware based on raw code, so there is no need for additional processors for malware execution and no risk of direct execution, but it is vulnerable to malware deformability. Dynamic code analysis is robust to malware variability because it detects malware based on actual execution, but there are intense time complexity, large resource consumption, and poor scalability. To work out both limitations, we propose a deep learning method to detect malware based on raw code and to generate malware with arbitrary modified features so that detector can learn features of it even with unseen obfuscations. Before generating the data directly, it is important to know the features of them. Because VAE learns to project data with encoder to a specific latent space and reconstructs them with decoder, it can represent the characteristics of data. We exploit the decoder and encoder to generate the competitive new data for GAN and to confirm that G generates appropriate data, respectively.

2 Related Works

Many studies have been conducted to deal with malware because the damage incurred by it has been increasing and the need for malware detection methods is evident. We review the approaches for malware detection in two categories: static code analysis and dynamic code analysis.

In the approach of static code analysis, Nataraj et al. preprocessed and classified malware data written in the binary code into images [2]. However, if the null values inserted in malware changes, this method cannot detect malware well. Grace et al. tried detecting Android malware using first-order and second-order analyses [3]. They used ensemble technique which mixes two modules that detect malware in different ways. Garcia et al. extracted features of malware using image transformation technique proposed by Nataraj et al. and detected the malware using random forest [4]. Wang et al. proposed a method to detect malware using an adversary resistant deep learning model with random feature nullification [5]. They nullified characteristics randomly, resulting in the risk of eliminating meaningful features as well as null values.

There are also several studies that detect malware based on dynamic code analysis. This could recognize malware behavior, but it is limited to detect malware which matches with stored templates. Ye et al. used Windows Audit Log to detect malware [6], but they have disadvantage that the malware which has only pre-defined features can be detected; thus, they cannot capture modified features. Lin et al. proposed a virtual time control (VCT) mechanics to detect malware in a short time [7].

These studies proposed methods that extract features and detect malware based on them, or execute it directly and capture it with logs, but they did not deal with obfuscation; thus, detection of modified malware is not guaranteed. To detect them, we propose a method that exploits deep learning with GAN based on VAE. The proposed model extracts appropriate features with VAE, generates virtual malware for expanding the range of malware with GAN, and finally detects malware with the generated and the real data by transferring encoder to detector.

3 The Proposed Method

3.1 Overview

Whole process of the proposed method is illustrated in Fig. 1, which is divided into three parts: (1) feature extraction, (2) data generation, and (3) detection.

Before using binary codes of malware in the proposed method, they are pre-processed, and used as the input to the first part where the data is projected to a specific latent space according to its features and reconstructed. We use LSC-VAE in the first part, and reuse encoder of LSC-VAE and decoder of LSC-VAE as G in the second part. G generates fake malware data from a specific latent space with D to learn the characteristics of malware data while encoder projects it back to specific latent space. Finally, the encoder is transferred to the detector, and it is trained to detect malware data.

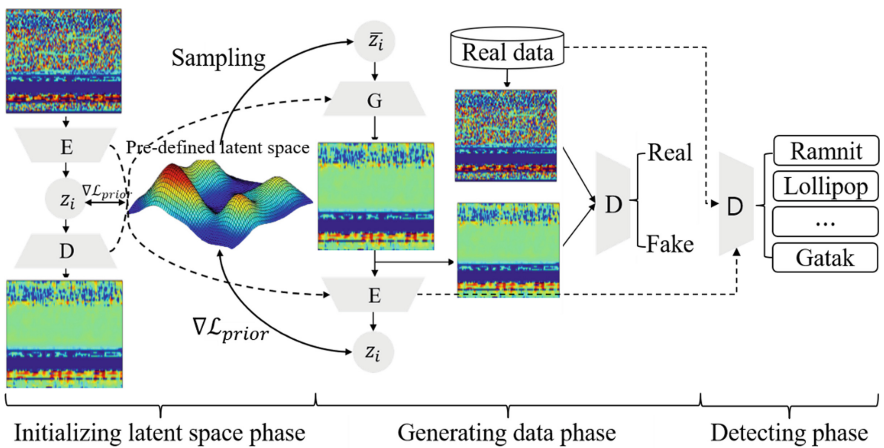


Fig. 1. The architecture of the proposed method. The process under the dashed line is the detection phase and all process is conducted separately.

3.2 Feature Extraction with VAE

Autoencoder has traditionally been used to learn representation of data without supervision. VAE, one of autoencoders, is one of the most popular approaches to unsupervised learning of complicated distributions [8]. The basic process of autoencoder is as follows:

$$z = \text{Enc}(x) \sim Q(z | x), \quad (1)$$

$$x = \text{Dec}(z) \sim P(x | z), \quad (2)$$

where x is a data and z is a latent variable. We project x to z with an encoder Q , and reconstruct z back to x with a decoder P . It can learn a representation of data, but it is hidden to us. To show which features are projected and reconstructed, we project and reconstruct the data x_i with i -feature into a specific space $Q(z_i | x_i)$. The changed autoencoder process is as follows.

$$z_i = \text{Enc}(x_i) \sim Q(z_i | x_i), \quad (3)$$

$$x_i = \text{Dec}(z_i) \sim P(x_i | z_i), \quad (4)$$

where index i means a feature which is included in data x or latent variable z . The encoder is regularized by imposing a prior over the latent distribution $P(z)$. In general, $z \sim N(0, I)$ is chosen, but we choose $z_i \sim \mathcal{N}(\mu_i, I)$ for dealing with a specific feature, where μ_i is a prototype vector of data with i -feature. We call VAE that goes through the process of Eqs. (3) and (4) as LSC-VAE. The loss of LSC-VAE is sum of loss of VAE and prior regularity as in Eq. (5).

$$\mathcal{L}_{\text{LSC-VAE}} = \mathbb{E}_{z_i \sim Q(z_i|x_i)} [\log P(x_i | z_i)] + \mathcal{D}_{\text{KL}}[Q(z_i | x_i) || P(z_i)] = \mathcal{L}_{\text{VAE}} + \mathcal{L}_{\text{prior}}, \quad (5)$$

where \mathcal{D}_{KL} is the Kullback-Leibler divergence. LSC-VAE is also used in training GAN, which is discussed more in details in the next section. We use deconvolution layers to increase the size of latent variables to that of malware data [9]. We put batch normalization layer [10], leaky ReLU activation layer and dropout layer after every layer in above except the last layer.

3.3 Generating Data Using LSC-GAN

GAN has led to significant improvements in data generation [11]. The basic training process of GAN is to adversely interact and simultaneously train G and D. Equation (6) shows the objective function of a GAN. p_d is the probability distribution of the real data. $G(z)$ is generated from a probability distribution p_z by the G .

$$\min_G \max_D V(D, G) = \min_G \max_D E_{x \sim p_d(x)} [\log D(x)] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad (6)$$

Since original GAN has disadvantage that the generated data are insensible because of the unstable learning process of the generator, we pre-train G with decoder of VAE. The result of VAE is that $|p_d^{LSC-VAE} - p_G^{LSC-VAE}| \leq |p_d - p_G|$ which yields a goal of GAN ($p_d \approx p_G$) stably.

From the game theory point of view, the GAN converges to the optimal point when the discriminator and the generator reach the Nash equilibrium. In this section, let p_G be the probability distribution of data created from the generator. We show that if $G(z) \approx x$, i.e., $p_d \approx p_G$, the GAN reaches the Nash equilibrium. We define $J(D, G)$ and $K(D, G)$ as in [12]. Then, we can define the Nash equilibrium of the GAN as a state that satisfies Eqs. (7) and (8). Fully trained generator and discriminator are denoted by G^* and D^* , respectively.

$$J(D^*, G^*) \leq J(D^*, G) \quad \forall G \quad (7)$$

$$K(D^*, G^*) \leq K(D, G^*) \quad \forall D \quad (8)$$

Theorem 1. If $p_d \approx p_G$ almost everywhere, then the Nash equilibrium of the GAN is reached.¹

GAN used in this paper is based on Wasserstein GAN. It defines the distance between distributions as Wasserstein, not Jensen-Shannon used in original GAN [13]. For verifying that the data generated from the space representing the i -feature really has the i -characteristic, we project it back to latent space with encoder Q . Objective function of LSC-GAN is in Eq. (9).

$$\max V_{LSC-GAN}(D, G) = \max V(D, G) - \mathcal{D}_{KL}[Q(z_i | G(z_i)) || \mathcal{N}(\mu_i, I)] \quad (9)$$

Reason for using GAN is to classify new data with surrounding noise with model learned from existing data. Because GAN generates data from a random distribution, new data has some variants compared to existing data. It expands a knowledge space of data. Encoder and D are trained with expanded space. Therefore, encoder and D are robust to deformation [14].

4 Experiments

4.1 Dataset and Experimental Setting

To validate the performance of generating fake data and detecting malware data, we used the malware dataset from the Kaggle Microsoft Malware Classification Challenge². Ramnit, Kelihos ver 3 (K3), Simda, and Kelihos ver1 (K1) are a botnet which can be used to perform distributed denial-of-service attack (DDos attack), steal data,

¹ The proof of Theorem 1 was discussed by Kim et al. [14].

² <https://www.kaggle.com/c/malware-classification>.

and allows the attacker to access the device and its connection. Vundo, Simda, Tracur, and Gatak are a trojan horse malware which is misleads user’s true intentions. It appears to be a normal software, but it will run malicious code if run. Lollipop is an adware that generates revenue for its developer by automatically generating online advertisements in the user interface of the software. Obfuscator, ACY (O.ACY) is a combination of several malicious methods.

The data are given in form of assembly and binary code, and we only used the binary code. The values in hexadecimal are converted to decimal numbers from 0 to 255, which are normalized into values of 0 to 1. As Nataraj did, we convert the malware code to image, called *malware image*.³ Then, because malware images were too large, the images were reduced to 0.2 times their original sizes, resulting in the size of 256×128 . The number of malware for each type is shown in Table 1. We set the size of latent space as 270 dimension and assign 90 dimension for each malware class.

Table 1. Summary of malware data.

Type	Train (test)	Type	Train (test)	Type	Train (test)
Ramnit	1387 (153)	Vundo	435 (40)	K1	358 (40)
Lollipop	2249 (229)	Simda	35 (7)	O.ACY	1110 (118)
K3	2620 (322)	Tracur	688 (63)	Gatak	898 (115)

4.2 Result of Detection

In this section, the performance of the proposed model is compared with other conventional machine learning algorithms, a convolutional neural network (CNN) (which has the same architecture to that of detector), and a GAN (which is not based on VAE). When going from the generation phase to the detection phase, D of GAN is transferred because there is no encoder to be transferred to the detector. Architectures of CNN and D of GAN are same to detector of our proposed model. Parameters of other machine learning methods is set to default value in scikit-learn library.

The results of these experiments are summarized in Fig. 2 and Table 2. The averaged accuracy is 96.97% which is better than other conventional method. Some studies of malware detection used the same dataset that we used here [15–17]. However, there is no significant difference in the malware detection ability after reducing the scale of the malware data. We also verify the statistical meaning of difference among CNN, GAN, and the proposed model with *t*-test, resulting in meaningful difference.

³ We use the ‘jet’ colormap to represent the values between 0 and 1 in color.

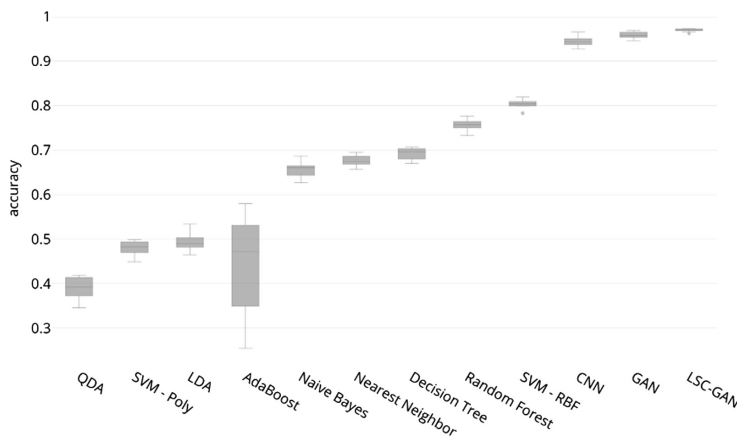


Fig. 2. The results of experiments in box plot.

Table 2. The numerical results of experiments.

	CNN	GAN	LSC-GAN
Accuracy	94.49	95.79	96.97
Std. dev	1.34e-04	5.78e-05	1.22e-05
p-value	4.17e-06	3.11e-04	-

4.3 Analysis of Result

Generated malware images are illustrated in Fig. 3 with the real images. They look very similar to the actual image, but we cannot return the image to the code because the image was reduced in the preprocessing step. Data similar to the generated data is found in the training dataset using structural similarity index (SSIM). The SSIM value and standard deviation of malware data containing real and generated data are 0.1019 and 0.3308, respectively, but those of only real data are 0.1068 and 0.3224, respectively. It shows that the generated data increase the diversity of malware.

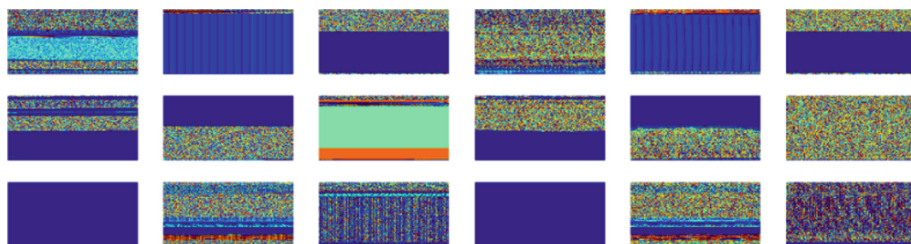


Fig. 3. The actual malware images (left 9) and the generated malware images (right 9). The malware image to the left of the three spaces of the generated malware image is the actual image with the largest SSIM value.

We calculate precision, recall, and F1 scores for each malware type and lists them in Table 3. Most of malware data have high F1-score except for Simda, whose type of malware is scarce.

Table 3. Precision, recall, and F1 scores for different malware types.

	R	L	K3	V	S	T	K1	O	G
Precision	0.987	0.987	1.000	0.886	0.500	0.836	1.000	0.966	0.974
Recall	0.955	0.978	0.997	0.975	0.286	0.968	0.950	0.949	0.983
F1-score	0.970	0.982	0.998	0.929	0.364	0.897	0.974	0.957	0.978

5 Conclusions

In this paper, we raise the problems caused by malware and attempt to solve them. The newly generated malware has some obfuscations compared with existing malware, e.g., code exchange, null value insertion, and reorganization of subroutine. To deal with these modification and detect modified malware, we propose a LSC-GAN which extracts features with VAE and generates virtual data to expand a range of knowledge space. The proposed detector inherits the ability of the encoder, so it can know a wider knowledge space. It achieves 96.97% of accuracy which is a better performance than other conventional machine learning algorithms.

In the future, we will address the issue of different lengths of various malware, since the malware code was converted into malware images through crop and operations in this study. After this issue, we will convert generated malware data to malware code. We plan to build a complete system for detecting malware and use it for the real world.

Acknowledgment. This work was supported by Air Force Defense Research Sciences Program funded by Air Force Office of Scientific Research.

References

1. Dhammi, A., Singh, M.: Behavior analysis of malware using machine learning. In: IEEE International Conference on Contemporary Computing, pp. 481–486 (2015)
2. Nataraj, L., Karthikeyanm, S., Jacob, G., Manjunath, B.S.: Malware images: visualization and automatic classification. In: Conference on Visualizing for Cyber Security, pp. 1–7 (2011)
3. Grace, M., Zhou, Y., Zhang, Q., Zou, S., Jiang, X.: Riskranker: scalable and accurate zero-day android malware detection. In: Proceedings of International Conference on Mobile Systems, Applications, and Services, pp. 281–294 (2012)
4. Garcia, F.C.C., Muga, I.I., Felix, P.: Random Forest for Malware Classification. arXiv preprint [arXiv:1609.07770](https://arxiv.org/abs/1609.07770) (2016)

5. Wang, Q., et al.: Adversary resistant deep neural networks with an application to malware detection. In: International Conference on Knowledge Discovery and Data Mining, pp. 1145–1153 (2017)
6. Ye, Y., Chen, L., Hou, S., Hardy, W., Li, X.: DeepAM: a heterogeneous deep learning framework for intelligent malware detection. *Knowl. Inf. Syst.* **54**, 1–21 (2017)
7. Lin, C.H., Pao, H.K., Liao, J.W.: Efficient dynamic malware analysis using virtual time control mechanics. *Comput. Secur.* **73**, 359–373 (2018)
8. Kingma, D.P., Welling, M.: Auto-encoding Variational Bayes. arXiv preprint [arXiv:1312.6114](https://arxiv.org/abs/1312.6114) (2013)
9. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 818–833. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10590-1_53
10. Ioffe, S., Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv preprint [arXiv:1502.03167](https://arxiv.org/abs/1502.03167) (2015)
11. Goodfellow, I. et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
12. Kim, J.Y., Bu, S.J., Cho, S.B.: Zero-day malware detection using transferred generative adversarial networks based on deep autoencoders. *Inf. Sci.* **460–461**, 83–102 (2018)
13. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: International Conference on Machine Learning, pp. 214–223 (2017)
14. Radford, A., Metz L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint [arXiv:1511.06434](https://arxiv.org/abs/1511.06434) (2015)
15. Kim, J.Y., Bu, S.J., Cho, S.B.: Malware detection using deep transferred generative adversarial networks. In: Liu, D., Xie, S., Li, Y., Zhao, D., El-Alfy, E.S. (eds.) Neural Information Processing. ICONIP 2017. Lecture Notes in Computer Science, vol. 10634, pp. 556–564. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70087-8_58
16. Drew, J., Moore, T., Hahsler, M.: Polymorphic malware detection using sequence classification methods. In: Security and Privacy Workshops, pp. 81–87 (2016)
17. Narayanan, B.N., Djaneye-Boundjou, O., Kebede, T.M.: Performance analysis of machine learning and pattern recognition algorithms for malware classification. In: Aerospace and Electronics Conference on and Ohio Innovation Summit, pp. 338–342 (2016)



Inferring Temporal Structure from Predictability in Bumblebee Learning Flight

Stefan Meyer¹(✉), Olivier J. N. Bertrand¹, Martin Egelhaaf¹,
and Barbara Hammer²

¹ Department of Neurobiology, University of Bielefeld, Universitätsstrasse 25,
33615 Bielefeld, Germany

{s.meyer,olivier.bertrand,martin.egelhaaf}@uni-bielefeld.de

² Department of Machine Learning, University of Bielefeld, Inspiration 1,
33619 Bielefeld, Germany

bhammer@techfak.uni-bielefeld.de

<http://web.biologie.uni-bielefeld.de/neurobiology/>,

<https://www.cit-ec.de/en/tcs>

Abstract. Insects are succeeding in remarkable navigational tasks. Bumblebees, for example, are capable of learning their nest location with sophisticated flight manoeuvres, forming a so-called learning flight. The learning flights - thought to be partially pre-programmed - enable the bumblebee to memorise spatial relations between its inconspicuous nest entrance and environmental cues. To date, environmental features (e.g. object positions on the eyes) and learning experience of the insect were used to describe the flights, but its structure, thought to facilitate learning, has not been investigated systematically. In this work, we present a novel approach, to examine whether and in which time span flight behaviour is predictable based on intrinsic properties only rather than external sensory information. We study the temporal composition of learning flights by estimating the smoothness of the underlying process. We then use echo state networks (ESN) and linear models (ARIMA) to predict the bumblebee trajectory from its past motion and identify different time-scales in learning flights using their prediction-power. We found that direct visual information is not necessary within a 200ms time-window to explain the bumblebee behaviour during its learning flight.

Keywords: Time series prediction · Modelling
Echo state network · Learning flight · Bumblebee

1 Introduction

One of the goals of neurobiology is to understand how behaviours of animals are generated and shaped. Insects, equipped with a small brain compared to human ones, are capable of showing sophisticated navigational behaviour.

© Springer Nature Switzerland AG 2018

H. Yin et al. (Eds.): IDEAL 2018, LNCS 11314, pp. 508–519, 2018.

https://doi.org/10.1007/978-3-030-03493-1_53

For example, bees [1], wasps [2] and bumblebees [3], when leaving their nest for the first time, engage in a series of flight manoeuvres necessary to learn the surroundings of their nest, to be able to find their nest hole again after a foraging trip. The nest entrance is only slightly larger than the size of the bee itself, often located on highly structured ground camouflaging it [4]. Finding the nest hole of a bumblebee hive is, therefore, a challenging task even for humans. Bumblebees, however, manage to return to their hive despite a visual resolution much lower than a human eye or a digital camera. One reliable way to find the nest hole is by using spatial relations between clearly visible environmental features, such as trees or the nest entrance [5]. However, bumblebees born in the nest do not know the spatial relationships and thus engage in a learning flight on their first flight out of their colony. Since these bees do not have any prior visual experience, the overall spatiotemporal structure of the learning flight is likely to be, at least partially, pre-programmed. However, the shape and duration of the learning flight decrease with experience and thus the pre-programmed behaviour is adaptable [3].

Assuming, at least partially pre-programmed behaviour that is temporally structured to facilitate the learning of spatial relations between the nest and environmental features, one would expect correlations between past and future behaviour within a certain time window. However, bumblebees are visually guided animals, and, thus, visual information is expected to influence the temporal structure of their flights. Therefore, future behaviours may strongly be dictated by visual information after a specific time-delay, and, thus, correlations between past and future behaviours are more complex and less obvious to determine.

Characterising learning flights within the scope of pre-programmed and sensory-driven behaviour is crucial in understanding how flying insects perceive and learn about their environment, but also have potential technical applications. For example, during the last decades, virtual reality environments have been designed and used to study the closed-loop mechanics of animal behaviours [6]. In VR, there is, however, often a lag between the measurement of the animal position and the update of visual information displayed. These lags can have dramatic effects on the animal experience. Being able to predict the movement of the animal independently of the sensory information can thus be used to reduce the lag in VR environments, and thus provide lag-free experience to the animal.

Here, we present, a first attempt in investigating the temporal composition of learning flights with the help of predictive models. More precisely, we hypothesise that flight behaviour which can be modelled based on the intrinsic temporal dynamics only corresponds to preprogrammed behaviour, while flight segments which cannot be modelled based on the intrinsic temporal dynamics require additional information such as visual input. We thus purposefully excluded visual input from our investigation to focus on the preprogrammed part of behaviour. This way, we transform the biological problem of understanding the driving incentives of bumblebees' flights into a time series prediction problem. For the latter, we first applied one predictive model to estimate the smoothness of the

function connecting future with past behaviour on different time-scales, this way providing a baseline for the prediction accuracy based on the intrinsic geometry properties alone. Then, we applied two classical time-series-prediction models (linear ARIMA; and non-linear Echo State Network, which constitutes a universal approximator for time-series), capable of capturing -at least in theory- dependencies between past and future values on a range of time-scales. Each model was used to predict future bumblebee behaviour from its past behaviour. The models predicted the behaviour with different accuracy. The accuracy was used to determine the predictive power of each model. Comparing the prediction-power and comparing the assumptions made in the design of models with our smoothness estimate, we have been able to characterise the time-scale at which the learning flight is determined strongly by pre-programmed behaviour to be 200 [ms].

2 Background

When exiting their nest for the first time, bumblebees perform learning flights, which consist of convoluted manoeuvres in order to learn the surroundings of their nest entrance, often hidden between grass blades and therefore difficult to find. At the beginning of the learning flight, the bumblebees tend to stay in the vicinity of the nest entrance and look towards it. They perform loop-like movements first in the region of the nest, and then increase and decrease the distance to the nest while looking at it for most of the time [7]. Since they perform this behaviour when leaving the nest hole for the first time and, thus, do not have any prior visual experience, the flight's overall spatiotemporal structure is likely to be, at least partially, pre-programmed. Surprisingly, individuals of the same population show a considerable degree of variability regarding the shape of the flight, even if they are confronted with the same environmental conditions [3].

In the past, efforts have been made to unravel how the behaviour of flying insects is structured. Braun et al. [8,9] were able to decompose behaviour of blowflies and honeybees into smaller blocks, called movement primitives, by applying a k-Means cluster algorithm. The movements primitives found in the two species were similar, as well as their occurrences during the flights, suggesting similarities in the behavioural structure amongst different flying insects. More specifically, nine prototypical movement blocks could be identified: two of these reflect fast rotational movement (called saccades), and the rest translational movements of different velocities (e.g. lift and forward flight). By decomposing the learning flights of honeybees into movement primitives, Braun et al. suggested specific primitives to be more likely to appear at particular locations within the environment than others, hence, the research suggests that there do exist regimes where likely visual input is relevant. Even though these findings represent significant milestones in the understanding of insect flight structure, they - in contrast to our work- do not take the temporal structure of behaviour into account. However, behaviour is a function of time, hence it is an interesting endeavour to complement cluster-based data analysis by methods which focus

on the temporal dynamics. Our specific goal is to investigate whether such technology can highlight parts of bumblebee flights which can be predicted based on the internal temporal dynamics only, hence suggesting regimes of flights where visual input does not play a role.

Our hypothesis is, that there do exist time-scales, which enable a prediction of the temporal behaviour based on the previous time steps only (excluding vision), hence suggesting that these parts correspond to preprogrammed behaviour.

3 Modelling and Experimental Evaluation

3.1 Experimental Data of Flight Behaviour

The dataset of Lobecke et al. [3] consists of learning flights of bumblebees captured at 148 fps in an octagonal arena of about 70 cm in incircle diameter (see Lobecke et al. [3] for more details). The high sampling rate of data leads to small variation in the bumblebee position between frames that can be easily predicted by a simple linear model. Since we are not interested in trivial relationships present, between past and future motion, at very short time-scales, we downsampled the trajectories by a factor of three.

The learning flights of bumblebees varied in length, between individuals. Moreover, certain flights contained missing points along trajectories due to occlusion by objects and had to be cut into continuous blocks. However, this led to very short trajectories unrepresentative of the structure of the learning flights. We, therefore, kept flights of at least 2000 frames after downsampling, equivalent to $\simeq 40.5$ s. In our analysis, 48 trajectories were used. We denote the resulting set of trajectories as Y . This consists of time series (y_1, y_2, \dots, y_T) of different length $t \in \mathbb{N}$ where entries are three dimensional coordinates with their respective rate of change $y_i \in \mathbb{R}^6$.

3.2 Learning Task

In this paper, we considered the predictability of future spatial features from a series of past spatial features. A given training set can thus be denoted by:

$$\tau = \{(Y_{t,m} := [y_{t-m}, \dots, y_{t-1}, y_t], y_{t+k}) \mid y \in Y, t \leq T - k \text{ where } T = \text{len}(y)\} \tag{1}$$

where $m \in \mathbb{N}$ is a meta-parameter of the prediction model and optimized on the data set. The prediction horizon $k \in \mathbb{N}$ is varied in subsequent experiments, to test which prediction horizon enables a good prognosis, and which doesn't - the latter serving as an indication that additional information such as visual input is required for a prediction.

For our approach, we assume that there is a relation between future and past position of the animal. However, the nature of this relation is characterized by the time passing between one sequence of observation and its predecessors. Formally, we try to model:

$$y_{t+k} = f_k(Y_{t,m}) + \epsilon \tag{2}$$

where $f_k(Y_t, m)$ is the relation of observations for time horizon k with unknown f_k , and ϵ denotes a noise term. The idea is, that the relation f for a given k is a function, which reflects the dependency between future and past spatial measurements.

3.3 Motivation

For spatial time series data, the time series alone usually displays a smoothness. High local smoothness therefor represents trivial, almost linear-relationships, as it might be present during inertia, while low local smoothness indicate more complex relationships. For some $k > k^*$ we might observe very low smoothness in f , which would render prediction impossible. This could be a result of the fact, that future behaviour might dependent on sensory input rather then previous behaviour at a given time-scale.

3.4 Smoothness Approximation

Since f is a function of unknown type, we can not investigate its local smoothness directly. However, we are able to approximate it with an Euler-based method, which approximates the prediction with a local linear model. For two subsequent points y_t and y_{t-1} we first predict the next observation y_{t+1} by adding the observed vector of change Δy_{t-1} with respect to the time that has passed Δt . By multiplying this rate of change with the prediction size h , we obtain an estimate for y_{t+h} . The full equation is given by:

$$\hat{y}_{t+h} = y_t + h \cdot \frac{\Delta y_{t-1}}{\Delta t} \quad (3)$$

Next, we calculate the error between the estimate \hat{y}_{t+h} and the real observation \hat{y}_{t+h} . Low errors indicate high local smoothness, since the rate of change stayed constant. We associate this with short-term behaviour of the animal. High errors, on the other side, indicate low local smoothness, thus reflecting more complex behaviour.

3.5 Prediction Models

Local smoothness provides a baseline about the predictability of the time series due to its geometric embedding only. Now we test for different time lags, in how far time series prediction models can surpass this baseline. If so, we judge this as an indicator that the temporal behavior is determined by the trajectory and thus the bee's behavior likely preprogrammed. We test two models: standard linear time series prediction with ARIMA, and nonlinear prediction with echo state networks as universal approximator.

Linear Prediction: In order to investigate this assumption, we applied a so-called autoregressive integrated moving average model (ARIMA) [10]. This model predicts the next value by linearly adding a number of previous weighted observations together with a number of weighted observed differences of observations. In this study, ARIMA models will be used to describe a dependency we call short-term behaviour. We expect it to occur at a larger time-scale than inertia.

An ARIMA(p,d,q) model with lag p , order of differentiation d and number of prediction errors q is fully described by:

$$(1 - \phi_1 L - \dots - \phi_p L^p)(1 - L)^d y_{t+1} = c + (1 + \theta_1 L + \dots + \theta_q L^q) e_{t+1} \quad (4)$$

where L is the lag-operator, d is the order of differencing y_{t+1} is the next observation, e_{t+1} is the next error, c is a trend and θ and ϕ are parameters to be fitted.

Low local smoothness indicates more complex behaviour, which might be reflected by non-linear relations. These relations might be captured by a complex model like the so-called Echo state network (ESN) [11]. The idea is, that complex temporal dependencies are caught in a high dimensional space and are therefore accessible for prediction. In particular, ESNs represent a class of models with high variance and as such have been shown to be universal function approximators.

Echo State Networks: An echo state network [11] is a particular form of a recurrent neural network (RNN). ESNs are highly interconnected RNNs, where neurons are so interconnected, that there is no distinguishable layer architecture left. All hidden neurons are existing together in a so-called reservoir. Connections between these neurons are initialised once and remain unchanged. The training happens between the reservoir and the output layer. We used ESNs to model non-linear dependencies within the trajectory, which might be interpreted as behaviour. Formally, the ESN is described by the following equations [11]. The reservoir state is described by:

$$x(t + 1) = f(Wx(t) + W^{in}y(t + 1) + W^{fb}y(t)) \quad (5)$$

Each component of the input vector $y(t + 1)$ is randomly connected to the reservoir via an input weight matrix W^{in} . The reservoir itself is given by N neurons, that are randomly connected to each other via the internal weight matrix W . The activation of the reservoir is denoted by the N -dim vector $x(t)$, where each component represents the activation of one neuron in the reservoir. In this work, the ESN is trained by using a method called ‘teacher forcing’ [12]. The network gets a starting point and has to predict the next observation. Then prediction and actual observation are compared and the mean squared error calculated. The following input for the network is the actual observation, and the process starts again. We used the following equation to calculate how the activation of the reservoir affects the output $y(t)$:

$$y(t) = f(W^{out} [y(t), x(t)]) \quad (6)$$

where $[y(t), x(t)]$ is a simple concatenation of the input and the reservoir activation. W^{out} is a $L \times (H + N)$ matrix, which can be regarded as the parameters of a linear regression, which are optimised by maximum likelihood estimation [MLE].

3.6 Error Function

The future behaviour of the bumblebee may be predicted with varying accuracy depending on the time-scale and the modelled dependency between subsequent data points. For training, the standard mean squared error is used as the cost function. Yet for interpretation more flexibility is needed. Predicting the exact position of the bee at a given time is in our study less relevant than predicting the overall shape of the trajectory because we are interested in similarity in dependencies of observations across individuals.

The dynamic time warping is considering more the shape of the trajectory than the exact temporal alignment. We, therefore, quantify the prediction-power of a given model by applying dynamic time warping (DTW) [13] on target trajectories and prediction. DTW consists of calculating the sum of Euclidean distances over the optimal wrapping path of all frames within the trajectories. Moreover, we divided the sum by the length of the trajectory to compare the DTW between different trajectory lengths. Therefore, smaller values represent lower errors.

3.7 Experiments

Each trajectory has been split into a training set and a test set, where the first part of the trajectory was used for training and the remainder for testing of the models. We denote the full trajectory length with T , T_{tr} the first time steps representing the training set, and T_{te} the remaining time steps representing the test set.

We always trained the models on the first part of the trajectory and tested on the remaining part, thus, assuming that the temporal structures between the two parts are similar. Testing our assumptions will require a larger dataset than the one of Lobecke et al. and will require long recordings to assess if later parts of the learning flight can be less predicted by our models than earlier parts. However, as we will see below, our model is still able to predict the bee behaviour in certain time windows, and thus certain temporal structures are conserved between the train and test part of the learning flight.

The training set has been created by shifting the actual observations against themselves, such that the training set can be described by

$$Y_{tr} = \{(Y_{t,m}, y_{t+k}) \mid \forall t \in T_{tr}; t + k \in T\} \quad (7)$$

Note that a part of the test set serves as target, since an overlap is unavoidable. Testing is then performed on the complementary set given by

$$Y_{te} = \{(Y_{t,m}, y_{t+k}) \mid \forall t \in T_{te}; t + k \in T\} \quad (8)$$

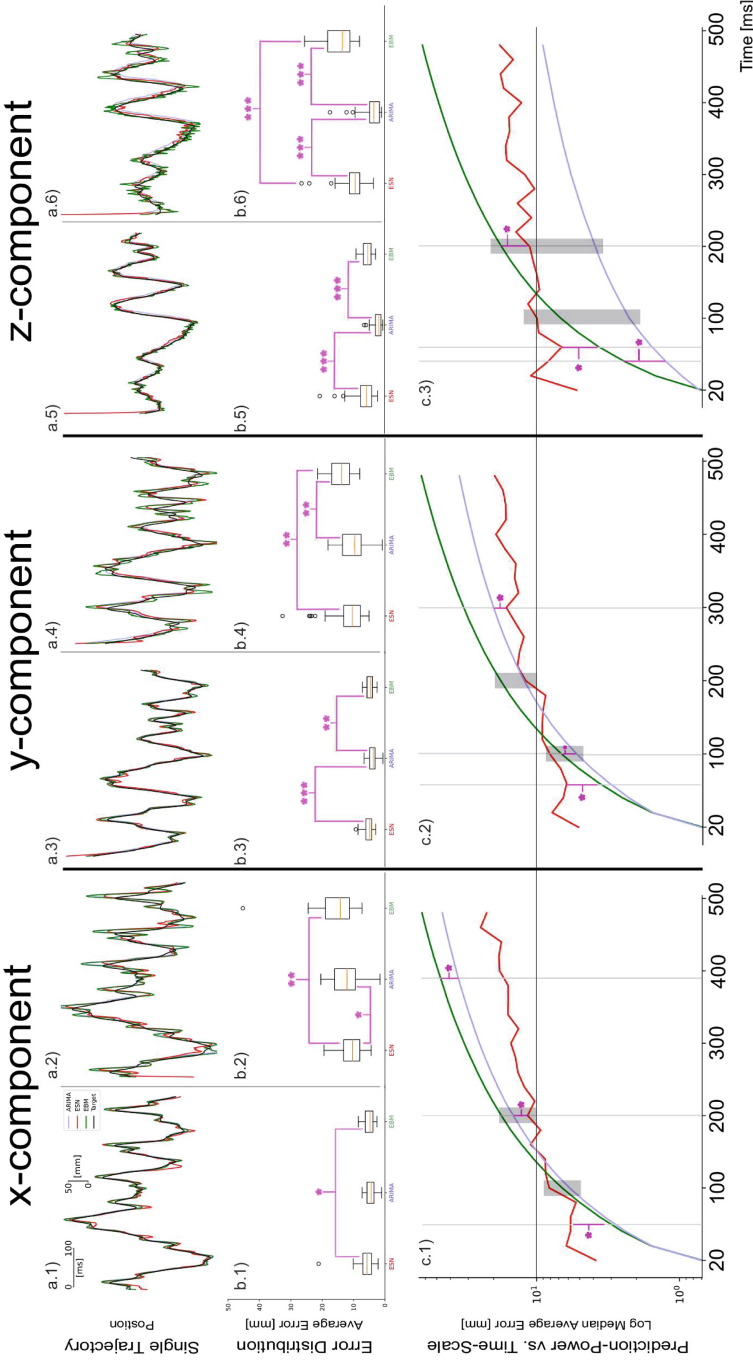


Fig. 1. Visualisation of the results. First row (a.x) shows the estimates made by each model together with the target trajectory for each spatial component in a single trajectory example. Odd numbers for .x show results obtained for 100 [ms] prediction, even numbers for 200 [ms]. Second row (b.x) shows the average DTW-error for each trajectory within the test set as a boxplot. Odd and even numbers for .x convey the same information as in row one. Third row (c.x) shows the median average error on a log scale for all models, with respect to the forecasting horizon. The purple bar indicates at which forecasting horizon performance differed significantly. Significance has been tested with a KS-test ($p < 0.05^*$, $p < e0.01^{**}$, $p < e0.001^{***}$) (Color figure online)

Note, that some values at the end of each trajectory are discarded because their observation is not available.

As for the actual training, two slightly different methods have been used to train the linear and non-linear model.

First, the initial parameters of the ARIMA model on the training set have been chosen by investigation of significant lags within an autocorrelation function (ACF) and partial autocorrelation function (PACF) with respect to all trials. Then, the model has been fine-tuned by evaluating a rolling forecasting origin.

We selected the candidate with the smallest error on the test set, which was an ARIMA(5,2,0) model. We fitted one instance per trial of this model on the training set and predicted on the test set, with a rolling forecast.

The pseudoinverse of the ESN has been optimised via teacher forcing [12] on the training set. Performance has then been evaluated on the test set for different hyperparameter selections. Due to the random nature of reservoir connection, for each hyperparameter configuration, we initialised 20 ESN instances and compared their mean error to find the best setting. The best performing model consisted of 50 neurons within the reservoir, each having a non-linear activation function (hyperbolic tangent). We selected the Ridge-Regression parameter to be 0.99 and the spectral radius of W to be 0.85.

We created one instance of an ESN which we trained on all trajectories. We then tested its performance on the test sets.

The final error we report is the test error of the best performing models. We investigated this error over three different tasks.

First, to increase our understanding of how estimates made by our models differed to the actual observation and among each other, we visualised results obtained for a single trajectory. We split the actual results into x , y , and z because each of these components is subject to different physical constraints. Gravitation restricts more changes in height for example (z -component) than x and y .

One example of predictions can be seen in the first row of Fig. 1. Results show the same trajectory and its predictions for two different time scales (100 [ms] and 200 [ms]). The trajectory and its predictions almost overlay indicating that our approach can be used to predict bumblebee behaviour relatively well. However, differences in prediction-power can be seen qualitatively.

Second, as mentioned above, the learning flights of bumblebees vary between individuals. The variation in the flights may have an impact on predictability. We, therefore, extended our analysis of a single trajectory to all trajectories within the dataset and calculated the DTW errors made for each trajectory. An example of this analysis for the time-scale of 100 [ms] and 200 [ms] can be seen as a boxplot of the errors in the second row of Fig. 1.

The ESN performs worse than ARIMA and EBM when predicting in the near future (100 [ms]). We observe growing magnitudes of error when the prediction-size is increased from 100 [ms] to 200 [ms] for all models (compare I and J; K and L; M and N). However, ESN and ARIMA are showing smaller error growth

than EBM, for all three components. ESN is performing better on x-component, while ARIMA models are showing lower errors for y and z-component.

Third, and most relevant for our research question understanding which processes shape behaviour observed during learning flights of bumblebees recorded in the context of navigation behaviour, we investigated errors made by our models for different prediction-sizes. This helps us in assessing the different regimes (inertia, physical constraints, pre-programmed behaviour) sufficient to explain bumblebee behaviour at different time-scales.

To tackle this task, we considered the median error made by each model, for each component and time-scale. Results can be seen in the first row of Fig. 1.

For small prediction-sizes, (up to 100 [ms]) ARIMA and EBM perform better than ESN, while we observe ARIMA models to outperform EBM in general. Beyond 100 [ms], a single model does not give the highest prediction-power on all components. For x, ESN is capable of outperforming ARIMA at 200 [ms], while for the y-component this occurs only when predicting more than 300 [ms] into the future. In regards to z-component, the ESN is not capable of performing better than the ARIMA at any time-scale.

Beyond 300 [ms] all models produce error magnitudes larger than the actual size of the bee and are therefore considered to fail. Beyond this time-scale, future observations cannot be explained by past observations with the help of our models. The failure to predict beyond 300 [ms] may be due to inadequate models or due to a lack of dependency in the data. It is worth noting, that ARIMA, as well as ESN, are still performing significantly (KS-test, $p < 0.01$) better than the smoothness estimator.

4 Concluding Remarks

We presented a novel method of investigating the temporal structure of Bumblebee learning flights with the goals of identifying at which time-scales inertia, physical constraints, and pre-programmed behavioural components shape the overall behaviour. We presented three different models, a Euler-based method (EBM) that represents inertia, an ARIMA model which describes the linear dependency of future behaviour from past behaviour (and can model physical constraints) and an echo state network (ESN) that captures possibly hidden dependencies such as pre-programmed behaviour. We assume, that, if a model can predict with an error lower than another model, the first model captures the dynamic of the bumblebee flight trajectory better than the second model. Therefore, the assumed dependency between observations behind the first model is more likely to be similar to the actual dependence of data points representing the bumblebee behaviour.

First, our findings in regards to the variance in prediction error on different trajectories are in line with the high variation observed in different flights reported by [3]. This variation can be seen in the second row of Fig. 1).

Second, by investigating prediction-power of our models on different time-scales, we have been able to characterise the dependency of future behaviour

from past behaviour. Our results suggest that inertia and simple linear relations are sufficient to explain the bumblebee behaviour within 100 [ms]. This time-scale is comparable to the duration during which prototypical movement blocks described by Braun et al. [8] and Boeddecker et al. [14] happened during flights. One might, thus, argue that movement primitives are the results of physical constraints, while a combination of movement primitives forms behaviour. Assuming that spatial values within one movement primitive are highly correlated, one would expect that this correlation vanishes when the primitive is over, that is, as the time-scope widens. It is therefore expected, that beyond 100 [ms], descriptions of behaviour provided by models that assume inertia or physical constraints would fail, which is what we observe, at least partly. However, this demands further investigation.

When considering predictability at a time-scope beyond 100 [ms], x and y coordinates differ from the z components. The ESN model predicted better the x and y components than the ARIMA or the EBM models, indicating a non-trivial relation between past and future observations. This relation may be due to dependencies between different movement primitives. This dependency would imply, that a certain block of behaviour has some dependence on previous movement primitive. This indicates that pre-programmed behaviour might be observable at a time-scale of 200 [ms].

Beyond 200 [ms] all our models fail, indicating either a more complex dependency or no dependency on future behaviour on previous behaviours - without additional information - at all. The latter would suggest, that an animal might use additional information provided by sensory input to shape behaviour on a larger time-scale. It is reasonable to think that additional information plays a role in future behaviour since experiments show, that, e.g. visual cues play a role in navigation, and learning flight might be shaped to maximise information gain [15].

Our study might be a basis for understanding the mechanisms underlying the learning flight of bumblebees. We have been able to uncover time-scale in which pre-programmed behaviour occurs. One limitation of our approach is, however, that our model and prediction are relying on the sole basis of past behaviour. Our models may not have been sophisticated enough to capture more complex correlations. Furthermore, bumblebees are visually guided animals, and thus visual information influences their behaviour. We will, therefore, in our future works take into account the scenery seen by bumblebees during their flight.

References

1. Winston, M.L.: The Biology of the Honey Bee. Harvard University Press, Cambridge (1991)
2. Collett, T., Lehrer, M.: Looking and learning: a spatial pattern in the orientation flight of the wasp *vespula vulgaris*. Proc. Roy. Soc. Lond. B. **252**, 129–134 (1993)
3. Lobecke, A., Kern, R., Egelhaaf, M.: Taking a goal-centred dynamic snapshot as a possibility for local homing in initially naïve bumblebees. J. Exp. Biol. **221**, jeb168674 (2018)

4. Goulson, D.: *Bumblebees: Behaviour, Ecology, and Conservation*. Oxford University Press on Demand, Oxford (2010)
5. Robert, T., Frasnelli, E., de Ibarra, N.H., Collett, T.S.: Variations on a theme: bumblebee learning flights from the nest and from flowers. *J. Exp. Biol.* **221**, Article no. 4 (2018)
6. Stowers, J.R., et al.: Virtual reality for freely moving animals. *Nat. Methods* **14**(10), 995 (2017)
7. Collett, T.S., Zeil, J.: Flights of learning. *Curr. Dir. Psychol. Sci.* **5**(5), 149–155 (1996)
8. Braun, E., Geurten, B., Egelhaaf, M.: Identifying prototypical components in behaviour using clustering algorithms. *PLOS ONE* **5**(2), 1–15 (2010)
9. Braun, E., Dittmar, L., Boeddeker, N., Egelhaaf, M.: Prototypical components of honeybee homing flight behavior depend on the visual appearance of objects surrounding the goal. *Front. Behav. Neurosci.* **6**, 1 (2012)
10. Box, G.E., Jenkins, G.M.: *Time Series Analysis: Forecasting and Control*, Revised edn. Holden-Day, San Francisco (1976)
11. Jaeger, H.: The echo state approach to analysing and training recurrent neural networks with an erratum note. GMD Report 148 (2010)
12. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: *Deep Learning*, vol. 1. MIT Press, Cambridge (2016)
13. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. Acoust. Speech Sig. Process.* **26**(1), 43–49 (1978)
14. Boeddeker, N., Mertes, M., Dittmar, L., Egelhaaf, M.: Bumblebee homing: the fine structure of head turning movements. *PloS ONE* **10**(9), e0135020 (2015)
15. Zeil, J., Kelber, A., Voss, R.: Structure and function of learning flights in ground-nesting bees and wasps. *J. Exp. Biol.* **199**(1), 245–252 (1996)



Intelligent Wristbands for the Automatic Detection of Emotional States for the Elderly

Jaime A. Rincon¹, Angelo Costa²(✉), Paulo Novais², Vicente Julian¹,
and Carlos Carrascosa¹

¹ D. Sistemas Informáticos y Computación, Universitat Politècnica de València,
Valencia, Spain

{jrincon,vinglada,carrasco}@dsic.upv.es

² Algoritmi Center/Department of Informatics, University of Minho, Braga, Portugal
{acosta,pjon}@di.uminho.pt

Abstract. Over the last few years, research on computational intelligence is being conducted to detect emotional states of people. This paper proposes the use of intelligent wristbands for the automatic detection of emotional states to develop an application which allows to monitor older people in order to improve their quality of life. The paper describes the hardware design and the cognitive module that allows the recognition of the emotional states. The proposed wristband also integrates a camera that improves the emotion detection.

Keywords: Wearable devices · Emotional models
Ambient assisted living

1 Introduction

With the start of mobile technology every day our devices are connected to the world-wide network (WWW). This connection allows us to share our photos, send messages, use the social network, etc. However, in recent years we can see how different devices can be connected to the network with the aim to control and to monitorize our environment. This new technology called Internet of Things (IoT), allows the creation of applications in different environments. This has been made possible by the emergence of several smaller and more powerful devices. These devices are increasing applications in domains such as smart homes, smart cities, healthcare and robotics.

The elderly people care is one of the most interesting applications, since globally, the elderly population is increasing, according to demographic projections [1]. According to the OMS, the amount of people aged over 60 is expected to double between 2000 and 2050 [2]. As less developed countries start to evolve, this trend is onset immediately [2]. A common societal issue that emerges from a rapid elderly population growth is the exponential demand or care (medical and otherwise).

These care demands can be met by using the new technology in IoT and embedded systems, these devices can be used as wearable devices. These devices can obtain information related to the person movement into environment and acquire some bio-signals such as electrocardiogram (ECG), heartbeat and skin resistance. These data can be used to know not only the good state of health, if not, can be used to know the emotional state. Some studies [3,4] associate the emotional change with health problems.

In this work we propose to detect the emotional state of older people in an Ambient Intelligence (AmI) application with the help of wearables. The application would be used by the caregivers to try to improve the activities to be done with the elderly and by the way, improving their quality of life. Thus, the main goal of the proposed system is to use the knowledge of the emotional state of older people to maintain a state as close as possible to happiness or to detect unwanted situations from an emotional point of view. For this purpose, an intelligent wristband has been designed in such a way that it integrates the necessary components to take biometric measurements to infer the person's emotional state.

The rest of the paper is structured as follows. Section 2 presents the related work. Section 3 describes the proposed system, which has been divided into the hardware description and the cognitive service. Section 4 briefly presents a case study. Finally, some conclusions are shown in Sect. 5.

2 Related Work

Other efforts that are in the same or related domains (emotional detection through body signals) of our project are presented in this subsection. They represent the most advanced methods and technologies, being reference points of best, and sometimes, worst practices. These research projects provide cues to what will be the next developments in the domain.

In terms of new approaches to the detection of emotions through body sensing, [5] present a novel approach to detecting emotions through ECG. Their initial results show over 90% accuracy in detecting emotions in a controlled environment. The feature extraction (in this case the emotion) goes through an unorthodox process of data processing, they have altered the classical signal processing and data classification structure. The authors have first implemented a quantization method that compares the incoming signal to a dataset, meta-classifying them. The authors justify this approach by assuming that early data processing stage can constrain the data. Then, they compress the ECG meta-data using an ECG dataset as reference. Finally, they classify the ECG using probability methods. The main issue of this project is that the number of training individuals is low (only 24) and the emotional identification lacks nuance to detect muted emotions.

Following the classical methods of acquisition, signal processing and classification we have the [6] that present an approach to the detection of emotional features through the use of ECG and GSR. They have used the Matching Pursuit

algorithm and a Probabilistic Neural Network for the detection of the emotions. The authors have restricted the quantity of emotions to 4, being: scary, happy, sad, and peaceful. These are lifted from the Pleasure Arousal Dominance (PAD) model. The authors used music as activator on 11 students and affirm a high level of accuracy in emotion detection in most of the cases (over 90%). They have discovered that the GSR has little impact for emotions detections. They affirm that the detection of emotions was clear in terms of the arousal, and far less significant in the other fields. [7] shows the ATREC project (a military development) that accesses the stress levels of military personnel through the usage of body sensors. They have discovered that it is possible to determine a high level of valence markers and alert levels from ECG and GSR palaced in the throat, which are directly related to stress levels. Furthermore, their tests have revealed that the speech, GSR (on the hands/arms) or skin temperature provide little additional information about the current emotions. [8,9] presents a study that presents a great accuracy in emotions detections, from the combination of ECG with forehead biosignals. The authors found that actions like frowning or facial movements convey a high level of information about the emotion the subject is feeling. Contradicting the low significance of the usage of the GSR is the [10] that have found that each sensor is related to an emotional field. GSR is related to extremes of valence emotion while ECG is related to emotionally active states (subtle displays of emotion). The authors have determined that direct data fusioning is worst than give different tasks to the sensors. They have found that the activation (when differing from neutral state) should be performed by the GSR, while the classification should be done with the data of the ECG.

As it can be observed from this short sample, there are different approaches (even conflicting ones) to the detection of human emotions with minimal intrusion. Two things are clear from their research, ECG is crucial for the detection and classification of emotions, and that using various sensors may improve the classification accuracy or help detecting trigger events.

3 Description

In this section we describe the proposal of the bracelet-camera that incorporates a camera in a wristband to monitor the elderly. In current years, the use of wearable devices has been growing, devices such as *Samsung*¹ with the *Gear Fit*, *Gear S2*, *Gear S3*, or *Apple*² with the *Apple Watch* are only some examples. These devices can measure heart rate beat or hand movement using the IMU (Inertial Measurement Unit). Based on these devices and using the current technology in embedded systems, it is possible to create new smart bracelets. These new devices have been seen to create applications in IoT, since these new devices are smaller and more powerful.

¹ <http://www.samsung.com>.

² <http://www.apple.com>.

The use of these devices has many fields of application, the most common being in sport, nevertheless, these devices can be used to monitor older people. However, these devices have a problem, as they all need a smartphone to work properly. This device is normally designed to communicate with this smartphone via Bluetooth communication. This means that the smartphone processes and analyses the signals and records user information such as number of slopes per day, sleep analysis, etc. In recent years, new devices with different communication protocols such as WiFi, Bluetooth and LoRa have appeared. All these features are advantageous for the creation of applications for the monitoring of elderly people. This monitoring can be through the acquisition of signals such as electrocardiography (ECG), photoplethysmography (PPG), respiratory rate, etc.

However, it is possible to introduce other types of sensors such as recessed cameras. These cameras can be used as a small device for videoconferencing, emotion recognition or as a tool to know if the person has fallen. However, most of these devices do not take into account the introduction of a camera with new technologies, so it might be possible to create a wristband that has a camera and uses it as another input. A camera can be used in different areas, as a way to recognize emotions, video conferencing and if the bracelet detects that the person has fallen, you can send a message to the caregiver and see the image on your terminal.

To make this application possible, it is necessary to use different technologies not only in hardware, but also in a service that analyses the information sent by the different devices. This service needs to have the capability of pattern recognition, image analysis, emotion analysis, stress detection, etc.

3.1 Hardware Description

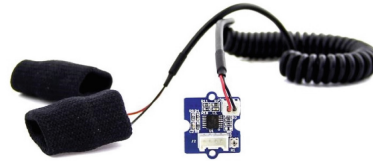
The wristband has been manufactured using different sensors (Fig. 1), which can acquire bio-signals, and a camera that can be used to take pictures of the user. Two sensors were used to acquire the bio-signs, one to detect the heartbeat and the other to measure the skin's resistance. The sensor used to obtain the heartbeat is a PPG sensor (Fig. 1(c)). A PPG is often obtained using a pulse oximeter that illuminates the skin and measures changes in light absorption. A conventional pulse oximeter monitors blood perfusion to the dermis and subcutaneous tissue of the skin. With each heart cycle, the heart pumps blood to the periphery. Although this pressure pulse is somewhat cushioned when it reaches the skin, it is enough to distend the arteries and arterioles in the subcutaneous tissue. If the pulse oximeter is connected without compressing the skin, a pressure pulse can also be seen from the venous plexus, such as a small secondary peak.

The skin is the largest organ in the human body and has different properties, one of which is the ability to vary resistance. This variation has different names such as electrodermal activity (EDA), galvanic skin response (GSR), electrodermal response (EDR) and psychogalvanic reflex (PGR). The galvanic response variation can be used either to detect stress [11], or even emotions [12]. The sensor to detect such variation can be seen in the Fig. 1(b).

At the same time a mini-camera (Fig. 1(a)) is introduced that allows us to use the photos it acquires to analyze the emotional states. This camera (OV2640) has a low voltage CMOS image sensor that provides full functionality of a single-chip VGA camera (an image processor in a small package). The image is processed by a ESP-32 chip. This chip has wifi and bluetooth communication protocols, that make it especially important for IoT applications.



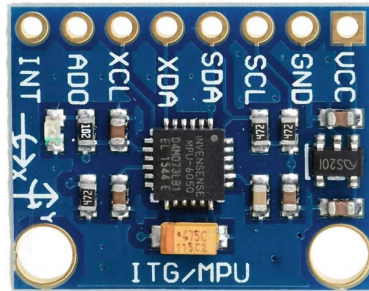
(a) Camera developer board



(b) GSR Sensor



(c) Photo Plethysmography Sensor



(d) Inertial Measurement Unit

Fig. 1. Sensors used in the wristband.

However, this information needs to be processed. As the chip has a low computing power, it is necessary to have a central unit that can be used to manage this information. It is for this reason that we see the need to introduce a cognitive service to analyze the information sent by the sensors and the camera. This cognitive service is explained below.

3.2 Cognitive Service

The Cognitive Service is a new tool that uses a machine learning technique to create smarter and more engaging applications. This cognitive service introduces API to detect emotion, speech recognition, conversion of text to speech and more. Some of the most important services that can be used right now are *Microsoft Cognitive Service (formerly Project Oxford)*³, *IBM Watson*⁴, *Google*⁵ and *AMAZON AWS*⁶.

The cognitive service was divided into two parts, one part specialized in the recognition of emotions through image processing (sending data through the camera) and the other part in which bio-signals are used to recognize emotions (sending data through sensors). These elements are explained below.

Emotion Detection Through Image Processing. The emotion classification uses *Convolutional Neural Networks* [13]. They are very similar to ordinary Neural Networks. They are made up of neurons that have learnable weights and biases. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other.

Our cognitive service can classify 7 emotions: fear, angry, upset, happy, neutral, sad and surprised. To train the model, the KDEP dataset [14], which consists of a total of 980 images, was used. In this dataset the 70 actors, 35 women and 35 men, represent the 7 different emotions. Each image has a resolution of 562 pixels wide and 762 pixels high. Before performing the training, each image is pre-processed to detect and extract the face. Once extracted, a colour transformation is performed in grayscale. Finally, it is necessary to resize the image to 128×128 , and to train the model using Tensor Flow⁷.

The structure of the network was modified changing different parameters such as the number of convolution filters in the different layers. This codification allowed us to obtain the best results. In the end, the best network has the following structure as showed in the Table 1.

Table 1. CNN architecture

	Layer 1	Layer 2	Layer 3
Num input channels	5	5	5
Convolutional layer	32	32	64
Conv filter size	3	3	3

³ <https://azure.microsoft.com/en-us/services/cognitive-services/>.

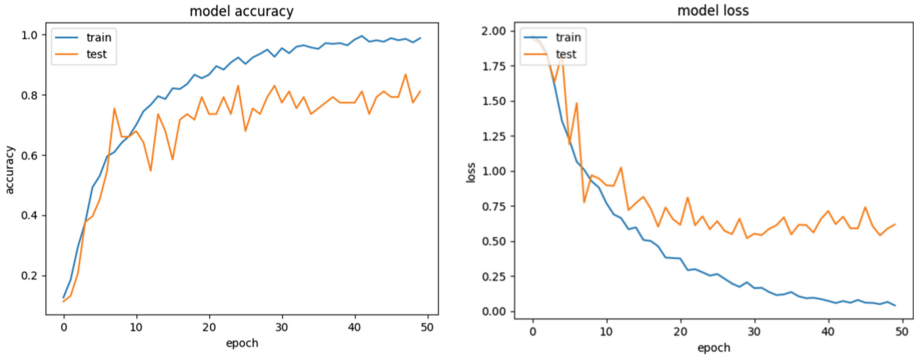
⁴ <https://www.ibm.com/watson/>.

⁵ <https://cloud.google.com/>.

⁶ <https://aws.amazon.com>.

⁷ <https://www.tensorflow.org/>.

Figure 2(a) shows the Model Accuracy in the Train Data and Validation Sets, seeing how the test graph follows the training graph. This behaviour allows the pressure level between the training data set and the test data to be determined. Figure 2(b) shows the Model Loss in the Training and Validation Data Sets. This graph shows the validation process between the training model and our test data.



(a) Model Accuracy on Train and Validation Datasets (b) Model Loss on Training and Validation Datasets

The web service can be used in two modes, a static mode in which the user loads the image and returns the emotion, and a dynamic mode. This second mode is used by any device that has access to web services, allowing them to send the streaming image to the web services, where it is analysed and the web services return the emotion that was detected.

Emotion Detection Through Bio-Signal Processing. To detect emotion through bio-signals we used the DEAP [12] data set, which is structured as a series of participant classifications, physiological recordings and facial video of an experiment in which 32 volunteers viewed a subset of 40 of the previous music videos. The electroencephalogram and physiological signals were recorded and each participant also rated the videos as mentioned above. In our case, to train our deep learning model, we use the GSR and PPG bio-signals. The other signals were dismissed as it is not comfortable to wear an EEG helmet or use an ECG holter to acquire these signals. The signals acquired by the wristband were filtered, we used a butterworth filter to eliminate the noise introduced by the electric field. This process is done in the web service, as the wristband does not have the necessary computational power to perform this filtering.

Figure 2 shows two signals, the input wave is the original signal that, as can be observed, has noise. This noise must be reduced, because if it does not, it may lead to a bad classification. The other signal is a filtering signal, so you can see that the noise is reduced. Once you get this new filtering signal, the next step is to train the network.

In the same way that the parameters of the network used to analyse the images were modified, the network that analyses the signal was modified to try to obtain the best results (Table 2).

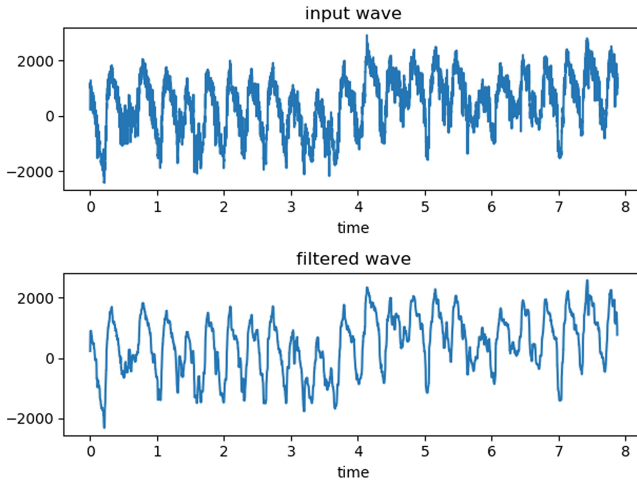


Fig. 2. Unfiltered dataset signal

Table 2. Optimized CNN architecture

	Layer 1	Layer 2	Layer 3
Num input channels	3	3	3
Convolutional layer	32	64	64
Conv filter size	3	3	3

4 Case Study

Elderly people are susceptible of great emotional changes even when subjected to minor interactions, and these changes are aggravated when they suffer from cognitive disorders. The goal of this project is to counterweight environmental changes with positive *stimuli*, thus trying to achieve a neutral/positive emotional state. A case study was designed to subject the elderly people to a set of *stimuli*, in a controlled manner, so that our methods could be validated accordingly.

The case study was done in the *Centro social Irmandade de São Torcato* to test the bracelet and to evaluate if the mixture of both perception methods outputs relevant classification results of emotions. A prototype bracelet (as described in Sect. 3.1) was used by five persons (from now referred to as users) between the ages of 65 and 70, and the test consisted in recognizing the emotions while watching a music video. The users were chosen specifically to be a representation of the elderly community. They are 3 woman and 2 men, with none to mild cognitive disorders, representing the scale of cognitive disorders that allow the bearers to use fairly complex technological devices.

The test operation consists in record the bio-signals for one minute (via the bracelet); next, send a warning vibration (through a motor in the bracelet) to

grab the attention of the user, and concurrently, tracking the user’s face and taking pictures. This information is synchronized with the server for immediate processing.

The server analyses the information received, and outputs the classification error and the mixture of both methods result, as these values are the main goal of this case study. Moreover, the server also outputs the classification of the user’s emotional state. In a production stage the information about the emotional state can be directed to a caregiver, allowing him/her to change the activities/calendar to suit the user’s or to improve their mood.

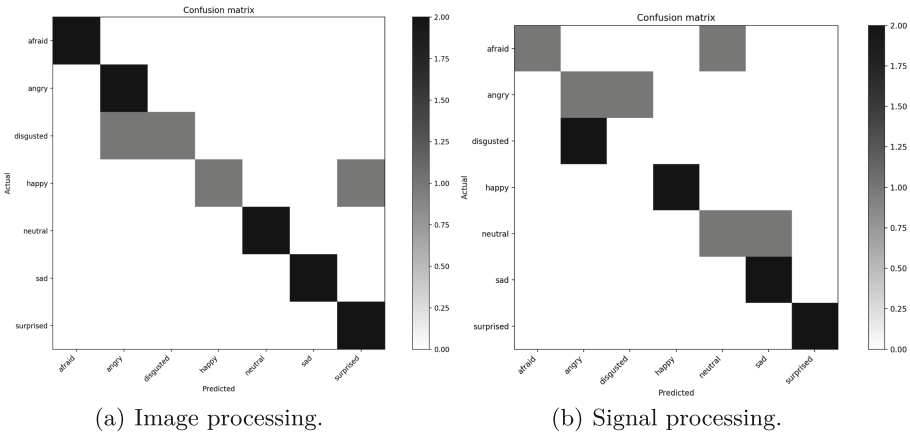


Fig. 3. Confusion matrix obtained from the classification of emotions

The results of this case study are showed in Figs.3(a) and (b), which show the confusion matrix resulting from the classification of emotions through image processing and the confusion matrix resulting from the classification of emotions through signal processing, accordingly. As it can be observed, the integration of the two methods is complementary, achieving 90% accuracy through image processing and 79% through the bio-signal method.

This means that our initial hypothesis is correct and it produces relatively stable results. Although there are clear issues that may undermine the overall results (like the values of bio-signals), we believe that they can be improved by using more accurate or different collection methods (like a body vest with sensors). With these positive results, a production-ready bracelet can be produced to be used by a larger set of users, or in a group environment to test social interactions.

5 Conclusions and Future Work

In this work we have presented how to integrate non-invasive bio-signals and cameras for the detection of human emotional states into an intelligent wrist-

band. In this way, the proposed wristband integrates two ways to detect human emotions to improve the emotional detection.

The main application domain of the designed wristband is the elderly care in nursing homes, but it can be used in other types of domains.

The wristband is able to recognize emotions through the processing of images and biological signals. To do this, the wristband acquires the bio-signals and images through the corresponding sensors and sends them to a cognitive service for the analysis. The proposed approach is being validated by workers and patients of a daycare centre *Centro Social Irmandade de São Torcato*. The validation is being performed through simple interactions with the patients under the supervision of caregivers.

Preliminary results show that the wristband is well accepted by the elderly people resident of the center. Future work will initially focus on the development of new tests with a larger number of users. These new tests will allow the wristbands to be used to improve the activities and tasks performed at the centre in order to avoid situations where older people have undesirable emotional states.

Acknowledgements. This work is supported by COMPETE: POCI-01-0145-FEDER-007043 and FCT - Fundacao para a Ciencia e Tecnologia within the projects UID/CEC/00319/2013 and Post-Doc scholarship SFRH/BPD/102696/2014 (Angelo Costa). This work is partially supported by the MINECO/FEDER TIN2015-65515-C4-1-R.

References

1. United Nations, Department of Economic and Social Affairs, Population Division. World population ageing 2015. (ST/ESA/SER.A/390) (2015)
2. United Nations, Department of Economic and Social Affairs, Population Division. World population prospects: the 2017 revision, key findings and advance tables. (ESA/P/WP/248) (2017)
3. Smith, T.W., Glazer, K., Ruiz, J.M., Gallo, L.C.: Hostility, anger, aggressiveness, and coronary heart disease: an interpersonal perspective on personality, emotion, and health. *J. Pers.* **72**(6), 1217–1270 (2004)
4. Richman, L.S., Kubzansky, L., Maselko, J., Kawachi, I., Choo, P., Bauer, M.: Positive emotion and health: going beyond the negative. *Health Psychol.* **24**(4), 422 (2005)
5. Brás, S., Ferreira, J.H.T., Soares, S.C., Pinho, A.J.: Biometric and emotion identification: an ECG compression based method. *Front. Psychol.* **9**, 467 (2018)
6. Goshvarpour, A., Abbasi, A., Goshvarpour, A.: An accurate emotion recognition system using ECG and GSR signals and matching pursuit method. *Biomed. J.* **40**(6), 355–368 (2017)
7. Seoane, F., et al.: Wearable biomedical measurement systems for assessment of mental stress of combatants in real time. *Sensors* **14**(4), 7120–7141 (2014)
8. Naji, M., Firoozabadi, M., Azadfallah, P.: A new information fusion approach for recognition of music-induced emotions. In: *IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)*. IEEE, June 2014

9. Naji, M., Firoozabadi, M., Azadfallah, P.: Emotion classification during music listening from forehead biosignals. *Signal, Image Video Process.* **9**(6), 1365–1375 (2015)
10. Das, P., Khasnobish, A., Tibarewala, D.N.: Emotion recognition employing ECG and GSR signals as markers of ANS. In: 2016 Conference on Advances in Signal Processing (CASP). IEEE, June 2016
11. Villarejo, M.V., Zapirain, B.G., Zorrilla, A.M.: A stress sensor based on galvanic skin response (GSR) controlled by ZigBee. *Sensors* **12**(5), 6075–6101 (2012). (Switzerland)
12. Koelstra, S., et al.: DEAP: a database for emotion analysis; using physiological signals. *IEEE Trans. Affect. Comput.* **3**(1), 18–31 (2012)
13. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*, pp. 1097–1105 (2012)
14. Calvo, M.G., Lundqvist, D.: Facial expressions of emotion (KDEF): identification under different display-duration conditions. *Behav. Res. Methods* **40**(1), 109–115 (2008)



Applying Cost-Sensitive Classifiers with Reinforcement Learning to IDS

Roberto Blanco¹, Juan J. Cilla¹, Samira Briongos¹,
Pedro Malagón^{1,2}(✉), and José M. Moya^{1,2}

¹ Integrated Systems Laboratory, Universidad Politécnica de Madrid ETSI
Telecomunicación, Av. Complutense 30, 28040 Madrid, Spain

malagon@die.upm.es

² Center for Computational Simulation, Universidad Politécnica de Madrid,
Campus de Montegancedo, 28660 Boadilla del Monte, Madrid, Spain

Abstract. When using an intrusion detection system as protection against certain kind of attacks, the impact of classifying normal samples as attacks (False Positives) or attacks as normal traffic (False Negatives) is completely different. In order to prioritize the absence of one kind of error, we use reinforcement learning strategies which allow us to build a cost-sensitive meta-classifier. This classifier has been build using a DQN architecture over a MLP. While the DQN introduces extra effort during the training steps, it does not cause any penalty on the detection system. We show the feasibility of our approach for two different and commonly used datasets, achieving reductions up to 100% in the desired error by changing the rewarding strategies.

Keywords: Intrusion detection · Cost-sensitive classification
Reinforcement learning

1 Introduction

The task of a Network Intrusion Detection System (NIDS) is to detect suspicious traffic patterns to react and stop an attack against the system. There are several approaches to face the detection problem, including the use of machine learning algorithms to detect such packets. Multiple supervised and unsupervised methods have been proposed, and tested with publicly available datasets. The metrics used to evaluate the classifiers performance use the False and True Positives (classified as attack) and the False and True Negatives (classified as normal traffic). Additionally, many datasets are not class balanced, having much more records of one type of sample.

The impact of a misclassification error is asymmetrical in many applications. For instance, a preliminary algorithm to discard medical diseases based on images must be accurate when classifying a sample as benignant, and might introduce some errors when classifying it as a disease, which raises an alert to use other methods to confirm the diagnosis. The cost of a misclassification varies from

one class to another depending on the application. The most popular classifiers used as NIDS are cost-insensitive, and do not consider the imbalances. As their performance in a concrete application depends on the imbalances described, there are metrics that compensate these imbalances when evaluating classifiers.

Cost-insensitive algorithms can be transformed into cost-sensitive algorithms, by modifying their parameters or using meta-algorithms to lean the classification method to one class. We propose the use of Deep Q-Networks (DQN), a reinforcement learning algorithm (RL), to transform a cost-insensitive Neural Network into a cost-sensitive classification algorithm.

2 Related Work

The multilayer perceptron (MLP) has been evaluated with KDD99, considering new attacks [4], with NSL-KDD [9] and with UNSW-NB15 [8]. The training process of the MLP reduces the global amount of errors performed by the classifier, with no distinction between false positives or false negatives, and ignoring the ratio between attacks and normal traffic in the dataset. There are many proposals to deal with the class imbalance in a dataset, as summarized in [3]. There are proposals to lean the scale of the classification to focus on reducing one type of errors in scenarios where the impact of misclassification is asymmetrical. Cost-sensitive learning methods or classifiers have been proposed to solve both problems [5]. The two main approaches are thresholding [10], which modifies the threshold of the output to assign a record to a class, and weighting [11], which treats samples differently depending on the label. Both algorithms were compared applied to neural networks in [12] showing similar results for the class imbalance problem. Reinforcement Learning was tested for IDS with NSL-KDD, with Q-learning algorithm, in [2]. Reinforcement learning applied to solve the asymmetry in the misclassification has never been tested.

3 Proposal

3.1 Reinforcement Learning

Reinforcement Learning is a learning technique used when the environment is unknown. An RL agent aims to predict how the environment will react to each action at every state and to figure out which actions would yield the most favourable response. The environment reacts to the applied action, affecting not only the next step but future states, making it harder to find the optimal behaviour. A classical approach to the RL problem uses the Markov Decision Process (MDP) formulation as a simple model of the interaction between the agent and its environment. The agent is able to sense the state of the environment and to take actions to obtain some reward.

The main elements of a RL system are: (1) environment state (S_t) as observed by the agent; (2) actions that can take the agent (A_t); (3) rewards given by the environment (R_t). Figure 1 shows the RL process in a typical system.

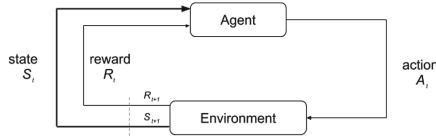


Fig. 1. RL process

3.2 DQN Architecture

Deep Reinforcement Learning (DRL) combines Reinforcement Learning with Deep Neural Network approaches. Deep Q-Networks (DQN) were introduced in [6] to overcome the main problems of Neural Fitted Q-iteration (NFQ). This method uses an Artificial Neural Network (ANN) to approximate the Bellman equation $[B_{k, \Theta_{t,g}} = r_k + \gamma * \max_{a'} q_{\Theta_{t,g}}(s_{k+1}, a')]$, where $B_{k, \Theta_{t,g}}$ is the approximate Bellman operator for the transition k following the $\Theta_{t,g}$ policy; r_k is the instantaneous reward given by the environment to the action taken; γ is a coefficient which sets the importance of the future decisions; and $\max_{a'} q_{\Theta_{t,g}}(s_{k+1}, a')$ is the estimation of the maximum value function or long term reward expected by following the optimum path of the $\Theta_{t,g}$ policy. The DQN architecture aims to learn the parameters that minimize the approximate Bellman error.

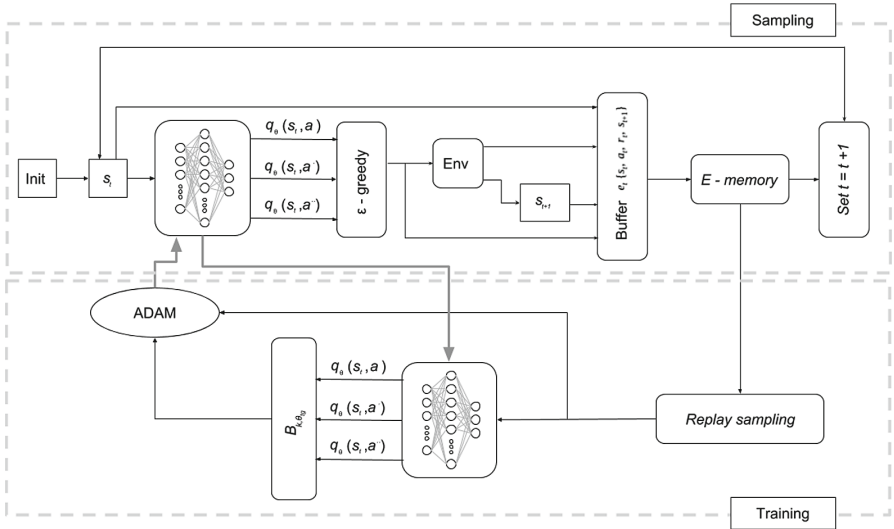


Fig. 2. DQN block architecture

As it is shown in Fig. 2, a DQN architecture has two different blocks: Sampling and training. The sampling block is responsible for generating the training data. It contains an artificial neural network (N_{SA}) which is fed with the features of

every state in every instant t (S_t). The last layer of this neural network must have the same number of neurons than the actions of the agent in the problem. This network is followed by a decision block called epsilon-greedy. This block has the task of selecting one of the possible actions of the agent depending on the outputs of the previous neural network. The action (a_t) is chosen based on a parameter epsilon which is the probability of selecting the action which corresponds to the output with the maximum value (exploit the knowledge of the network) or randomly select another one (explore the environment to learn about it). Once the action has been chosen, it is included in the environment which returns a certain reward (r_t) and the next state for the given state and action (S_{t+1}). This four pieces of information are finally stored together as a tuple in a buffer of memory and the next state returned is selected to be the next state fed to the network after the training step.

The training block works sequentially with the sampling one. Every time a sample is stored in the memory buffer, a small number of tuples is selected from the whole buffer memory and placed in a minibatch. Then, a second neural network (N_{TG}) with the same structure as the first one N_{SA} is fed with the state and next state features of the minibatch and with its predictions. Then, the approximated Bellman equation value for all the tuples [$B_{k,\theta_{tg}} = r_k + \gamma * \max_{a'} q_{\theta_{tg}}(s_{k+1}, a')$] is computed. This equation intends to approximate the expected future reward given by the environment (for the state and action given) making use of the nonlinear $\max()$ operator, to the outputs of the N_{TG} for the next state and the concept of bootstrapping. When the approximated values are computed, the sampling network N_{SA} is trained using a back-propagation optimization method and the minibatch state features as inputs.

The N_{SA} network is trained in every iteration with the information stored in the tuples of the minibatch and the values of the approximated Bellman equation whereas the N_{TG} network is never trained, its weights are copied from the first one at the end of every epoch, which consist in many iterations.

3.3 DQN Network Traffic Classifier

We propose to use a DQN to implement a cost-sensitive network traffic classifier that minimizes the rate of false alarms or false negatives when distinguishing connections that are performing different kind of attacks from regular traffic. We don't consider Deep Packet Inspection nor session or login information, as this information would be more complex to obtain in a firewall when doing online classification. We consider connection oriented traces, a subset of the features available in the public datasets NSL-KDD and UNSW, that can be obtained from the packets seen in a router as input features and that are not categorical. In the NSL-KDD there are 25 features matching our criteria, while in the UNSW there are only 23 features. The Neural Network we propose in the DQN architecture consist in a very simple multilayer perceptron of 23–25 neurons in the input layer depending on the dataset used, 128 neurons followed by an hyperbolic tangent activation function in the hidden layer, and two neurons in the output layer. We have chosen the hyperbolic tangent activation function

instead of others like rectified linear unit or just linear unit because we wanted to grant the convergence of the model and we had big negative reward values. The epsilon-greedy value is a decaying value starting in 0.9 and ending in 0.1 in order to maximize the exploration at the beginning and exploit the knowledge acquired in long time. The size of the minibatch is 8 tuples and the size of the buffer memory is equal to the total number of samples in the training data.

As we are working with datasets, we don't have a real online environment to include in our model, so we have made some modifications so it can be suitable for our purpose. We are considering a random order of the samples in the dataset in order to make the environment sequence of states. Also the gamma value to weight the long term expected reward has been set to 0 because we want the model to learn properly the instantaneous reward for every sample in the dataset independently of the order of them. Besides, we make balanced rewards according with the different kind of mistake instead of using the numerical value of the dataset labels directly.

4 Experimental Setup

4.1 Dataset

First, we consider the UNSW dataset [7] that includes modern attacks, with 10 different classes of traffic: regular traffic and 9 different categories of attacks. Background traffic is synthetic and the traces are not balanced between attacks and non-attacks. From the 49 features included in each entry we discard the categorical features and the features that cannot be obtained in a router, reducing the input set to 23 features. We focus our classifier in detecting and classifying attacks in TCP traffic, which is the predominant kind of traffic in the original dataset with a 58% of the traces. In order to maximize the training fitness, the subset is binary balanced, reducing the amount of attacks. We divide TCP-only dataset entries in 10 subsets, 9 of them used to train the system and the other one to validate the results.

Second, we consider the NSL-KDD training set [1], with 24 attack classes that are grouped into 4 major categories. Following the same guidelines for feature selection, we select 25 of the 41 original features. Remaining entries are divided into 10 subsets as well, 9 for training and 1 for validation. The subsets are balanced in entries between the 4 categories: regular traffic and the three selected attack categories.

For testing purposes, we consider both complete datasets to get the metrics with all available entries. NSL dataset includes attacks that have not being included in any entry of the training set. Dataset entries are normalized by columns using 0 mean and 1 standard deviation.

4.2 Setup

We conduct a set of experiments using different rewards to optimize the result of the classifier. We use a Simple Multilayer Perceptron (MLP) as baseline with

each dataset. It has one hidden layer with 128 cells and one output layer with 2 cells: attack or non-attack. We implement the proposed DQN architecture on this MLP with different goals and rewarding strategies. We pursue three different goals in each experiment: reducing every error, reducing the number of false positives or the number of false negatives. For the first one, we use balanced rewards, +1 for success and -1 for an error. In order to perform an asymmetric training, we follow two strategies: soft unbalanced, +1 for correct prediction, -10 for less cost error and -100 for the desired reduction error; or hard unbalanced, +1 for success, -1 for less cost error and -100 for the desired reduction error. These approaches result in 5 experiments with each dataset.

The experimental implementation has been developed using Python3.5 with the following libraries: Keras version 2.1.2, TensorFlow version 1.1.0, Numpy version 1.13.0, Scipy version 0.19.0 and Pandas version 0.20.2.

5 Results

UNSW Dataset Classifier. The results obtained using a balanced rewards strategy are similar to MLP, as expected. With a balanced training the result is the optimal cost-insensitive. Table 1 shows the results obtained for the MLP and the other reinforcement strategies for the validation subset (and for the full dataset in parenthesis). The metrics included are the raw values obtained, false and true negatives and positives, and the metrics that show the performance of the algorithm, accuracy, sensitivity, specificity and F1. When we unbalance the negative rewards for the FPOS and FNEG predictions to -10 and -100 respectively (Soft FNEG), we manage to reduce the number of FNEG predictions from 1 to 0 in the validation test and from 9 to 2 in the full dataset test, which is a reduction of 100 and 78%. If we set the rewards to -1 and -100 in the same cases (Hard FNEG) we manage to increase the reduction from 78 to 89%. On the other hand, when we lean the balance the other way for the FPOS and FNEG, the number of FPOS is reduced from 78 to 46 in validation and from 24009 to 15426 in the full test using the Soft FPOS imbalanced rewards, which corresponds to a reduction of 41 and 36%. We get a bigger decreasing if we set the Hard FPOS rewards, with a reduction of 100 and 98%. In this case it is remarkable that the number of FNEG has increased above the number of attacks detected, which is far away from desirable in a real application and makes the model quite inaccurate.

NSL-KDD Dataset Classifier. The experiments using balanced rewards leads again to the optimal balanced solution found with the simple MLP, where the number of prediction errors depends more on the structure and nature of the data itself than on the chosen predictive model. In the same way, we present in the Table 2 the results obtained in both tests for the model trained with the NSL-KDD data. We manage to reduce the number of FNEG predictions in the validation test a 87 and 100% using Soft and Hard imbalanced rewards. The same models show as well a decreasing in the full dataset test of 70 and 96%.

Table 1. UNSW classifier with validation data

	Simple MLP	Soft FNEG	Hard FNEG	Soft FPOS	Hard FPOS
TNEG	5697 (1412881)	5860 (1409157)	5643 (1398077)	5729 (1421464)	5775 (1436382)
TPOS	5860 (58175)	5861 (58182)	5861 (58183)	5343 (53282)	1021 (10499)
FPOS	78 (24009)	98 (27733)	132 (38813)	46 (15426)	0 (508)
FNEG	1 (9)	0 (2)	0 (1)	518 (4902)	4840 (47685)
Accuracy	0.993 (0.984)	0.992 (0.981)	0.989 (0.974)	0.952 (0.986)	0.584 (0.968)
Sensitivity	1 (1)	1 (1)	1 (1)	0.912 (0.916)	0.174 (0.180)
Specificity	0.986 (0.983)	0.983 (0.981)	0.977 (0.973)	0.992 (0.989)	1 (1)
F1-Score	0.993 (0.829)	0.992 (0.808)	0.989 (0.750)	0.959 (0.840)	0.297 (0.303)

With the number of FPOS predictions the reduction is 100% for both kinds of rewards in the validation test and 98 and 97% in the full dataset test. With this dataset it seems that the normal samples are easier to be classified so it is harder to reduce the number of FNEG predictions as it is shown in the results with a smaller reduction for the same imbalanced rewards.

Table 2. NSL-KDD classifier with validation data

	Simple MLP	Soft FNEG	Hard FNEG	Soft FPOS	Hard FPOS
TNEG	4841 (53438)	4724 (52122)	4043 (44472)	4865 (53596)	4865 (53595)
TPOS	4913 (48803)	4939 (48969)	4943 (49030)	4807 (47761)	4677 (46476)
FPOS	24 (162)	141 (1478)	822 (9128)	0 (4)	0 (5)
FNEG	301 (237)	4 (71)	0 (10)	136 (1279)	266 (2564)
Accuracy	0.994 (0.996)	0.985 (0.985)	0.916 (0.911)	0.986 (0.988)	0.973 (0.975)
Sensitivity	0.994 (0.995)	0.999 (0.999)	1 (1)	0.972 (0.974)	0.946 (0.948)
Specificity	0.995 (0.997)	0.971 (0.972)	0.831 (0.830)	1 (1)	1 (1)
F1-Score	0.995 (0.996)	0.986 (0.984)	0.923 (0.915)	0.986 (0.987)	0.972 (0.973)

6 Conclusions

Depending on the system, it is more important for the IDS to focus on reducing False Negatives (FNEG) or on reducing False Positives (FPOS) rather than reducing errors globally. We propose using reinforcement learning to implement a cost-sensitive meta-classifier to prioritize the absence of one kind of error. We select a DQN architecture over a MLP for our prototype. The DQN introduces extra effort in training, with no penalty on testing or online execution. We trained and tested it with two different IDS datasets, NSL-KDD and UNSW, representing two different scenarios with different properties.

Changing the values of the rewards for each error we introduce asymmetry to the training system. The results show that balanced rewards are similar to

having no RL. Using UNSW, with 3% of attacks, attacks are easier to detect, with a lower ratio of FNEG. Using NSL-KDD, with 43% of attacks, normal traffic is easier to detect, with a lower ratio of FPOS.

Using the proposed architecture, we perform a soft and a hard imbalance strategy for FPOS and for FNEG with each dataset. The soft strategy reduces the amount of selected errors with a low overhead in the other class. It is successful when the Simple MLP shows better results in the selected error: soft FNEG with UNSW and soft FPOS with NSL-KDD. The hard strategy reduces the errors in the selected class, no matter the selected dataset, introducing a greater overhead in the other class errors. The rewarding strategy depends on the traffic distribution of the scenario and the selected goal.

Acknowledgements. This work was supported by the Spanish Ministry of Economy and Competitiveness under contracts TIN-2015-65277-R, AYA2015-65973-C3-3-R and RTC-2016-5434-8.

References

1. NSL-KDD data set for network-based intrusion detection systems, March 2009. <http://nsl.cs.unb.ca/NSL-KDD/>
2. Sengupta, N., Sen, J., Sil, J., Saha, M.: Designing of on line intrusion detection system using rough set theory and q-learning algorithm. *Neurocomputing* **111**, 161–168 (2013)
3. He, H., Garcia, E.A.: Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.* **21**(9), 1263–1284 (2009)
4. Kukielka, P., Kotulski, Z.: Analysis of neural networks usage for detection of a new attack in IDS. *Ann. UMCS Inf.* **10**(1), 51–59 (2010)
5. Kulluk, S., Özbak, L., Tapkan, P.Z., Baykasoglu, A.: Cost-sensitive meta-learning classifiers. *Know.-Based Syst.* **98**(1), 148–161 (2016)
6. Mnih, V., et al.: Playing Atari with deep reinforcement learning. *CoRR* abs/1312.5602 (2013)
7. Moustafa, N., Slay, J.: UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: *Military Communications and Information Systems Conference (MilCIS)*, pp. 1–6. *IEEE Stream* (2015)
8. Moustafa, N., Slay, J.: The evaluation of network anomaly detection systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Inf. Secur. J.: Glob. Perspect.* **25**, 1–14 (2016)
9. Revathi, S., Malathi, A.: A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. *Int. J. Eng. Res. Technol.* *ESRSA Publications* **2**(12), 1848–1853 (2013)
10. Sheng, V.S., Ling, C.X.: Thresholding for making classifiers cost-sensitive. In: *Proceedings of the 21st National Conference on Artificial Intelligence, AAAI 2006*, vol. 1, pp. 476–481 (2006)
11. Ting, K.M.: Inducing cost-sensitive trees via instance weighting. In: Żytkow, J.M., Quafafou, M. (eds.) *PKDD 1998. LNCS*, vol. 1510, pp. 139–147. Springer, Heidelberg (1998). <https://doi.org/10.1007/BFb0094814>
12. Zhao, H.: Instance weighting versus threshold adjusting for cost-sensitive classification. *Knowl. Inf. Syst.* **15**(3), 321–334 (2008)



ATM Fraud Detection Using Outlier Detection

Roongtawan Laimek^{1(✉)}, Natsuda Kaothanthong¹, and Thepchai Supnithi²

¹ Sirindhorn International Institute of Technology, Thammasat University,
Bangkok, Thailand

`roongtawan.l@dti.or.th, natsuda@siit.tu.ac.th`

² Language and Semantic Technology (LST) Laboratory, National Electronics
and Computer Technology Center (NECTEC), Khlong Neung, Thailand
`thepchai.supnithi@nectec.or.th`

Abstract. A fraud detection model that applies accounts' behavior features and outlier detection methods are proposed. Given a set of transactions, the accounts are grouped into two groups according to the location that the transactions took place, i.e., 'local-only' and 'has-abroad'. A feature is extracted to reflect the normal behavior of the accounts in each group. Only known legitimate transactions are used to extract a set of features for representing a legitimate behavior. An unknown transaction is classified either normal or fraud using an outlier detection. The experimental result shows that the proposed feature with an Isolation Forest outlier detection technique is able to detect all fraud transactions.

Keywords: ATM fraud · Data mining · Outlier detection

1 Introduction

ATM fraud is a financial fraud, in which the fraudster uses a stolen or fake card to withdraw money from a victim's account from the ATM machine [1]. Comparing to other types of financial fraud, ATM frauds is intermittent. However, each incident can cause significant loss to the victim's account. To maintain customers' trusts, banks deploy a number of fraud prevention such as using chip card to secure the data on the card and detecting any suspicious transactions by human investigators.

As a number of transactions has extensively growth, manual fraud detection becomes infeasible. In this way, the classification methods based on machine learning has been introduced [2–12]. Dorronsoro et al. [2] proposed a credit card fraud detection system called Minerva. This classification system is based on multi-layer perceptron (MLP) neuron network, which is feasible to perform a real time fraud detection. CARDWATCH [3] applied a a three-layers feed-forward MLP neural network architecture. The system can detect 85% of fraudulent transactions and 100% of non-fraudulent transactions.

The major difficulty of fraud detection systems is an unbalanced data between frauds and non-fraud transactions. For this reason, the detection system is able to classify the non-fraud cases rather than the fraud. Instead of relying on the characteristics of fraud transactions, the normal transaction is being focused. The transactions whose characteristics deviate from normal transactions is considered as an ‘outlier’ or an abnormal transaction [13]. Many outlier techniques have been proposed [10–12]. CoDetect [10] applied graph-based anomaly detection to detect financial frauds based on transaction network, and feature-based anomaly detection. The model worked well in money laundering. Wu et al. [11] proposed technique based on coevolutionary algorithm to detect frauds in Automated Banking Machine (ABM) transactions using an algorithm that evolves a boundary between normal and abnormal transactions. Yamanishi et al. [12] proposed SmartSifter (SS). It is a probabilistic model that can be generated with the new input, and be compared with the previous model to find any outliers.

From our observations, the fraud ATM transactions mostly occurred in the accounts that had transactions abroad. In this work, the accounts are grouped as ‘local-only’ if the transactions of the account were only took place domestically. Otherwise, they are grouped as ‘has-abroad’. A set of features are extracted from the legitimate transactions to represent a normal behavior. Given a feature of an unknown transaction, it is transformed and classified by outlier detection technique.

The rest of this paper is organized as follows. Section 2 presents the previous works. Section 3 explains the data set and analysis of raw transactional data. Section 4 presents the proposed method. Section 5 presents the experimental result and the discussion. The conclusion can be found in Sect. 6.

2 Previous Works

Many methods have been proposed for detecting outliers. Most outlier detection approaches identify outliers as observations that do not conform to the normal characteristics by measuring distances, or densities.

Local outlier factor (LOF) [14] is a density-based method that relies on a nearest neighbors search. It scores each observation by computing the ratio of the average densities of the observation’s neighbors to the density of the observation itself. The degree of being an outlier is assigned to indicate whether or not the observation is an outlier. The computation of LOF value for each observation requires a number of k -nearest neighbors queries that causes a high computation time and space [15].

Isolation Forest algorithm [16] isolates observations in each tree level by randomly selecting a feature, then randomly selects a split value between the maximum and minimum values of the selected feature. Isolation Forest is an algorithm with a low linear time complexity and a small memory requirement. It also has ability to deal with high dimensional data with irrelevant attributes.

3 ATM Data Analysis

In this paper, the ATM transactional data was obtained from a bank in Thailand. It was collected from 1/1/2017 to 31/12/2017. The features of each transaction are described in Table 1.

In this work, a preprocess is applied to obtain only a withdrawal transaction from an ATM machine and the transaction amount greater than zero are applied. In addition, the transactions, which the terminal country is unknown, are removed from the data set. In total, there are 1,045,847 transactions where 227 transactions of 43 accounts are frauds.

Table 1. Description of an ATM transaction

Features	Description
ACCOUNT_ID	Refers to the account number of ATM card number
CARD_NO	Refers to the card number that was inserted
TRANSACTION_DOM	Date of month between 1 and 31
TRANSACTION_DOW	Day of week between 0 and 6
TRANS_AMT	The amount of money in the transaction
TRANSACTION_CD	Refers to the result of transaction
TERMINAL_ID	Refers to the ATM terminal
TERMINAL_BANK_ID	Refers to the bank owner of the ATM terminal
TERMINAL_COUNTRY	Refers to a country or a province in Thailand
TRANSACTION_TYPE	Refers to the type of the transaction
VELOCITY	The rate of change in locations between consecutive transactions
FRAUD_FLAG	The transaction is either non-fraud (0) or fraud (1)

Among 227 fraud transactions, 35 are local (Thailand) and 192 are abroad. The frauds rates are 0.0033 % of local transactions and 18.69 % of abroad transactions. From the percentage of frauds, the transactions that took place abroad have a higher chance to be a fraud. Figures 1a and b shows top countries and top provinces, which had high percentages of frauds, respectively.

By considering the location that the transactions took place, there are 10,552 accounts that only have local transactions and there are 188 accounts that made both local and abroad transactions. Among the transactions from 10,552 accounts, 0.003% are fraud. On the other hand, 1.085% of the transactions from 188 accounts are frauds. From these observations, it can be concluded that the account which has abroad transactions have a higher chance to encounter a fraud.

4 Methodology

The overview of the proposed method can be found in Fig. 2. The transactions are separated by the account number. Each account has already been classified

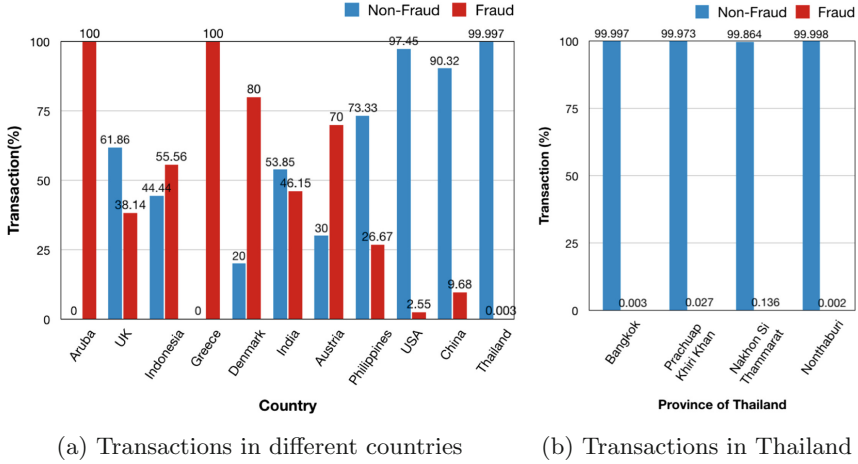


Fig. 1. Percentages of frauds and non-frauds in different locations.

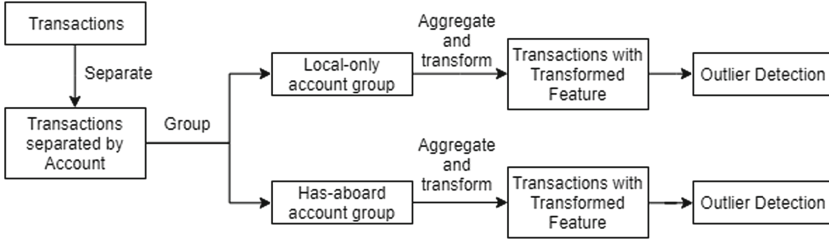


Fig. 2. The ATM fraud detection model.

as either ‘local-only’ account or ‘has-abroad’ account. In each class, we extract normal behaviors from each account and use them to transform each transaction. The result is the transactions with transformed features. Finally, outlier detection is applied to identify outliers as frauds.

4.1 Normal Behavior Extraction

To obtain normal behavior, only normal transactions are used to determined the aggregated features for characterizing the specific behavior of each account. The aggregated features are computed from numeric features and non-numeric features. Given a set of all normal transactions A_n of an account n , let $a_n(f)$ be an aggregated numeric feature of a feature f of an account n , which can be defined as follow:

$$a_n(f) = \frac{1}{size(A_n)} \sum_{i \in A_n} f_i \tag{1}$$

where f_i is a feature value of a transaction i .

Let $a_n(f')$ be an aggregated non-numeric feature of a feature f' of an account n . It can be defined as in (2), where x_i is the amount of the transaction i in the set of normal transaction of an account n denoted by A_n .

$$a_n(f') = \frac{1}{\text{size}(A_n(f'))} \sum_{i \in A_n(f')} x_i \quad (2)$$

In this paper, the normal behavioral model of an account consists of aggregated features derived from transaction amount, velocity, day of week, date of month, terminal ID, bank ID and terminal country. Only transaction amount and velocity are numeric features, and the rest are non-numeric.

4.2 Transaction Aggregation and Transformation

The original features of a transaction contain information specific to only one transaction, which has less information related to the owner's account behavior. In this way, the original features are aggregated to represent the account's behavior. A confidence value is derived from the aggregated features to determine the deviation from the normal behavioral feature.

Let $C(f_i)$ be a confidence on the value of feature f_i for each transaction i in an account n as shown in (3), where f_i is the value of the feature in the transaction i and $a_n(f)$ is the aggregated feature of f for the account n . The value is computed for the feature f of transaction amount, velocity, day of week, date of month, terminal ID, bank ID, and terminal country.

$$C(f_i) = \frac{f_i - a_n(f)}{a_n(f)} \quad (3)$$

In addition to confidence values, the risks value is computed from the known fraud transaction. Let $R(f_i)$ be a risk on the feature f_i of each transaction i in an account n . It can be computed as follow:

$$R(f_i) = \frac{\tilde{K}(f_i)}{K(f_i)} \quad (4)$$

where $\tilde{K}(f_i)$ is the number of known frauds that associate with value of feature f_i and $K(f_i)$ is the number of all known transactions that associate with value of feature f_i . The risk values are determined for feature f of terminal ID, bank ID and terminal country.

4.3 Outlier Detections

The distributions of data on transformed features are shown in Fig. 3(a) and (b). It can be seen that the features are able to separate frauds from non-fraud transactions. In this work, an outlier detection is applied to identify the transactions whose transformed characteristics are not conformed to the characteristics of

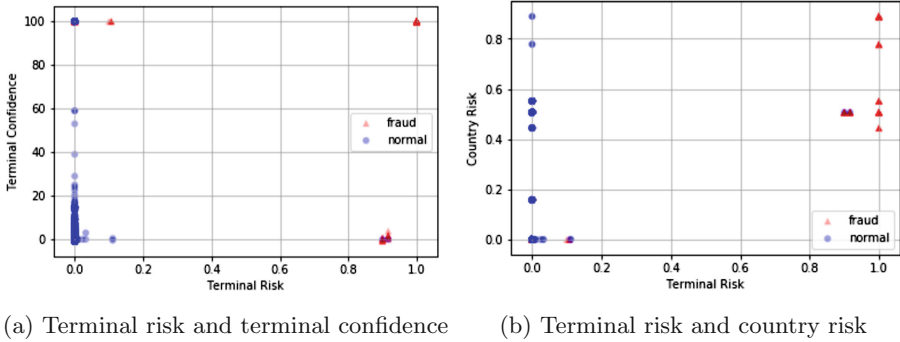


Fig. 3. Distributions of transformed transactions on different features. A blue circle is a normal transaction and a red triangle is a fraud. (Color figure online)

the normal transactions. Two outlier detection techniques are considered: local outlier (LOF) factor and isolation forest.

LOF utilizes two parameters, which are number of neighbors and contamination. The number of neighbors is the minimum number of k -nearest neighbors of each transaction required to classify the transaction as normal. Contamination is estimated ratio of frauds in population, which can be set between 0 and 0.5. Isolation forest utilizes the maximum number of samples, maximum number of features, and contamination that was randomly chosen by an algorithm to train each isolation tree.

5 Experiment

For ‘local-only’ group, 735,865 transactions are used for constructing a model and 289,351 transactions for testing. Among testing transactions, the number of fraud transactions is 16. For ‘has-abroad’ group, 12,939 transactions are used for constructing a model and 3,391 transactions for testing. Among testing transactions, the number of fraud transactions is 16.

The aggregated and transformed of ATM transactions for each ‘local-only’ and ‘has-abroad’ were split into two sets for training and testing. The first data set was the transactions that were took place between 01/01/2017 to 30/09/2017. The second data set was the rest of the transactions.

Four outlier detection models were used in this experiment. The models were implemented in SciKit-Learn library [17]. The performance of ATM fraud detection model is evaluated in terms of confusion matrix. Confusion matrix provides the number of frauds correctly predicted as frauds (TP), the number of normal transaction correctly predicted as normal transactions (TN), the number of normal transactions incorrectly predicted as frauds (FP) and the number of frauds incorrectly predicted as normal transactions (FN).

5.1 Has-Abroad Dataset

The parameters setting for LOF model is 4 neighbors and 0.5 contamination. For Isolation Forest model, the number of max_samples is 1000, a max_features is 8, and a contamination is 0.05. For Neural Network, the hidden layers of 40, 20, 30, and 10 are used. Lastly, Random Forest model applies 10 estimators with maximum tree depth of 10 in each estimator.

Table 2 shows the confusion matrix of the ‘has-abroad’ set. It can be seen from the result that the Isolation Forest can detect all the fraud transactions. It achieves 91.57% accuracy, which is the highest. On the other hand, LOF can detect 50% of the fraud transactions and the accuracy is 54.11%. The neural network and random forest models were also constructed, but cannot detect any fraud transactions.

Table 2. Confusion matrix of ‘has-abroad’ account group.

Predicted	LOF		Isolation forest		Neural network		Random forest	
	Fraud	Not fraud	Fraud	Not fraud	Fraud	Not fraud	Fraud	Not fraud
Fraud	8	8	16	0	0	16	0	16
[wd]Not fraud	1548	1827	286	3089	0	3375	44	3331

5.2 Local-Only Dataset

For LOF model, the true positive rate is 18.75%. It can detect only 3 out of 16 fraud transactions. For Isolation Forest model, the true positive is 81.25% and the false positive is 20.46%. Although the false positive rate is low, but 59,737 normal transactions of local-only account group identified as frauds. The Neural Network and Random Forest models could not detect any frauds. However, Neural Network can detect all legitimate transactions.

5.3 Cost for Handling Fraud Cases

In financial industry, the cost of processing the fraud is concerned. Each detection or missed detection induces a cost. The costs of processing the predicted fraud refer to investigation costs (p). The cost of unnoticed fraud transactions (u) costs a lot of damage. The total cost is $c = p \cdot (TP + FP) + u \cdot FN$.

In this experiment each cost is assigned as follow: $p = 10$ and $u = 7$, 237 Thai Baths for the transactions in has-abroad account. The total cost of LOF and Isolation Forest models in Table 2 is 73456 Thai Bath and 3020 Bath, respectively.

6 Conclusion

The proposed transformation method attempt to characterize each transaction with the confident and risk values based on the derived normal behavior model. LOF and Isolation Forest are applied as the outlier detection in order to identify frauds. The experiment results shows that the proposed detection models were better for the ‘has-abroad’ account group than for the ‘local-only’ account group.

Acknowledgement. This work was supported by the SIIT Young Researcher Grant, under contract no. SIIT 2017-YRG-NK04.

References

1. ATM Fraud Law and Legal Definition. <https://definitions.uslegal.com/a/atm-fraud/>. Accessed 9 July 2018
2. Dorronsoro, J., Ginel, F., Sgnchez, C., Cruz, C.: Neural fraud detection in credit card operations. *IEEE Trans. Neural Netw.* **8**, 827–834 (1997)
3. Aleskerov, E., Freisleben, B., Rao, B.: CARDWATCH: a neural network based database mining system for credit card fraud detection. In: *Proceedings of Computational Intelligence for Financial Engineering (CIFEr)*, pp. 220–226 (1997)
4. Sherly, K., Nedunchezian, R.: BOAT adaptive credit card fraud detection system. In: *International Conference on Computational Intelligence and Computing Research*, pp. 1–7 (2010)
5. Gehrke, J., Ganti, V., Ramakrishnan, R., Loh, W.: BOAT–optimistic decision tree construction. In: *Proceedings of International Conference on Management of Data*, pp. 169–180 (1999)
6. Kirkos, E., Spathis, C., Manolopoulos, Y.: Data mining techniques for the detection of fraudulent financial statements. *Expert. Syst. Appl.* **32**, 995–1003 (2007)
7. Ngai, E., Hu, Y., Wong, Y., Chen, Y., Sun, X.: The application of data mining techniques in financial fraud detection: a classification framework and an academic review of literature. *Decis. Support. Syst.* **50**, 559–569 (2011)
8. Tao, G., Gui-Yang, L.: Neural data mining for credit card fraud detection. In: *2008 International Conference on Machine Learning and Cybernetics*, pp. 3630–3634 (2008)
9. Whitrow, C., Hand, D.J., Juszczak, P., Weston, D., Adams, N.M.: Transaction aggregation as a strategy for credit card fraud detection. *Data Min. Knowl. Discov.* **18**, 30–55 (2008)
10. Huang, D., Mu, D., Yang, L., Cai, X.: CoDetect: financial fraud detection with anomaly feature detection. *IEEE Access* **6**, 19161–19174 (2018)
11. Wu, S., Banzhaf, W.: Combatting financial fraud: a coevolutionary anomaly detection approach. In: *10th Annual Conference on Genetic and Evolutionary Computation*, pp. 1673–1680 (2008)
12. Yamanishi, K., Takeuchi, J., Williams, G.: On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. *Data Min. Knowl. Discov.* **8**, 275–300 (2004)
13. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection. *ACM Comput. Surv.* **41**, 1–58 (2009)

14. Breunig, M., Kriegel, H., Ng, R., Sander, J.: LOF: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, pp. 93–104 (2000)
15. Elahi, M., Li, K., Nisar, W., Lv, X., Wang, H.: Detection of local outlier over dynamic data streams using efficient partitioning method. In: World Congress on Computer Science and Information Engineering, pp. 76–81 (2009)
16. Liu, F., Ting, K., Zhou, Z.: Isolation-based anomaly detection. *ACM Trans. Knowl. Discov. Data* **6**, 1–39 (2012)
17. Pedregosa, F.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)



Machine Learning for Drugs Prescription

P. Silva^{1(✉)}, A. Rivolli^{2(✉)}, P. Rocha^{3(✉)}, F. Correia^{3(✉)}, and C. Soares^{1(✉)}

¹ School of Engineering, University of Porto, Porto, Portugal
oliveira.silva@fe.up.pt, csoares@fe.up.pt

² Federal University of Technology, Parana, Apucarana, Brazil
rivolli@utfpr.edu.br

³ Glintt - Healthcare Solutions, S.a., Porto, Portugal
procha@glintt.com, fcorreia@glintt.com

Abstract. In a medical appointment, patient information, including past exams, is analyzed in order to define a diagnosis. This process is prone to errors, since there may be many possible diagnoses. This analysis is very dependent on the experience of the doctor. Even with the correct diagnosis, prescribing medicines can be a problem, because there are multiple drugs for each disease and some may not be used due to allergies or high cost. Therefore, it would be helpful, if the doctors were able to use a system that, for each diagnosis, provided a list of the most suitable medicines. Our approach is to support the physician in this process. Rather than trying to predict the medicine, we aim to, given the available information, predict the set of the most likely drugs. The prescription problem may be solved as a Multi-Label classification problem since, for each diagnosis, multiple drugs may be prescribed at the same time. Due to its complexity, some simplifications were performed for the problem to be treatable. So, multiple approaches were done with different assumptions. The data supplied was also complex, with important problems in its quality, that led to a strong investment in data preparation, in particular, feature engineering. Overall, the results in each scenario are good with performances almost twice the baseline, especially using Binary Relevance as transformation approach.

Keywords: Machine learning · Classification problems · Multi-label classification · Health systems · Medicines prescriptions

1 Introduction

Each time a patient goes to a hospital, the doctor has to analyze the symptoms and define a diagnosis, and, if necessary, provide a set of medicines that will help. The definition process can be a difficult task for the doctor because every patient is unique and numerous diseases have similar symptoms. This can lead to errors and inaccurate diagnoses. The process of diagnosis identification is iterative. The doctors may need multiple appointments with the patients to make a decision. In each one, they analyze the patient's symptoms, their historical data and

previously conducted exams to create a list of the most suitable diseases [10] and for each one, the best drugs. In order to reduce the number of hypotheses, more exams may be required for the next appointment.

Considering the final diagnosis, the doctor has to decide the best drugs for the problem. Like with diagnosis definition, providing the correct medicines takes into consideration patient's medical history. This is an important factor because some people can not take several remedies due to allergies to them, such as Penicillin [7]. The specialist has to identify multiple drugs for the diagnosis, based on their medical experience, and, from then, decide which should be taken by the patient.

With Machine Learning approaches supporting the physicians, each patient could have better treatment and appointments would take less time and more patients would be treated. It may help the hospitals because it will reduce the number of patients that are waiting for an appointment. If this feature was implemented in the medical systems, it would decrease the costs associated with hospitalized patients because the doctors will have support in their decisions, releasing space to other patients.

2 Multi-label Classification

In data mining, there are several different tasks, in particular, regression and classification [14]. Both of them are based on the application of algorithms to a dataset in order to predict a target value. In classification, the target is a categorical value, also known as class or label. Whereas, in regression, the target is a numerical value.

One classification task is Multi-Label Classification (MLC), in which, instead of predicting a categorical value, each instance can be associated with more than one. In order to solve these problems a set of instances $X = \{x_1, x_2, \dots, x_n\}$ is necessary, where x_i is an object to be classified and n the number of instances and a set of labels $L = \{\lambda_1, \lambda_2, \dots, \lambda_l\}$, where l is the number of labels. With these variables, a training set can be $D = \{(x_i, Y_i); 1 \leq i \leq n\}$, where $x_i \in X$ and $Y_i \subseteq L$, with Y_i being the labels for the instance x_i . This set can be used to create a function $f : x_i \rightarrow 2^{Z_i}$, that for each unseen object $x_i \in X$ predicts the more suitable labels $Z_i \subseteq L$ [17]. Multi-Label classification is suitable when there are dependencies between the labels, like in text mining or music categorization [17].

2.1 Multi-label Classification Methods

There are two approaches in Multi-Label Classification: *problem transformation methods* and *algorithm adaptation methods*.

Problem transformation algorithms decompose Multi-Label problems in one or more single-label problems that are solved using single-label classifiers. From those methods, these were used: *Binary Relevance* (BR) [1], *Binary Relevance Plus* (BRPlus) [1] and *Ensemble of Classifier Chains* (ECC) [5].

Some of the single-label classifiers that are used with these approaches are *Java implementation of C4.5* (J48) [6], *Classification And Regression Trees* (CART) [6], *Support Vector Machines* (SVM) [11] and *Extreme Gradient Boosting* (XGB) [9].

Algorithm adaptation methods are created based on single label algorithms that have been changed to be able to output results with multiple labels. From those, *Multi-Label K-Nearest Neighbor* (ML-KNN) was used, that was an extension of the *K-Nearest Neighbor* [16].

2.2 Evaluation Metrics

For this paper, three metrics were used to evaluate the results: *F1-Score*, *Precision* and *Recall* [17]. *Precision* was used for measuring if the drugs predicted are correct. *Recall* to measure how many drugs are being predicted correctly. *F1* was also considered to evaluate the combination of the two previous metrics.

As baseline, it was used *General B*, that predicts the k most frequent labels, where k is the average number of labels in each instance, *Label Cardinality* (LC) [8]. With *General B*, it is possible to analyze how can the number of medicines used impact the results.

And, in order to validate the results, it was used the *Nemenyi post hoc test* [3]. This approach is used to compare all the algorithms pairwise, with different datasets. For each j algorithm in the i dataset, it is calculated the algorithm rank, r_i^j . Two algorithms have different performances if the difference between their average rank is greater than the critical difference [3].

2.3 Related Work

In the medical area, there are some projects using classification approaches taking as input demographic and medical history data in order to provide the most likely diagnosis. The most common diseases are cancer [12], diabetes [4] and heart problems [2,13]. However, no information was found regarding the use of classification techniques for drugs prescription, at least publicly.

3 Drug Prescription Using Data Mining

The data supplied came in 3 sets: *diagnoses*, *prescriptions* and *social-demographics* and had information from 2010 to 2017. All the information about the patients has been anonymized for confidentiality reasons.

An episode will have all the medical information during the time that the patient is being treated, in other words, an episode stores the progress of each patient during the treatment, such as diagnoses and prescriptions. In Fig. 1, part of an episode starting on 30th January 2018 is shown. This patient was diagnosed with two diseases on 1st February 2018 and the doctors prescribed some drugs in three different periods of time.



Fig. 1. Example of an episode.

In total, there are 11727 diagnoses and from those 7339 are hospitalizations, the type of episodes that were considered because it is the most critical area. Each episode can have multiple diagnoses at the same time, however, only one is the most important and requires the main focus. Each diagnosis has a unique code based on *International Statistical Classification of Diseases and Related Health Problems* (ICD) list [15]. The ICD code has a hierarchical structure with two or more levels.

The prescriptions dataset had 707666 instances with 420 unique drugs, so it was selected only the top 20 and top 40 most prescribed medicines, which corresponds to 51% and 68% of the total number of prescriptions. The five most prescribed medicines were *Paracetamol*, *Sodium Chloridium*, *Metoclopramide*, *Pantoprazole* and *Insulin*.

From the patients, the dataset had 13912 instances, where the average age was 52 years old with 45% being male.

The dataset did not have enough information about the historical records from the patients and the relationship between diagnoses and prescribed medicines was not established. So, it was performed some feature engineering and created attributes about past episodes, diagnoses and prescriptions per patient. In Table 1 are the 32 attributes used.

Diagnoses and prescriptions did not have a time relation because, in hospitalizations, it is not mandatory that it happens. For example, when a new diagnosis is defined, before prescribing the medicines, the doctor needs to analyze the previous medical history or if the current prescriptions are good enough. This process takes time, sometimes days to understand the impact of the actual treatment.

In order to overcome the problem with the missing association between diagnoses and treatments, several attempts were made but only one is worth mentioning because it was the closest to reality. It was considered the prescriptions made three days before the diagnosis date and until three days after. With this

Table 1. List of attributes used.

Attribute	Description
EPISODE	Episode
TOP_1_MED	Medicine most prescribed by patient
TOP_2_MED	Second medicine most prescribed by patient
TOP_3_MED	Third medicine most prescribed by patient
TOTAL_MEDS_PER_PAT	Total number of medicines prescribed for a patient
TOTAL_MEDS_DIFF_PER_PAT	Total number of different medicines prescribed for a patient
RATIO_MEDS_DIF_TOTALS	Ratio between the last two attributes
AVG_MEDS_PER_EP	Average number of medicines in each episode
AVG_MEDS_PER_PAT	Average number of medicines for each patient
NR_MEDS_PRESC	Number of medicines prescribed in an episode
TOTAL_EP_PER_PAT	Total number of patients episodes
TOTAL_DIAGS_PER_PAT	Total number of patients diagnoses
FIRST_EP	Date of the first patient episode
LAST_EP	Date of the last patient episode
START_DIAG	Initial date of the episode
FLG_PRIN	Important diagnosis identifier
DESCR_DIAG	Diagnosis description
COD_DIAG	Diagnosis code
FIR_LEVEL	First level of the diagnosis code
SEC_LEVEL	Second level of the diagnosis code
AVG_DIAGS_PER_EP	Average number of diagnoses in each episode
MIN_DIAGS_PER_EP	Minimum number of diagnoses in each episode
MAX_DIAGS_PER_EP	Maximum number of diagnoses in each episode
GENDER	Patient gender
BIRTH_YEAR	Patient birth year
BEST_MED_PRESC_PRIN	Most prescribed medicine in important diagnoses
SEC_BEST_MED_PRESC_PRIN	Second most prescribed medicine in important diagnoses
BEST_MED_PRESC_SECUN	Most prescribed medicine in non important diagnoses
SEC_BEST_MED_PRESC_SECUN	Second most prescribed medicine in non important diagnoses
VISIT_NR	Episode number

formulation, it is possible to cover the cases where the doctors only prescribed drugs some days after the diagnosis definition.

4 Experimental Setup

Diagnosis and prescription time is very important. A new prescription may be necessary due to the failure of the previous one, for example. Based on this, the dataset was ordered by the episode initial date. And then, split between 70% for training and 30% for testing. For the predictions was used F1, Precision and Recall.

As referred before, *Nemenyi post hoc test* was used as validation technique. However, since there is one dataset with information from eight years and this test requires multiple datasets, it was split by year resulting in eight datasets. The number of diagnoses in each year is not balanced, so there are datasets with more instances than others. In the end, each algorithm was tested 7 times.

For each algorithm, the first year dataset was used for training and the next one for testing. On the remaining iterations, the previous datasets were used for training and the next one for testing. The statistical results were evaluated using F1-Score because it combines Precision and Recall.

5 Results

Using the top 20 drugs, the final dataset has 1949 instances and Label Cardinality of 6. In Fig. 2, is the results got by each algorithm with F1, Precision and Recall.

From all the algorithms used, SVM and XGB were the best, almost doubling the baseline and outperforming all the others, especially ML-KNN and the decision tree algorithms (J48 and C4.5), when they were not used with ECC.

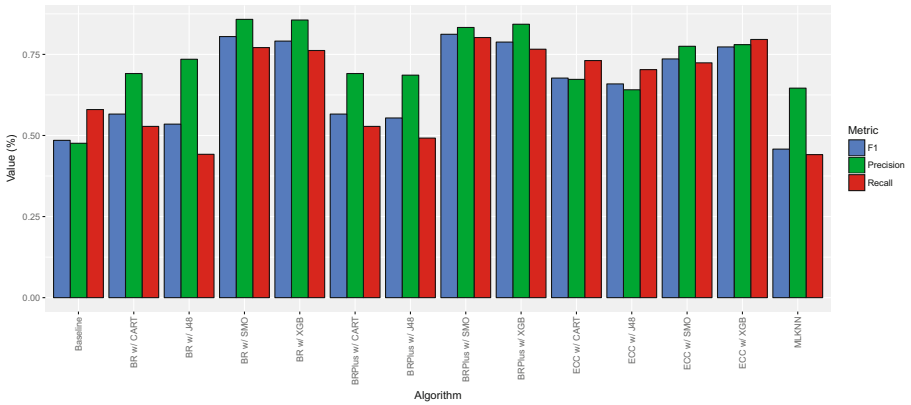


Fig. 2. Results using 20 medicines.

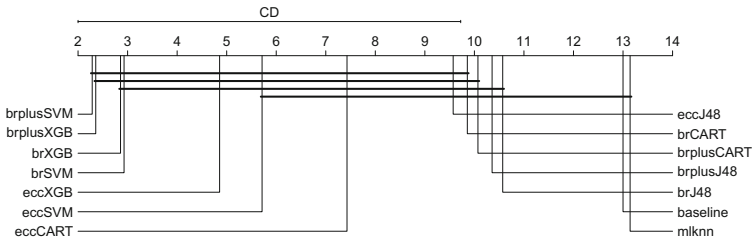


Fig. 3. Statistical results using 20 medicines with F1.

In Fig. 3, we can see a comparison of the algorithms results across the eight datasets created. The best ones are those with lower rank value, in this case, SVM and XGB using BR and BRPlus with rank values around 2.

With 40 drugs, the dataset size increased to 1964 with $LC = 7$. Just like in the previous scenario, but with around less 10% points, SVM and XGB outperformed the baseline and ECC was the best approach for J48 and CART because with BR and BRPlus, they were close to the baseline.

6 Conclusions and Future Work

In this work, the goal was to develop a system that would give drugs suggestions to doctors based on diagnoses. The development started with the analysis of the problem and the data that would be used. The dataset was created using information from diagnoses, prescriptions and social-demographics. Since it did not have enough data about the medical history for each patient, new attributes were created. The number of drugs was very large, so it was selected the top 20 and top 40 most prescribed drugs.

It was considered only the prescriptions made three days before the diagnosis date and until three days after due to the problem complexity. Considering all the work developed and the problem, the results obtained were good, many of algorithms had performances higher than the baseline and the difference was considerable. This may be an evidence that MLC techniques can also be used in drugs prescription problems.

For better results, may be necessary a new data sample with less noise and better structure. The development of a grid search for tuning algorithm parameters may be also a good option.

Acknowledgements. I would like to thank Glintt - Healthcare Solutions S.a., who provided data and expertise that greatly assisted the development of this work.

References

1. Alvares-Cherman, E., Metz, J., Monard, M.C.: Incorporating label dependency into the binary relevance framework for multi-label classification. *Expert Syst. Appl.* **39**(2), 1647–1655 (2012). <https://doi.org/10.1016/j.eswa.2011.06.056>
2. Dai, W., Brisimi, T.S., Adams, W.G., Mela, T., Saligrama, V., Paschalidis, I.C.: Prediction of hospitalization due to heart diseases by supervised learning methods. *Int. J. Med. Inform.* **84**(3), 189–197 (2015). <https://doi.org/10.1016/j.ijmedinf.2014.10.002>
3. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **7**, 1–30 (2006)
4. Fatima, M., Pasha, M.: Survey of machine learning algorithms for disease diagnostic. *J. Intell. Learn. Syst. Appl.* **09**(01), 1–16 (2017). <https://doi.org/10.4236/jilsa.2017.91001>
5. Gibaja, E., Ventura, S.: A tutorial on multilabel learning. *ACM Comput. Surv.* **47**(3), 1–38 (2015). <https://doi.org/10.1145/2716262>
6. Hssina, B., Merbouha, A., Ezzikouri, H., Erritali, M.: A comparative study of decision tree ID3 and C4.5. *Int. J. Adv. Comput. Sci. Appl. (IJACSA)* **4**(2) (2014). <https://doi.org/10.14569/SpecialIssue.2014.040203>. Special Issue on Advances in Vehicular Ad Hoc Networking and Applications

7. Levine, B.B.: Immunologic mechanisms of penicillin allergy. *N. Engl. J. Med.* **275**(20), 1115–1125 (1966). <https://doi.org/10.1056/NEJM196611172752009>
8. Metz, J., de Abreu, L.F.D., Cherman, E.A., Monard, M.C.: On the estimation of predictive evaluation measure baselines for multi-label learning. In: Pavón, J., Duque-Méndez, N.D., Fuentes-Fernández, R. (eds.) *IBERAMIA 2012. LNCS (LNAI)*, vol. 7637, pp. 189–198. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34654-5_20
9. Mitchell, R., Frank, E.: Accelerating the XGBoost algorithm using GPU computing. *PeerJ Comput. Sci.* **3**, e127 (2017). <https://doi.org/10.7717/peerj-cs.127>
10. Norman, G., Barraclough, K., Dolovich, L., Price, D.: Iterative diagnosis. *BMJ* **339**(7339), b3490 (2009). <https://doi.org/10.1136/bmj.b3490>. (Clinical research ed.)
11. Platt, J.: Sequential minimal optimization: a fast algorithm for training support vector machines. Technical report, Microsoft (1998)
12. Sontakke, S., Lohokare, J., Dani, R.: Diagnosis of liver diseases using machine learning. In: 2017 International Conference on Emerging Trends Innovation in ICT (ICEI), pp. 129–133 (2017). <https://doi.org/10.1109/ETHICT.2017.7977023>
13. Tantimongcolwat, T., Naenna, T., Isarankura-Na-Ayudhya, C., Embrechts, M.J., Prachayasittikul, V.: Identification of ischemic heart disease via machine learning analysis on magnetocardiograms. *Comput. Biol. Med.* **38**(7), 817–825 (2008). <https://doi.org/10.1016/j.combiomed.2008.04.009>
14. Tomar, D., Agarwal, S.: A survey on data mining approaches for healthcare. *Int. J. Bio-Sci. Bio-Technol.* **5**(5), 241–266 (2013). <https://doi.org/10.14257/ijbsbt.2013.5.5.25>
15. Topaz, M., Shafran-Topaz, L., Bowles, K.H.: ICD-9 to ICD-10: evolution, revolution, and current debates in the United States. *Perspect. Health Inf. Manag.* **10**(Spring), 1d (2013)
16. Zhang, M.L., Zhou, Z.H.: ML-KNN: a lazy learning approach to multi-label learning. *Pattern Recognit.* **40**(7), 2038–2048 (2007). <https://doi.org/10.1016/j.patcog.2006.12.019>
17. Zhang, M.L., Zhou, Z.H.: A review on multi-label learning algorithms (2014). <https://doi.org/10.1109/TKDE.2013.39>



Intrusion Detection Using Transfer Learning in Machine Learning Classifiers Between Non-cloud and Cloud Datasets

Roja Ahmadi^(✉), Robert D. Macredie^(✉), and Allan Tucker^(✉)

Brunel University London, Uxbridge, UK

{Roja.AhmadiShirvani, Robert.Macredie, Allan.Tucker}@brunel.ac.uk

Abstract. One of the critical issues in developing intrusion detection systems (IDS) in cloud-computing environments is the lack of publicly available cloud intrusion detection datasets, which hinders research into IDS in this area. There are, however, many non-cloud intrusion detection datasets. This paper seeks to leverage one of the well-established non-cloud datasets and analyze it in relation to one of the few available cloud datasets to develop a detection model using a machine learning technique. A complication is that these datasets often have different structures, contain different features and contain different, though overlapping, types of attack. The aim of this paper is to explore whether a simple machine learning classifier containing a small common feature set trained using a non-cloud dataset that has a packet-based structure can be usefully applied to detect specific attacks in the cloud dataset, which contains time-based traffic. Through this, the differences and similarities between attacks in the cloud and non-cloud datasets are analyzed and suggestions for future work are presented.

Keywords: Cloud intrusion detection model · Transfer learning
Machine learning · DDoS attack

1 Introduction

The Distributed Denial-of-Service (DDoS) attack type is one of the most significant security issues and powerful threats to cloud computing. This type of attack aims to saturate the network with a huge volume of unwanted traffic to prevent users from accessing services and applications. In a cloud-computing environment, because of the distributed characteristics of the cloud and its nature as a shared-resource platform, the impact of DDoS attacks is greater than in traditional network infrastructures [1]. As such, DDoS typically takes advantage of the characteristics of cloud computing in order to launch attacks and this can cause significant issues. For example, in February 2018 a DDoS attack caused a service outage on GitHub of 126.9 million packets per seconds [2].

In an attempt to address such attacks, IDS have been implemented in cloud environments to provide well-defined security mechanisms to monitor and analyze the behavior of users and network activities with the aim of detecting possible intrusions. Where a potential intrusion is identified, the IDS automatically alerts the administrator

so that they make take further action. As such, IDS systems need to develop over time to cope with changes in the attacks.

When undertaking research in this area, there are though, several issues. First, there is a lack of public cloud datasets, which hinders the development of models to detect attacks in cloud environments [3]. One solution is to try to use non-cloud and cloud datasets together, with the non-cloud data being used to develop the intrusion detection model which is then applied to the cloud dataset. Second, assuming the use of non-cloud and cloud datasets as suggested, there are limited common features between the datasets to use as classifiers in the model. Third, datasets are often structured differently, with some presenting packet based data and others data that is time-based. Finally, datasets may focus on different attack types. All of these issues make using existing datasets challenging.

This paper will explore an approach to developing a model using limited overlapping features that are common in a non-cloud and cloud dataset. Section 2 will present the non-cloud dataset (NSL-KDD), cloud dataset and feature selection process. Section 3 will explain the approach taken to overcome the difference in the structure of two datasets and each stage of developing the model, reporting the use of machine learning techniques to train the model with the non-cloud dataset to see its ability in the classification of attacks by using the application of cross-validation and sensitivity analysis. Section 4 will then report the results of testing the model with the cloud dataset with different attack types; this will use a novel application of a confusion matrix to explore the similarities and differences in the attack types through a sensitivity analysis and will then analyze the results.

2 Datasets and Feature Extraction

2.1 NSL-KDD Dataset

The NSL-KDD dataset was selected as the training dataset. The NSL-KDD dataset is a refined version of the Knowledge Discovery of Database (KDD) dataset that is a standard benchmark dataset for network intrusion detection system evaluation [4]. The NSL-KDD dataset contains instances of network traffic that include 41 features and one class attribute, which indicates the type of the network connection traffic (either the attack type or that the data instance represents normal traffic). These 41 features are classified into three groups of features. The first is the group of basic features related to the attributes of the observed network packet headers. The second is the group of the features associated with the content of the packet. Finally, the third is the group of network traffic features, statistical information representing all connections to the same destination machine within a two-second timeframe (This group does not have the same structure as the cloud dataset, and cannot be used for comparison purposes). The study reported in this paper focused on the basic feature group since this is where information relevant to DDoS attacks might be identified. There are four different groups of attack in the dataset; as the focus of this research is on DoS attacks, we selected only the group of records where DoS attacks were signified. The DoS group comprises a seven attack types: Back; Land; Neptune; Ping of Death; Smurf; and Teardrop.

2.2 Cloud Intrusion Detection Dataset (CIDD)

Only two public cloud datasets were identified. The first contained data related to masquerade attacks, DoS attacks, U2R attacks and R2L attacks, but the dataset did not contain network packet features. Instead, the dataset included audit parameters that were not compatible with the NSL-KDD dataset selected for the training activity. The second dataset, the CIDD, contained the same attack types as the NSL-KDD dataset and had been developed to focus on important features in detecting DDoS attacks. Therefore, it was decided to select this dataset for the testing phase. The CIDD dataset was generated in 2016 from testbed experimental work on cloud DDoS attack detection [5]. The simulated cloud environment that generated the data consisted of six nodes each of which included six virtual machines (VMs). In the scenario of the research that generated the dataset, each VM was allocated a specific role. One VM hosted the cloud server and this was considering the target machine for the attack(s). Some of the VMs were equipped with DDoS attack tools for launching different types of DDoS attack. The rest of the VMs were designed to have a normal cloud user role, such as requesting webpages from the cloud server. A connection was initiated by sending a request from a user (the user of a VM might be an attacker or a normal user) to the servers. The generated traffic was recorded by the sniffer, a network packet analyzer in the form of software or hardware used to capture the TCP/IP packets over a network. The subsequent outcome of captured traffic included 5274 instances of network TCP/IP connections. Of these instances, 682 records were different types of DDoS attack: ICMP flood; Ping-of-Death; UDP flood; TCP SYN flood; TCP LAND; DNS flood; and Slowloris. A custom-built sniffer was used to extract the desired features from the IP packet header fields. Six distinct features were calculated per incoming IP address for each type of transport layer protocol (TCP, UDP, ICMP and other types), resulting in 24 features in total. The extracted data was processed in five-second time intervals to create a time-based, rather than packet-based, dataset.

2.3 Overlap Features Between Two Datasets

The two chosen datasets are structured in different ways: the selected group of features from the NSL-KDD dataset is packet-based whereas the entire CIDD dataset comprises time-based traffic, aggregated into records covering five-second intervals. This is one reason why the datasets have a restricted set of overlapping features, making using them together challenging (and potentially limiting the value of such an approach). As already noted, owing to the structure of the proposed model (which will be explained in the next section), the training set and testing set had to have similar features. Therefore, to be able to train the model with the NSL-KDD dataset and then test it with the CIDD dataset we selected features that were common between the two datasets, using them to form the initial, small feature set of the model. As a result, the initial model was developed based on the three features from the NSL-KDD dataset that were also contained in the CIDD dataset: protocol type; source bytes (*src_bytes*); and destination bytes (*dst_bytes*).

Protocol type identifies the type of connection to which the network traffic belongs. It also refers to the associated type of protocol of each layer of the network model, such

as UDP, TCP, ICMP, etc. The `src_bytes` feature of the dataset contains the number of bytes transmitted from the sender computer (source) to the destination computer. The `dst_bytes` feature contains the number of bytes sent from the destination computer to the source computer.

In the CIDD dataset, the `Bytes_In` and `Bytes_Out` features for TCP, UDP and ICMP protocols (therefore also giving the protocol type feature) were selected to subsequently test the model with the CIDD dataset. These features are similar to the three selected from the NSL-KDD dataset. `Bytes_In` and `Bytes_Out` refer to the bytes received from the source to destination, and from destination to source, respectively. As such they correspond to the `src_bytes` and `dst_bytes` features. The pre-processing that was undertaken to ‘unpack’ the time-based data into a form that was closer in structure to (but still not the same as) the non-cloud dataset will be discussed in the next section.

3 Developing the Model

This section reports the techniques that were applied to the datasets to prepare them for the processing task (i.e., it reports the data pre-processing). It then describes the machine learning approach, algorithm and tools that were used to develop the detection model. Finally, it presents the initial results of training the model with the NSL-KDD dataset and evaluates the results based on cross-validation and sensitivity analysis.

3.1 Discretization and Resampling

Discretization techniques were applied to both datasets to ensure that any continuous values were translated in suitable discrete values. The Weka [6] tool was used to undertake this discretization activity. The discretization filter was applied to the ‘`src_bytes`’ and ‘`dst_bytes`’ features of the two datasets by ‘binning’; that is, the range of numerical values for each feature were divided into three bins (or intervals) of equal size.

There were imbalances in the distribution of the data across the eight classes in the NSL-KDD (training) dataset which could have caused lower accuracy in the subsequently-developed model’s prediction of classes. To address this issue, resampling was undertaken in Weka. This resulted in over-sampling the minority classes and under-sampling the majority classes to create a more balanced distribution in the dataset.

3.2 ‘Unpacking’ the Time from the CIDD Dataset

Figure 1 presents an extract of the CIDD dataset to show its initial structure. As well as the attack type or normal traffic identifier, each row/instance of the CIDD dataset includes 5-s of aggregated packet data organised under four types of protocol: TCP; ICMP; UDP; and others. Data related to the ‘other’ protocol category were not used because the types of protocol were not specified.

A	B	C	D	E	F	G	H
Bytes_InTCP	Bytes_InUDP	Bytes_InICMP	Bytes_InOthers	Bytes_Out_UDP	Bytes_Out_ICMP	Bytes_OutOthers	Attack type
437	517	1	1	3157	1	1	Normal
1	592	1	1	3143	1	1	Normal
438	1069	1	1	6508	1	1	Normal
1	247	197481408	120225936	2560	1	1	UDP flood
2344	1134	1	1	6311	1	1	Normal
926	598	1	1	3612	1	1	Normal
4785	7918	1	1	2343	1	1	Slowlorries
1364	1167	1	1	7635	1	1	Normal
874	912	1	1	5254	1	1	Normal
438	635	1	1	3984	1	1	Normal
2343	1441	1	1	15774	1	1	Normal
6752	2148	1	1	558	1	1	Slowlorries
60	553	1	1	3155	1	1	Normal
433	740	1	1	4368	1	1	Normal
71583960	12191798	1	1	187663	1	1	TCP SYN
438	890	1	1	5236	1	1	Normal

Fig. 1. The initial structure of an extract of the CIDD dataset

To make the structure of the CIDD dataset closer to that of the NSL-KDD dataset, each record was divided to create three records, one for each protocol type considered (TCP; ICMP; and UDP). Figure 2 presents an extract that shows the ‘unpacked’ form of the CIDD dataset for a set of TCP records.

protocol_type	src_bytes	dst_bytes	'Attack types
TCP	B	A	Normal
TCP	A	C	Normal
TCP	A	A	Normal
TCP	A	C	Back
TCP	B	A	Normal
TCP	B	A	Normal
TCP	B	A	mailbomb
TCP	B	B	Normal
TCP	B	B	Normal
TCP	A	A	Normal
TCP	B	B	Normal

Fig. 2. The unpacking the time structure of an extract of the CIDD dataset.

As can be seen in Fig. 2, the resulting dataset has four columns which are similar to those extracted from the NSL-KDD dataset. At this stage, the CIDD pre-processed dataset was ready for use in testing the model once developed. The next sub-section reports the processes that were involved in developing the model.

3.3 Machine Learning Classification Algorithm

With the aim of developing the most accurate model possible given the constraints of the datasets, we applied different machine learning classifiers to the NSL-KDD dataset to assess how accurate each algorithm was in terms of attack prediction. In addition to the accuracy factor, speed was another important issue used to select the classification algorithm owing to the high dimensionality of both datasets. Based on analysis of relevant literature, the following classification algorithms were identified as being the most widely-used to solve the classification problem in IDS: decision tree; neural network; and Naïve Bayes. Of these algorithms, the Naïve Bayes classification algorithm was more accurate than the other two. (It also performed marginally faster but this was not

a key issue in the model development phase). Therefore, it was decided to use this algorithm as the classifier.

Naïve Bayes classification is a machine-learning algorithm and probabilistic classifier that has proved to be effective, fast and accurate for a range of real-world scenarios [7]. The Naïve Bayes model also has the advantage of being easy and simple to build. The Naïve Bayes classifier predicts the product classes (in this case the different types of DDoS attack) based on the independence hypothesis, which means that the presence of a specific feature in a class (in our case the protocol type, `src_bytes` and `dst_bytes`) is not dependent on the presence of the other features. Naïve Bayes uses Bayes theorem, which describes conditional probabilities – the likelihood of an event based on the relevant prior knowledge of that event. For example, in the context of our model, it is concerned with the likelihood of a specific attack occurring based on the values of the existing three features in the dataset.

3.4 Machine Learning Implementation

This study used Weka version 3.8 [6] to implement the chosen classifier and filter. Most of the available tools offer very similar functionality; Weka is, though, free and widely used, making it an appropriate choice for this work. Having selected the Weka environment, the NSL-KDD dataset was used to train the model and then test it with the testing dataset (CIDD). First, we wanted to know how well the classifier predicted the different types of DDoS attack identified in the non-cloud dataset to understand the level of accuracy of the model. The evaluation of the performance of the classifier will be reported in the next sub-section.

3.5 Evaluation of the Classifier

Cross validation and sensitivity analysis were used to evaluate the results of the classifier/resulting model. First, a 10-fold cross validation package was applied to the data. This software package divided the NSL-KDD dataset into 10 groups, of which nine groups were selected for training automatically by the Weka package and the remaining group was used to test the model. A confusion matrix was then used to present a summary of the prediction outcomes of the classification model, allowing sensitivity analysis. The generated matrix shows the number of correctly and incorrectly predicted instances of each class, providing insights into the errors being made by the classification model. Figure 3 presents the confusion matrix resulting from the classification model when applied to the NSL-KDD dataset.

The instances on the main diagonal (top left to bottom right) indicate the correct prediction (i.e., where each attack type from the dataset is correctly predicted by the model). If these numbers are high for all rows/attack types, it demonstrates a good classification outcome. In Fig. 3 most of the attack types are classified correctly. Of these attack types Land, Pod and some of the normal cases have been classified incorrectly.

The general accuracy of the classifier is 89.103%, suggesting high overall accuracy with the noted exceptions. The next section will report the results of applying the model

to the CIDD dataset and explain the way in which the confusion matrix approach was used in this part of the study in relation to each cloud attack type.

=== Confusion Matrix ===									
	a	b	c	d	e	f	g	h	<-- classified as
129	11	40	4	1	0	0	0	45	a = Normal
0	230	0	0	0	0	0	0	0	b = TCP SYN
1	0	195	0	0	0	0	0	0	c = Back
0	0	0	230	0	0	0	0	0	d = Smurf
0	0	0	0	188	0	0	0	0	e = Teardrop
0	0	0	38	0	0	0	0	0	f = Pod
0	7	0	0	0	0	0	0	0	g = land
0	0	0	0	0	0	0	230	0	h = mailbomb

Fig. 3. Confusion matrix showing the result of the classification applied to the NSL-KDD dataset.

4 Analysis of Results and Discussion

This section will report the results of applying the classification model that was trained on the non-cloud data (NSL-KDD) to the cloud data (CIDD). The predicted outcomes of the classification model will be then analyzed using a confusion matrix, but, in this instance, it will be used in a different/non-standard way. The standard confusion matrix describes the results of applying a developed classification model on a testing dataset that contains the same classifiers as the training set. For example, having trained our model with the non-cloud data, it would normally be tested with another non-cloud dataset containing the same features, with the aim of the classification model predicting the same type of attacks in this new dataset. However, in this work, the types of DDoS attack in the two datasets are not the same. The Naïve Bayes classification model was trained with, and tested on, the non-cloud data, which contained identifiers for Normal traffic and the following seven attack types: TCP SYN; Back; Smurf; Teardrop; Ping of Death (POD); Land. The cloud data contained identifiers for Normal traffic and the following seven attack types: TCP SYN; UDP Flood; ICMP Flood; DNS Flood; Ping of Death (POD); Land; and Slowloris.

Testing the model on the CIDD dataset leads to the creation of a confusion matrix that shows how the model has classified the Normal and attack type instances in the cloud data in terms of Normal traffic and the non-cloud attack types. In this way, the matrix can be used to identify where there is a set of cloud data instances of one attack type (for example, DNS Flood) classified as a particular non-cloud attack type (for example, TCP SYN). The values of the features/classifiers in the model for these (in our example, DNS Flood) cloud data instances can then be compared with the feature data for all of the instances of the identified/classified non-cloud attack (in our example, TCP SYN) in the original test data (NSL-KDD). This may help us to understand the correspondence between cloud and non-cloud attack types and subsequently identify areas in which the model would have to be developed to more effectively identify and classify cloud attacks.

A threshold was set to focus the analysis, with only the cases where there were above 30 or more instances of a cloud attack having been identified as a particular non-cloud

attack being explored. This threshold was determined by simple observation of the values in the matrix. These cloud data instances were then considered by extracting and analyzing the distributions of the (discretized) values of the three core features and comparing them to the comparable features from the relevant non-cloud attack instances (from the NSL-KDD dataset).

4.1 Transfer Learning: Applying the Classification Model to the CIDD Dataset

Transfer learning is the application of a model that was trained for solving one particular task is then re-used on another different task [8, 9]. Here we apply a model trained on non-cloud data to classify intrusions (of different types) on cloud data. Figure 4 presents the confusion matrix arising from applying the model to the CIDD dataset.

=== Confusion Matrix ===									
Normal	TCP SYN	Back	Smurf	Teardrop	Pod	land	mallbomb	←- classified as	
11653	77	0	0	0	0	0	0		a = Normal
156	0	0	0	0	0	0	0		b = TCP SYN
36	34	0	35	0	0	0	0		c = UDPflood
30	15	0	0	0	0	0	0		d = ICMPflood
35	35	0	35	0	0	0	0		e = DNSflood
24	12	0	0	0	0	0	0		f = Pod
69	0	0	0	0	0	0	0		g = land
1314	0	0	0	0	0	0	0		h = Slowloris

Fig. 4. Results of classification based on testing with CIDD dataset

As Fig. 4 shows, in the first row, normal instances were classified by the model correctly 11653 times, but 77 instances were classified as TCP SYN attacks. In the second row, all of the cloud TCP SYN instances were classified as normal traffic instances. In the third row, the UDP flood attack instances were classified as normal instance 36 times, as TCP SYN attacks 34 times, and as Smurf attacks 35 times. In the fourth row, the ICMP flood attack instances were classified as normal instances 30 times. In the fifth row, 35 DNS flood attack instances were classified as being normal traffic instances, 35 were classified as being TCP SYN attacks and 35 were classified as being Smurf attacks. In the last two rows of the matrix, all of the instances of Land attack and Slowloris attack were classified as Normal traffic (69 times and 1314 times, respectively). The next section will analyze the cases presented above.

4.2 Analysis of the Results of Applying the Classification Model to the CIDD Dataset

To understand the classifications made by the model, the correspondences and the differences between cloud attack types and non-cloud attack types were explored by analyzing the underlying data of the cases identified in the confusion matrix that met the set threshold. These ‘above threshold’ instances of Normal traffic and then each

attack type in the cloud dataset that were classified by the model as specific non-cloud attack types, or as Normal traffic, were isolated and examined. Table 1 represents those cloud cases in Fig. 4 that are above the threshold (as such, the Pod and ICMP Flood attack types are excluded).

Table 1. Analysis of the results of the ‘above threshold’ cases

Actual cloud attack and normal	Classified attack type and normal			The factor that the model suggests									
	Normal	TCP SYN	Smurf	Protocol_types			Src_bytes			Dst_bytes			
				TCP	UDP	ICMP	L	M	H	L	M	H	
Normal		✓		✓			✓						
TCP SYN	✓			✓	✓	✓	✓			✓	✓		✓
UDP flood	✓			✓	✓			✓			✓	✓	✓
UDP flood		✓		✓			✓						
UDP flood			✓			✓							
DNS flood	✓				✓			✓	✓			✓	✓
DNS flood		✓		✓			✓						
DNS flood			✓			✓							
Land	✓			✓	✓	✓	✓			✓	✓		✓
Slowloris	✓			✓	✓	✓	✓	✓	✓	✓	✓		✓

Each row in Table 1 reports cloud cases that are classified as other types of attack. The classification model classifies these cases on the basis of the protocol type (TCP, UDP and ICMP), src_bytes and dst_bytes (L, M and H indicate the discretized values). The first row of Table 1 represents Normal cases that are being classified as TCP SYN attacks. The rest of the table represents TCP SYN, UDP Flood, DNS Flood, Land and Slowloris cases that are being classified as a Normal, TCY SYN or Smurf attacks. For the UDP Flood and DNS flood types, three rows for each are presented as they are being classified as more than one non-cloud type; (Normal, TCP SYN and Smurf). The table then shows the classifier values on which analysis of the underlying data suggests the model may be classifying in each case.

As can be seen in Fig. 4, all of the cloud cases are classified as Normal traffic, TCP SYN and Smurf attacks. From the cloud data, most of the instances of TCP SYN, UDP flood, DNS flood, Land and Slowloris have been classified as Normal traffic (see rows 3, 6, 9 and 10 of Table 1). This is because there is the whole range of protocol types, and values for src_bytes and dst_bytes in these instances and there are no other features through which the model can discern them as the other types of attack. The classification as Normal traffic in these cases is likely to be because in the non-cloud dataset on which the model was trained, there are many ‘Normal’ packet-based cases with different combinations of values for the features, making it the ‘default choice’ for the model.

There are also instances of CIDD Normal traffic, and UDP flood and DNS flood attacks that are being classified as TCP SYN attacks on the basis of the TCP protocol type and a low discretized value for the src_bytes classifier. Finally, it seems that some instances of UDP flood and DNS flood in the cloud data are being classified as Smurf attacks on the basis of the ICMP protocol only.

As Fig. 3 shows, the outcome of classification model when applied to the NSL-KDD dataset is quite accurate even with the small feature set. Although there are cases such

as Land, Pod, and some of the normal cases that have not been classified correctly, the overall performance of the model is good on the non-cloud dataset.

When the model is applied to the cloud dataset, its performance is markedly different, as can be seen in Fig. 4. For example, the TCP SYN, Pod and ICMP flood (another name for Smurf) attack types are common to both datasets which should lead to them being classified as same type of attack by the model. However, the model mis-classifies every instance of these three attacks in the CIDD dataset.

This problem seems to arise from the structure of the cloud dataset where the ‘unpacking time’ issue, explained in Sect. 3.2, has influenced the structure of the cloud dataset. The structure of the cloud dataset means that each instance/row in the cloud dataset is very likely to have multiple protocol types (since packets are being aggregated into a single data instance/row) whereas each instance in the NSL-KDD dataset includes a single protocol type as it represents a single packet. As result, TCP SYN cases in the cloud data are very likely to have multiple protocol types, and that is why they have been classified as Normal cases. This is also true for UDP flood, DNS flood, Land and Slowloris. Moreover, where the UDP flood cases include TCP protocol packets they have been classified as TCP SYN attacks, and where they include ICMP protocol packets they have been classified as Smurf attacks. This also happens for DNS flood cases: where DNS cases include TCP protocol packets they have been classified as TCP SYN attacks, and where they include ICMP protocol packets they have been classified as Smurf attacks.

In addition to the structural issue of the datasets, the limited number of overlapping features causes the model to classify some instances of UDP flood as Normal, TCP SYN or Smurf (see Fig. 4) because there are no other features than can be discerned by the model to classify the instances as the UDP attack type. This is also true for DNS flood cases that have been classified as Normal, TCP SYN and Smurf.

5 Conclusions

This research aimed to leverage one of the well-established non-cloud datasets, and analyze it in relation to one of the few available cloud datasets to develop a detection model. This was explored through a combination of a machine learning classifier and transfer learning to remap the intrusion types. However, the results of the analysis show that, while the model performed well on the non-cloud dataset, it is of limited value in classifying attacks in the CIDD dataset owing to the different structures of the two datasets, the small overlapping feature set and the different attack types. Given that there are very few publicly-available cloud intrusion detection datasets, there is a need for generating new datasets of this type that have a common structure, address a standard set of attack types (which can be expanded as new attack types emerge) and include a wider range of features to be able to use them to compare results with those of existing work in the area and allow research to validate their work/findings on other comparable datasets. This will be the focus of the next stage of our research work.

References

1. Somani, G., Gaur, M.S., Sanghi, D., Conti, M., Buyya, R.: DDoS attacks in cloud computing: issues, taxonomy, and future directions. *Comput. Commun.* **107**, 30–48 (2017)
2. Sam, S.: February 28th DDoS incident report (2018). <https://githubengineering.com/ddos-incident-report/>. Accessed 17 June 2018
3. Kholidy, H.A., Baiardi, F.: CIDD: a cloud intrusion detection dataset for cloud computing and masquerade attacks. In: *Proceedings of the 9th International Conference on Information Technology, ITNG 2012*, pp. 397–402, April 2012
4. Shiravi, A., Shiravi, H., Tavallae, M., Ghorbani, A.A.: Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Comput. Secur.* **31**(3), 357–374 (2011)
5. Kumar, R., Lal, S.P., Sharma, A.: Detecting denial of service attacks in the cloud. In: *Proceedings - 2016 IEEE 14th International Conference on Dependable, Autonomic and Secure Computing, DASC 2016, 2016 IEEE 14th International Conference on Pervasive Intelligence and Computing, PICom 2016, 2016 IEEE 2nd International Conference on Big Data*, pp. 309–316 (2016)
6. Frank, E., Hall, M.A., Witten, I.H.: *The WEKA Workbench* (2016)
7. Ahmed, M., Naser Mahmood, A., Hu, J.: A survey of network anomaly detection techniques. *J. Netw. Comput. Appl.* **60**, 19–31 (2016)
8. Dai, W., Yang, Q., Xue, G.-R., Yu, Y.: Boosting for transfer learning. In: *Proceedings of the 24th International Conference on Machine learning - ICML 2007*, pp. 193–200 (2007)
9. Lazaric, A., Restelli, M., Bonarini, A.: Transfer of samples in batch reinforcement learning. In: *Proceedings of the 25th International Conference on Machine learning - ICML 2008*, pp. 544–551 (2008)



Concatenating or Averaging? Hybrid Sentences Representations for Sentiment Analysis

Carlotta Orsenigo[✉], Carlo Vercellis[✉], and Claudia Volpetti[✉]

Department of Management, Economics and Industrial Engineering,
Politecnico di Milano, Via Lambruschini 4b, 20156 Milan, Italy
{carlotta.orsenigo,carlo.vercellis,claudia.volpetti}@polimi.it

Abstract. Performances in sentiment analysis - the crucial task of automatically classifying the huge amount of users' opinions generated online - heavily rely on the representation used to transform words or sentences into numbers. In the field of machine learning for sentiment analysis the most common embedding is the bag of words (BOW) model, which works well in practice but which is essentially a lexical conversion. Another well-known method is the Word2vec approach which, instead, attempts to capture the meaning of the terms. Given the complementarity of the information encoded in the two models, the knowledge offered by Word2vec can be helpful to enrich the information comprised in the BOW scheme. Based on this assumption we designed and tested four hybrid sentence representations which combine the two former approaches. Experiments performed on publicly available datasets confirm the effectiveness of the hybrid embeddings which led to a stable increase in the performances across different sentiment analysis domains.

Keywords: Text classification · Sentiment analysis
Machine learning · Word vectors · Word2vec · Bag of words
Hybrid sentence representation

1 Introduction

With the rapid growth of opinionated texts produced daily by online users, the ability of classifying opinions has become imperative to understand political orientations [18], brand perception [5] or even to forecast the impact of news on financial markets [2]. Consequently, several research efforts in the machine learning domain have been recently devoted to design accurate text classification algorithms for sentiment analysis, in order to automatically assign a sentiment polarity to user-generated comments [9, 16].

To apply machine learning, sentences must be converted into a numeric format through a vector-based representation. Vectors can be derived directly from the raw text by means of several strategies which reflect different lexical, syntactical or semantic properties of the documents. Choosing a specific text

representation is crucial since it determines the information provided in input to the classifier and, therefore, considerably affects its performances [7].

For text classification the most common feature extraction method is the bag of words (BOW) model, in which each document is described in terms of a vocabulary of words built on a given training corpus. In the resulting dataset the numeric features indicate the presence or, alternatively, the frequency of each term in the document, where the former strategy has been shown to perform better compared to the latter for the purpose of sentiment classification [15]. Notice that, the bag of words approach discards any positional information about the terms in the sentences and relies only on the presence of a word or on how frequently the term occurs. Despite the effectiveness observed in practice, it is indeed an exclusively lexical transformation in which the word order is disrupted and syntactic structures are broken.

As an alternative to the BOW sparse vectors, some distributed representations of words as real-valued vectors, called word vectors, have been proposed to capture their semantic aspects [1]. Among these, the Word2vec model developed by [12] generates numeric vectors using neural networks, with the aim of grasping the meaning of each word considering its relations with other terms in the same context. [12] proposes two methods to learn these vectors from text data. The first is the Skip-Gram model which learns the word vectors by training a neural network to predict the surrounding context words given a central word. The second, called continuous bag of words, learns these vectors by predicting the central word given its context of surrounding terms in a fixed window. In both cases, the size of the window of neighboring terms is a parameter of the model. Since the construction of such representations is unsupervised and generally involves huge corpora of documents, such as Wikipedia [3] or Google News [12], a common strategy is to use pre-trained vectors instead of training them every time from scratch. The Word2vec representation bears some advantages over the BOW counterpart, since terms which are near in meaning are close in the word embedding space and, somewhat surprisingly, other semantic relationships such as gender-inflections or geographical connections can be recovered using algebraic operations between vectors [13, 14]. Therefore, if the purpose is to extract semantic features able to bring some extra information related to the similarities among words, one may effectively resort to the Word2vec models which widely proved their ability to encode such extra knowledge.

The aim of the present study is to investigate the effectiveness of combining the semantic features provided by Word2vec with the lexical embedding generated by the well-established and more commonly used BOW model. In particular, our goal is to evaluate at what extent the Word2vec features complement and enrich the information comprised in the BOW representation and, specifically, to verify on an empirical basis whether the joint use of Word2vec and BOW in text classification for sentiment analysis leads to a sustainable performance improvement over the latter approach used alone. To this end, we designed and applied four hybrid sentence representations to convert textual data into numeric vectors, which benefit from both Word2vec and BOW information. These hybrid

variants are compared against two baselines given by the classical BOW and Word2vec methods applied individually.

Several tests in the context of sentiment classification are performed on five publicly available Amazon datasets, containing the users' opinions on products coming from different categories [10,11]. The results of our experiments highlighted the usefulness of the novel hybrid representations across the different domains. In particular, the features obtained by concatenating the BOW model with the averaged form of Word2vec consistently outperformed the corresponding baselines.

The remainder of the paper is organized as follows. Section 2 describes the original BOW and Word2vec approaches and the proposed hybrid variants. Section 3 illustrates the classification results achieved on the benchmark datasets. Finally, Sect. 4 contains the conclusions and the future research developments.

2 Representation of Sentences

To use machine learning algorithms the corpus of sentences must be transformed into a rectangular matrix of numeric values. Unlike the BOW model which produces a set of vectors which can be fed directly to a classifier, as described below, word vectors must be manipulated to be converted into unique sentence-level vectors of the same size. When word vectors follow the principle of compositionality, as in the case of Word2vec, two widely used strategies can be adopted. The first simply computes the average of the word vectors in the sentence [17]. The second resorts to a weighted average by considering the TF-IDF (term frequency times inverse document frequency) value of each term [6], which expresses the relative importance of a word inside a sentence [8].

2.1 Bag of Words

Given a corpus defined as a collection $D = (s_1, s_2, \dots, s_m)$ of m sentences, the BOW model maps each s_i into a numeric vector of dimension V , where V is the size of the vocabulary extracted from D in the form of a set (v_1, v_2, \dots, v_V) of unique different terms. This vector can be built according to a presence-based approach or, alternatively, to a frequency-based one [15]. In the first case, a given sentence s_i is converted into a boolean vector $\mathbf{bow01}_i$ whose generic element j takes the value 1 if and only if v_j appears in the sentence. In the second case, s_i is converted into a vector \mathbf{bowTF}_i containing at position j the TF-IDF value of word v_j , defined as the product between the term frequency, i.e. the number of occurrences of v_j in s_i , and the inverse term frequency, given by the logarithm of the ratio of the number of sentences divided by the number of sentences in the corpus containing v_j . Regardless the approach used, D is therefore transformed into a rectangular matrix consisting of V columns and m rows, each one composed by the bag of words of the corresponding sentence.

2.2 Word2vec

The most straightforward way to build a sentence-level vector using Word2vec is to average the word vectors of the terms therein included.

Formally, let \mathbf{w}_{ij} denote the d -dimensional pre-trained word vector corresponding to the j^{th} term in sentence s_i . In the averaged Word2vec model, s_i is mapped into a d -dimensional vector \mathbf{avg}_i whose k^{th} element is defined as

$$\mathbf{avg}_{ik} = \frac{1}{n_i} \sum_{j=1}^{n_i} \mathbf{w}_{ijk} \quad k = 1, 2, \dots, d, \quad (1)$$

where n_i is the number of words in sentence i .

An alternative approach is represented by the weighted average Word2vec, in which each sentence s_i is converted into the vector \mathbf{avgTF}_i , whose element at position k is given by

$$\mathbf{avgTF}_{ik} = \frac{1}{n_i} \sum_{j=1}^{n_i} f_{ij} \mathbf{w}_{ijk} \quad k = 1, 2, \dots, d, \quad (2)$$

where f_{ij} is the TF-IDF value of word j in sentence i . Notice that, for both strategies the vector representing each sentence has the same dimension d of the input word vectors. As a consequence, the original corpus is mapped into a numeric matrix of size $m \times d$ which can be fed to any machine learning classifier.

2.3 Hybrid Features: Combining BOW with Word2vec

The hybrid sentence representations proposed in this paper draw inspiration from previous studies which analyzed, even if with some limitations, the usefulness of using BOW with Word2vec. The complementarity of the information encoded by the two models, in particular, was empirically highlighted in [4] where an ensemble classifier built on BOW and Word2vec achieved the best results in 9 of the 11 domains for the purpose of sentiment classification. The work of [6] further confirmed this evidence by showing that concatenating TF-IDF weighted average features with frequency-based BOW vectors can outperform other approaches based on the latter alone even if not in all cases.

Along this path and in order to enrich the studies conducted so far, in our work we focused on features engineering rather than ensemble methods such in [4] since they present the drawback of high computational costs. In addition, we also designed novel hybrid embedding variants resorting to different types of vectors concatenation which, at the best of our knowledge, haven't been tested yet. Unlike [6], we introduced the use of a presence-based approach as suggested in [15] and the simple average, which showed better performances, compared to the weighted approach, as reported in [4]. Specifically, by combining in all possible ways the BOW and Word2vec schemes described above, we derived the following four sentence representations

$$\mathbf{avg}_i \oplus \mathbf{bow01}_i \quad (3)$$

$$avg_i \oplus bowTF_i \tag{4}$$

$$avgTF_i \oplus bow01_i \tag{5}$$

$$avgTF_i \oplus bowTF_i \tag{6}$$

where the operator \oplus denotes the concatenation of vectors. Notice that for each hybrid encoding the vector representing sentence i has a final dimension equal to the sum of the sizes of the component vectors. As an example, if avg_i is a d -dimensional vector and $bow01_i$ has the same size V of the vocabulary of terms extracted from the corpus, $avg_i \oplus bow01_i$ is a $(d+V)$ -dimensional feature vector.

3 Experimental Design and Results

To evaluate the effectiveness of the hybrid variants we performed several experiments on a publicly available¹ corpus of Amazon reviews [10,11], referred to products from different categories such as Beauty, Video Games, Clothing, Health and Home. Each review is described by different features: among these, we extracted the review text and its rating, originally ranging from 1 to 5 stars. For the purpose of classification we first discarded the reviews with neutral rating (3 stars) and assigned a positive (negative) polarity to the remaining comments rated more (less) than 3 stars. The polarity was then taken as the binary target variable to predict. A summary of the datasets used in our tests in terms of number of reviews, positive and negative comments, average length of the reviews and size of the vocabulary, is provided in Table 1. Notice that these datasets were chosen since their intrinsic diversity allowed to analyze the performances of the novel encoding schemes across different text domains.

Table 1. Amazon reviews datasets.

Dataset	N. reviews	Positive	Negative	Avg. length	Vocabulary
Beauty	176,229	154,250	21,979	89	14,105
Video games	203,463	174,954	28,509	204	29,201
Clothing	248,230	221,578	26,652	60	11,656
Health	313,057	279,764	33,293	93	17,706
Home	506,423	455,049	51,374	96	15,654

Before the experiments common text data preprocessing in the form of lower case conversion, tokenization and stopwords removal was applied. On the contrary, stemming and lemmatization were not used since, based on a preliminary exploration, these tasks turned out to be ineffective for the specific analysis.

¹ <http://jmcauley.ucsd.edu/data/amazon/>.

Computational tests were performed on seven different sentence representations, described in Table 2, composed by three baselines and four newly introduced hybrid variants. In particular, the *bow01* encoding, referred to the presence-based BOW model, was obtained by retaining the first 1000 most frequent terms for each class. The *avg* and the *avgTF* mappings, corresponding to the basic Word2vec schemes, were instead generated by using the 300-dimensional word vectors generated by [12] and trained on a subset of the Google News dataset². Specific terms for which the word vector representation is not available were discarded. Finally, the remaining encodings were built by concatenating the above vectors as described in Sect. 2.3. The code repository is open source and is fully available on request.

Table 2. Alternative sentence representations.

Name	Description	N. features	Type
<i>bow01</i>	Presence-based bag of words	1000	baseline
<i>avg</i>	Averaged word vectors	300	baseline
<i>avgTF</i>	Weighted average of word vectors	300	baseline
$avg \oplus bow01$	Concatenate avg and bow01	1300	hybrid
$avg \oplus bowTF$	Concatenate avg and bowTF	1300	hybrid
$avgTF \oplus bow01$	Concatenate avgTF and bow01	1300	hybrid
$avgTF \oplus bowTF$	Concatenate avgTF and bowTF	1300	hybrid

To discriminate between positive and negative reviews we resorted to the Logistic Regression classifier implemented in Weka [19]. Specifically, for each dataset we randomly extracted a stratified training sample of 5000 comments, using the remaining reviews for testing. The most promising ridge regularization parameter, searched among the powers of 10 in the interval $[10^{-4}, 1]$, was obtained through a ten-fold cross-validation on the training set. Furthermore, given the high unbalance of the datasets in terms of class distribution, we selected as performance measure the F1-score on the minority (negative) class.

The results of our experiments are shown in Table 3, which indicates the F1-score computed on the different test sets. The first main outcome is the notable performance exhibited by the hybrid encoding variants compared to the classical BOW and Word2vec representations. In particular, the $avg \oplus bow01$ mapping obtained the highest accuracy across all the datasets. This empirical achievement emphasizes the benefits stemming from the joint use of BOW and Word2vec, suggesting that exploiting the information encoded in the two schemes by concatenating the corresponding sentence-level vectors outperforms the baseline approaches. Notice that, due to the large size of the test sets, it is easy to observe that the 95% confidence intervals around the F1-scores do not overlap. This implies the statistical significance of our results.

² <https://code.google.com/archive/p/word2vec/>.

Moreover, among the hybrid embeddings the one relying on the averaged Word2vec and on the presence-based BOW model consistently provided better results compared to the TF-IDF weighted alternatives. This evidence confirms that the weighting scheme based on TF-IDF, originally proposed to account for the importance of the words in a document within a corpus, is not always an effective choice in a sentiment analysis task.

Table 3. F1-scores on the test sets.

Representation	Beauty	Video games	Clothing	Health	Home
<i>bow01</i>	0.49	0.53	0.51	0.41	0.47
<i>avg</i>	0.47	0.52	0.52	0.37	0.45
<i>avgTF</i>	0.38	0.40	0.42	0.29	0.35
$avg \oplus bow01$	0.51	0.54	0.52	0.43	0.48
$avg \oplus bowTF$	0.51	0.54	0.51	0.43	0.47
$avgTF \oplus bow01$	0.50	0.53	0.50	0.42	0.45
$avgTF \oplus bowTF$	0.50	0.53	0.50	0.42	0.45

Finally, as a further result we observed that the simple *bow01* encoding generated better predictions compared to both Word2vec baseline schemes on all datasets except for Clothing. Looking at the properties of this dataset we noticed that it collects the shortest reviews, on average, and gives rise to the smallest vocabulary. These two dimensions deserve further investigation since they might play a prominent role on the performance of the Word2vec encodings in different text classification domains.

4 Conclusions and Future Developments

Vectorial embedding of sentences can encode different information about the texts they represent. The most common sentence encoding schemes in the context of machine learning for sentiment analysis are the bag of words (BOW) and the Word2vec models. In the present study we evaluate the usefulness of combining BOW and Word2vec by designing novel hybrid sentence representations which exploit the information provided by both strategies. Experiments on benchmark datasets, which collect the reviews of Amazon’s products, confirmed the effectiveness of the hybrid mappings which showed notable performances in terms of prediction accuracy compared to the BOW and Word2vec approaches applied individually. Our empirical finding supports the results obtained in previous studies which conjectured on how the information provided by the well-established BOW scheme can be completed and enriched by the one contained in the more recently proposed Word2vec models. Given the promising results achieved, the present work could be developed in several directions. First, it

would be worthwhile to explore the accuracy of the proposed hybrid variants on a wider collection of textual datasets. Moreover, experiments could be extended by considering alternative classifiers, to evaluate the robustness of the conclusions to the change of the algorithm used for prediction. Finally, other forms of text embedding derived by the combination of numeric sentence encodings could be designed and investigated.

References

1. Bengio, Y., Ducharme, R., Vincent, P., Janvin, C.: A neural probabilistic language model. *J. Mach. Learn. Res.* **3**, 1137–1155 (2003)
2. Bollen, J., Mao, H., Zeng, X.: Twitter mood predicts the stock market. *J. Comput. Sci.* **2**(1), 1–8 (2011). <https://doi.org/10.1016/J.JOCS.2010.12.007>
3. Collobert, R., Weston, J.: A unified architecture for natural language processing. In: *ICML 2008*, pp. 160–167. ACM Press (2008). <https://doi.org/10.1145/1390156.1390177>
4. Enríquez, F., Troyano, J.A., López-Solaz, T.: An approach to the use of word embeddings in an opinion classification task. *Expert Syst. Appl.* **66**, 1–6 (2016). <https://doi.org/10.1016/j.eswa.2016.09.005>
5. Jansen, B.J., Zhang, M., Sobel, K., Chowdury, A.: Twitter power: tweets as electronic word of mouth. *J. Am. Soc. Inf. Sci. Technol.* **60**(11), 2169–2188 (2009). <https://doi.org/10.1002/asi.21149>
6. Lilleberg, J., Zhu, Y., Zhang, Y.: Support vector machines and Word2vec for text classification with semantic features. In: *2015 ICCI*CC*, pp. 136–140. IEEE, July 2015. <https://doi.org/10.1109/ICCI-CC.2015.7259377>
7. Liu, B.: *Sentiment Analysis*. Cambridge University Press, Cambridge (2015). <https://doi.org/10.1017/CBO9781139084789>
8. Manning, C.D., Raghavan, P., Schütze, H.: Scoring, term weighting, and the vector space model. In: *Introduction to Information Retrieval*, pp. 100–123. Cambridge University Press (2008). <https://doi.org/10.1017/cbo9780511809071.007>
9. Mäntylä, M.V., Graziotin, D., Kuuttila, M.: The evolution of sentiment analysis—A review of research topics, venues, and top cited papers. *Comput. Sci. Rev.* **27**, 16–32 (2018). <https://doi.org/10.1016/J.COSREV.2017.10.002>
10. McAuley, J., Pandey, R., Leskovec, J.: Inferring networks of substitutable and complementary products. In: *ACM SIGKDD 2015*, pp. 785–794. ACM Press, New York (2015). <https://doi.org/10.1145/2783258.2783381>
11. McAuley, J., Targett, C., Shi, Q., van den Hengel, A.: Image-based recommendations on styles and substitutes. In: *SIGIR 2015*, pp. 43–52. ACM Press, New York (2015). <https://doi.org/10.1145/2766462.2767755>
12. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *Advances in Neural Information Processing Systems*, pp. 3111–3119 (2013)
13. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. *CoRR abs/1301.3781* (2013). <http://arxiv.org/abs/1301.3781>
14. Mikolov, T., Yih, W.t., Zweig, G.: Linguistic regularities in continuous space word representations. In: *NAACL HLT 2013*, pp. 746–751 (2013). <http://www.aclweb.org/anthology/N13-1090>

15. Pang, B., Lee, L.: *Opinion Mining and Sentiment Analysis*, vol. 2. Now Publishers, Inc., Delft (2008). <https://doi.org/10.1561/15000000011>
16. Piryani, R., Madhavi, D., Singh, V.: Analytical mapping of opinion mining and sentiment analysis research during 2000–2015. *Inf. Process. Manag.* **53**(1), 122–150 (2017). <https://doi.org/10.1016/J.IPM.2016.07.001>
17. Socher, R., et al.: Recursive deep models for semantic compositionality over a sentiment treebank. In: *EMNLP 2013*, pp. 1631–1642. ACL (2013)
18. Tumasjan, A., Sprenger, T.O., Sandner, P.G., Welpke, I.M.: Election forecasts with Twitter. *Soc. Sci. Comput. Rev.* **29**(4), 402–418 (2010). <https://doi.org/10.1177/0894439310386557>
19. Witten, I.H., Frank, E., Hall, M.A., Pal, C.J.: *Data Mining: Practical Machine Learning Tools and Techniques*. Elsevier Inc., Amsterdam (2016)



ALoT: A Time-Series Similarity Measure Based on Alignment of Textures

Hasan Oğul^(✉)

Faculty of Computer Sciences, Østfold University College,
P.O. Box 700, 1757 Halden, Norway
hasan.ogul@hiof.no

Abstract. Inferring the similarity between two time-series signals is a key step in several data analysis tasks for a variety of engineering applications. In this paper, we introduce a novel elastic similarity measure (ALoT) based on the alignment of textures, instead of observed values, extracted from input time-series signals. To obtain the texture information, Local Binary Patterns are adapted for one-dimensional signals. According to experiments performed on a large number of benchmark time-series classification datasets, the proposed method achieves higher accuracy than current pairwise similarity measures for several cases in a 1-Nearest Neighbor classification setup.

Keywords: Time-series analysis · Similarity measure · Classification
Sensory data

1 Introduction

Applications of time-series data analysis have become ubiquitous in domains such as health, finance, telecommunications, security, meteorology, and marketing [15]. The emergence of recent technologies in wireless sensor networks and the Internet of Things has increased the demand for more effective analysis of such data. The research in this field has been enriched by computational tasks such as the classification [2–8, 12, 18, 22–24, 26], clustering [17, 19], and retrieval [15] of data collected from these application domains. A key step that is common to all these tasks involves inferring the similarity between two time-series data in a quantitative way. It has been shown that the similarity measure is often more important to the results than the classification or clustering algorithm itself [5]. In fact, in many cases, a simple 1-Nearest Neighbor (1-NN) classifier can outperform more complex machine learning methods with a properly selected similarity measure [9].

Existing similarity (or distance) measures for time-series data can be broadly categorized into three classes: vector-based, feature-based, and alignment-based (usually called elastic) measures. Vector-based methods take the observed values of two signals as numeric vectors indexed by time points and compute the geometric distance between them without considering temporal details. A common example is the Euclidean distance. The major problem with this approach is the fact that the time shifts between two signals to be compared are totally ignored in the similarity calculation.

Featured-based methods transform the original time-series signals into a set of extracted numeric features to which a vector-based comparison is applied. The extracted features to be compared vary from simple time-domain statistics, such as the mean or variance, to more complex frequency- or wavelet-domain features obtained after some transformations. Determining which features to use is highly dependent on the application domain. For example, a recent study suggests that using the histogram of textures obtained from Local Binary Patterns (LBPs) as signal features can provide higher accuracy than using several time- and frequency-domain features in classifying time-series accelerometer signals [1]. Although feature-based methods can compensate the time-shift problem to some degree, their generalization ability is poor because the extracted features are usually ineffective in representing the global order of temporal values.

An obvious solution to these problems is to use elastic measures. The main principle of elastic measures is to align whole series of observed data in the time domain and report the alignment score, usually in the form of the quantity of matches, as a similarity (or distance) measure. A large majority of the effort in time-series similarity analysis has focused on developing more effective elastic measures. The most common elastic measure, Dynamic Time Warping (DTW) [16, 21], attempts to minimize the sum of the differences between aligned temporal values from two input signals. The Time Warp Edit Distance (TWED) [20] is an alternative alignment-based measure that controls the elasticity by a stiffness adjustment and applies a mismatch penalty when computing the alignment score. Two recent remarkable improvements of DTW are WDTW [14], a locally weighted version of the original DTW, and DD_DTW [11], a derivative-based extension. Another effective elastic measure is the Move-Split-Merge (MSM) metric [25], which uses a set of editing operations to transform input signals into new forms that minimize the alignment score. CID [5] is a more recent extension of DTW that considers the differences in complexity of two input signals. The main limitation of elastic measures is the fact that exact point measurements are forced to match by warping to minimize the difference between two input signals. This leads to some neglect of the local temporal behavior of time-series signals, regardless of their exact values.

In this study, a novel time-series similarity measure is introduced. The proposed ALoT (ALignment of Textures) combines two ideas borrowed from elastic measures and feature-based measures. First, similar to all other elastic measures, ALoT aligns two input signals to eliminate the problems of probable time shifts in the two instances. However, instead of targeting the exact temporal values, it attempts to align the local categories of the temporal changes in the input signals. This prevents the potential problems of different scaling, amplitude/offset invariance, and contextual changes in the observed values of similar time-series pairs. Second, as in other feature-based methods, ALoT extracts time-domain features to categorize any temporal observation to a texture. Whereas previous feature-based methods use these feature sets to make a vectorial comparison in computing an indirect measure of similarity, ALoT integrates this information into the alignment procedure. In this way, the local order of these features can also be considered. ALoT uses a dynamic-programming-based algorithm that is applied directly to texture information. The texture information is obtained through the analysis of LBPs adapted for one-dimensional signals.

To discern the ability of the proposed method, a set of time-series classification experiments are performed using a 1-NN classifier with ALoT to calculate the pairwise similarity between time-series samples. Experiments performed on a large number of benchmark datasets demonstrate that, in several cases, ALoT achieves higher accuracy than current elastic similarity measures.

2 Methods

A time-series signal is an ordered set of observations indexed by discrete time points. The time-series similarity is defined as a quantitative measure of the similarity between two time-series signals indexed by the same time scale. Given the time-series signals $X = (x_1, x_2, \dots, x_m)$ and $Y = (y_1, y_2, \dots, y_n)$, the task is to compute a score $S(X, Y)$ that grades the similarity of their content. While it is possible to make a vectoral comparison using metrics such as the Euclidean distance, the common approach is to use a non-linear (elastic) alignment that produces a more intuitive similarity measure by matching similar observations, even if they are out of phase with respect to the time axis. A typical and powerful example of elastic measures is DTW, which computes the distance score $D(X, Y)$, instead of a similarity score, by aligning observed values of the input time-series signals to minimize the total absolute difference between individual observations at each time point (Algorithm 1). The *dist* function in step 4 usually refers to the simple arithmetic difference of the inputs.

Algorithm 1. DTW(X, Y)

Parameters: None

- 1: Let H be an $m \times n$ matrix initialized to ∞
 - 2: $H(0,0) \leftarrow 0$
 - 3: for $i \leftarrow 1$ to m do
 - 4: for $j \leftarrow 1$ to n do
 - 5: $H(i, j) = \text{dist}(x_{i-1}, y_{j-1}) + \min(H(i-1, j-1),$
 $H(i-1, j),$
 $H(i, j-1))$
 - 6: return ($D(X, Y) \leftarrow H(m, n)$)
-

2.1 ALoT - Texture Alignment

The ALoT algorithm for calculating the similarity $S(X, Y)$ based on their alignment is given in Algorithm 2.

Algorithm 2. ALoT(X, Y)

Parameters: w, σ

- 1: Let H be an $m+1 \times n+1$ matrix initialized to zero
 - 2: $A \leftarrow toTexture(X, w)$
 - 3: $B \leftarrow toTexture(Y, w)$
 - 4: for $i \leftarrow 2$ to $m+1$ do
 - 5: for $j \leftarrow 2$ to $n+1$ do
 - 6: $H(i, j) = \max(H(i-1, j-1) - distHamming(a_{i-1}, b_{j-1}),$
 $\qquad\qquad\qquad H(i-1, j) - \sigma,$
 $\qquad\qquad\qquad H(i, j-1) - \sigma)$
 - 7: return ($S(X, Y) \leftarrow H(m+1, n+1)$)
-

There are three major differences between ALoT and classical DTW. First, ALoT compares the extracted textures of two signals, whereas DTW directly compares the exact values of the observations. The function *toTexture* (steps 2 and 3 in Algorithm 2) is the procedure that transforms the input signal into a sequence of textures extracted from the observed time-series data. This enables us to avoid the problems caused by different scaling, amplitude, and offset invariance and contextual changes. w is a texture extraction parameter (see Sect. 2.2). Second, while DTW uses a simple arithmetic difference to compare any observed values, ALoT computes the Hamming distance between the bit sequences that represent the observed local texture (the function *distHamming* in step 6 of Algorithm 2). Owing to this, the similarity between any local temporal behavior can be easily quantified, regardless of the exact values observed at those time points. The Hamming distance counts the number of bits that differ between two binary strings. More formally, the distance between two binary strings $P = p_1p_2\dots p_k$ and $Q = q_1q_2\dots q_k$ is given by:

$$distHamming(P, Q) = \sum_i |p_i - q_i|.$$

Third, ALoT directly computes a similarity, but not a distance, by attempting to maximize the alignment score. This is done using a mismatch penalty σ that is subtractive to final score in cases where two temporal observations do not match in the resulting alignment. This penalty becomes equal to the Hamming distance between two observed textures when they are determined to match. This property is similar to that of TWED [20], which uses a so-called stiffness parameter to control the elasticity. However, TWED still computes the mismatch penalty through the exact observation values, but not for their textures.

2.2 ALoT - Texture Extraction

To extract the texture from input time-series data, we use the concept of LBPs in image processing theory. LBPs were originally defined for two-dimensional image data, but we adapt them here for one-dimensional signals. This idea has been used successfully to make a feature-based representation of time-series data in different domains [1, 10, 13]. Given $X = (x_1, x_2, \dots, x_n)$, the texture transformation of X based on LBPs is given by $A = (a_1, a_2, \dots, a_n)$, where a_i is a sequence of $2w$ bits, i.e., $p_{i-w}p_{i-w+1} \dots p_{i-1}p_{i+1} \dots p_{i+w-1}p_{i+w}$. For any position $i + j$, the corresponding value of that bit is computed as:

$$p_{i+j} = \begin{cases} 1 & \text{if } x_i > x_{i+j} \\ 0 & \text{otherwise} \end{cases}.$$

In previous methods, the bit sequence a_i is converted to an integer variable to represent the local texture categorically. A histogram of these integer values is then used to represent the input time-series, which can be used as a fixed length feature set for the relevant data analysis. Here, we opt to use the bit sequence directly, instead of converting it to a categorical variable, as the similarity between these bit sequences given by the Hamming distance can unveil the similarity between local temporal behavior.

3 Results

3.1 Experimental Setup and Evaluation

We performed experiments on 81 datasets obtained from [3]. Each dataset refers to a distinct time-series classification task with varying numbers of class labels. The repository contains time-series signals collected from seven different sources, i.e., sensor, spectrograph, motion, image, EEG, device and simulated. A training/test split is provided for each dataset within the repository. For each of the datasets, all time-series samples are of equal length.

To discern the ability of ALoT for inferring time-series similarity, we used a 1-NN-based classification framework. In this framework, an unknown query sample is first compared against every instance of the training set, and then labeled with the same class as the instance that has the highest similarity with the query. As the classification method has no learning stage, the contribution of the similarity measure can be clarified in this setup. The classification accuracy is measured as the number of correctly predicted samples divided by the number of all samples in the test set. This evaluation is commonly used to benchmark time-series similarity measures [3].

3.2 Empirical Results

The ALoT algorithm has two free parameters, w and σ , denoting the length of the LBP window and the mismatch penalty for alignment, respectively. We intuitively set $w = 2$ in our experiments, as the texture information would not be sufficiently representative for $w = 1$, whereas the representation of texture categories would be sparse for larger

Table 1. Comparison of elastic measures for time-series similarity.

Dataset	Accuracy						
	DTW	MSM	TWED	WDTW	DD_DTW	CID	ALoT
Adiac	0.604	0.627	0.634	0.606	0.701	0.624	0.271
ArrowHead	0.703	0.806	0.794	0.817	0.789	0.829	0.772
Beef	0.633	0.467	0.600	0.700	0.667	0.633	0.867
BeetleFly	0.700	0.600	0.700	0.700	0.650	0.750	0.800
BirdChicken	0.750	0.850	0.850	0.750	0.850	0.750	0.700
Car	0.733	0.900	0.917	0.783	0.800	0.767	0.783
CBF	0.997	0.969	0.991	0.997	0.997	0.999	0.403
ChlorineConcentration	0.648	0.628	0.648	0.649	0.708	0.649	0.732
CinCECGtorso	0.651	0.962	0.846	0.859	0.725	0.946	0.985
Coffee	1.000	0.893	1.000	1.000	1.000	1.000	0.965
Computers	0.700	0.576	0.632	0.700	0.716	0.620	0.660
CricketX	0.754	0.764	0.749	0.790	0.754	0.777	0.493
CricketY	0.744	0.772	0.754	0.759	0.777	0.713	0.462
CricketZ	0.754	0.764	0.736	0.762	0.774	0.731	0.554
DiatomSizeReduction	0.967	0.948	0.948	0.967	0.967	0.935	0.875
DistalPhalanxOAG	0.717	0.736	0.739	0.717	0.732	0.736	0.767
DistalPhalanxOC	0.770	0.741	0.712	0.698	0.705	0.633	0.667
DistalPhalanxTW	0.590	0.640	0.633	0.604	0.612	0.626	0.660
Earthquakes	0.719	0.691	0.676	0.727	0.705	0.719	0.808
ECG200	0.770	0.910	0.890	0.880	0.830	0.890	0.840
ECG5000	0.924	0.932	0.927	0.925	0.924	0.927	0.935
ECGFiveDays	0.768	0.891	0.829	0.796	0.769	0.782	0.962
ElectricDevices	0.602	0.659	0.593	0.613	0.592	0.616	0.570
FaceAll	0.808	0.809	0.786	0.794	0.902	0.864	0.727
FacesFour	0.808	0.809	0.786	0.794	0.902	0.864	0.898
FacesUCR	0.830	0.943	0.852	0.875	0.830	0.875	0.820
FiftyWords	0.690	0.813	0.793	0.771	0.754	0.780	0.552
Fish	0.823	0.931	0.931	0.851	0.943	0.834	0.886
FordA	0.555	0.719	0.618	0.653	0.723	0.772	0.780
FordB	0.620	0.642	0.609	0.599	0.667	0.627	0.681
GunPoint	0.907	0.973	0.953	0.980	0.980	0.927	0.927
Ham	0.467	0.514	0.514	0.581	0.476	0.514	0.562
HandOutlines	0.881	0.876	0.870	0.870	0.868	0.862	0.646
Haptics	0.377	0.442	0.416	0.370	0.399	0.425	0.429
Herring	0.531	0.563	0.516	0.531	0.547	0.516	0.641
InlineSkate	0.384	0.453	0.422	0.404	0.562	0.415	0.371
InsectWingbeatSound	0.355	0.575	0.534	0.574	0.355	0.578	0.481
ItalyPowerDemand	0.950	0.944	0.948	0.950	0.950	0.956	0.929
LargeKitchenAppliances	0.795	0.669	0.752	0.795	0.795	0.792	0.493
Lightning2	0.869	0.820	0.836	0.902	0.869	0.869	0.541

(continued)

Table 1. (continued)

Dataset	Accuracy						
	DTW	MSM	TWED	WDTW	DD_DTW	CID	ALoT
Lightning7	0.726	0.753	0.753	0.767	0.671	0.699	0.200
Mallat	0.934	0.932	0.912	0.938	0.949	0.925	0.701
Meat	0.933	0.933	0.933	0.933	0.933	0.933	0.766
MedicalImages	0.737	0.741	0.711	0.737	0.737	0.742	0.540
MiddlePhalanxOAG	0.698	0.753	0.763	0.753	0.732	0.763	0.720
MiddlePhalanxOC	0.500	0.494	0.519	0.519	0.539	0.513	0.723
MiddlePhalanxTW	0.506	0.494	0.487	0.513	0.487	0.513	0.539
MoteStrain	0.835	0.855	0.797	0.859	0.833	0.796	0.866
NonInvasiveFatalECGThorax1	0.790	0.816	0.820	0.816	0.806	0.837	0,376
NonInvasiveFatalECGThorax2	0.865	0.883	0.888	0.884	0.893	0.879	0,458
OliveOil	0.833	0.833	0.867	0.833	0.833	0.867	0.534
OSULeaf	0.591	0.773	0.777	0.624	0.880	0.620	0.802
PhalangesOC	0.728	0.752	0.760	0.747	0.739	0.762	0.673
Phoneme	0.228	0.292	0.270	0.161	0.269	0.221	0.244
Plane	1.000	1.000	1.000	1.000	1.000	1.000	0.991
ProximalPhalanxOAG	0.784	0.787	0.808	0.784	0.794	0.790	0.785
ProximalPhalanxOC	0.805	0.790	0.805	0.805	0.800	0.790	0.725
ProximalPhalanxTW	0.761	0.756	0.746	0.751	0.766	0.751	0.715
RefrigerationDevices	0.464	0.483	0.512	0.427	0.445	0.445	0.468
ScreenType	0.397	0.459	0.427	0.411	0.429	0.405	0.371
ShapeletSim	0.650	0.867	0.578	0.756	0.611	0.744	0.939
ShapesAll	0.768	0.872	0.860	0.812	0.850	0.808	0.874
SmallKitchenAppliances	0.643	0.699	0.645	0.675	0.640	0.675	0.504
SonyAIBORobotSurface1	0.725	0.730	0.681	0.737	0.742	0.815	0.797
SonyAIBORobotSurface2	0.831	0.871	0.853	0.831	0.892	0.877	0.856
StarlightCurves	0.907	0.868	0.883	0.895	0.962	0.918	0.897
Strawberry	0.941	0.943	0.943	0.943	0.954	0.943	0.930
SwedishLeaf	0.792	0.896	0.891	0.874	0.901	0.882	0.694
Symbols	0.950	0.949	0.960	0.950	0.953	0.941	0.977
SyntheticControl	0.993	0.973	0.987	0.993	0.993	0.973	0.380
ToeSegmentation1	0.772	0.816	0.820	0.794	0.807	0.737	0.794
ToeSegmentation2	0.838	0.754	0.785	0.892	0.746	0.877	0.939
Trace	1.000	0.930	0.990	1.000	1.000	0.990	0.830
TwoLeadECG	0.905	0.947	0.974	0.905	0.978	0.881	0.753
TwoPatterns	1.000	0.999	0.999	1.000	1.000	0.998	0.768
Wafer	0.980	0.997	0.996	0.997	0.980	0.994	1.000
UWaveGestureLibraryAll	0.728	0.769	0.771	0.774	0.779	0.790	0.907
Wine	0.574	0.593	0.574	0.574	0.574	0.611	0.685
WordSynonyms	0.649	0.763	0.749	0.745	0.730	0.757	0.543
Worms	0.584	0.571	0.571	0.532	0.584	0.610	0.602
WormsTwoClass	0.623	0.675	0.610	0.571	0.649	0.675	0.707

values of w . For each of the 81 datasets, the value of σ was chosen from the set $\{2, 3, 4\}$ using cross-validation on the training set, which confirms that it was not optimized based on the results on the test data. Table 1 compares ALoT with previous elastic time-series similarity measures, namely DTW [16, 21], MSM [25], TWED [20], WDTW [14], DD_DTW [11], and CID [4], in terms of accuracy for each task.

As shown by these results, ALoT can achieve better performance than all other methods for 23 of the given tasks. The highest improvement was achieved for the dataset labeled as ‘MiddlePhalanxOutlineCorrect’, where ALoT improved the accuracy of the best previous method by 34%.

One of the major contributions of this study is the idea of aligning LBPs instead of vectorially comparing their histograms to infer the similarity between textures. To discern the benefit of this idea, we performed a detailed comparative evaluation between ALoT and the conventional use of LBPs. In Fig. 1, a pairwise comparison plot is presented for ALoT against the LBP histogram with respect to all classification tasks in the benchmarks datasets. ALoT outperforms the vector-based comparison LBP histograms for a large majority (80.2%) of tasks. According to paired t-tests, this result is statistically significant with a p-value of 1.9×10^{-13} . This comparison reveals the advantage of aligning textures instead of comparing their unordered content in inferring time-series similarity.

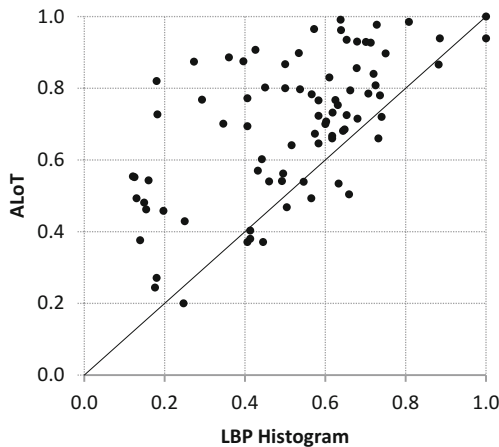


Fig. 1. Pairwise comparison plot for ALoT vs. LBP histogram.

4 Conclusion

Inferring the similarity between time series has regained importance because of the increasing amount of streaming data available through networked devices. This study has proposed a novel measure for time-series similarity that takes account of local temporal behavior in the input signals using extracted texture information. The

experimental results show that ALoT is a powerful alternative to conventional elastic similarity measures and verify that aligning one-dimensional textures is more effective than comparing their histograms in analyzing time-series data. It is anticipated that several extensions will soon be developed by the community to further improve the predictive performance of ALoT.

References

1. Asuroglu, T., Acici, K., Erdas, C.B., Oğul, H.: Texture of activities: exploiting local binary patterns for accelerometer data analysis. In: 12th International Conference on Signal-Image Technology and Internet-Based Systems (2016)
2. Bagnall, A., Lines, J., Hills, J., Bostrom, A.: Time-series classification with COTE: The collective of transformation-based ensembles. *IEEE Trans. Knowl. Data Eng.* **27**, 2522–2535 (2015)
3. Bagnall, A., Lines, J., Bostrom, A., Large, J., Keogh, E.: The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Min. Knowl. Discov.* **31**, 606–660 (2017)
4. Batista, G., Keogh, E., Tataw, O., deSouza, V.: CID: an efficient complexity-invariant distance measure for time series. *Data Min. Knowl. Discov.* **28**, 634–669 (2014)
5. Batista, G., Silva, D., de Souza, V.: Time series classification using compression distance of recurrence plots. In: 13th IEEE International Conference on Data Mining (2013)
6. Baydogan, M., Runger, G.: Time series representation and similarity based on local autopatterns. *Data Min. Knowl. Discov.* **2**, 476–509 (2016)
7. Baydogan, M., Runger, G., Tuv, E.: A bag-of-features framework to classify time series. *IEEE Trans. Pattern Anal. Mach. Intell.* **25**, 2796–2802 (2013)
8. Corduas, M., Piccolo, D.: Time series clustering and classification by the autoregressive metric. *Comput. Stat. Data Anal.* **52**, 860–1872 (2008)
9. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and mining of time series data: Experimental comparison of representations and distance measures. In: 34th International Conference on Very Large Data Bases (2008)
10. Ertuğrul, O.F., Kaya, Y., Tekin, R., Almali, M.N.: Detection of Parkinson's disease by shifted one dimensional local binary patterns from gait. *Expert Syst. Appl.* **56**, 156–163 (2016)
11. Gorecki, T., Luczak, M.: Using derivatives in time series classification. *Data Min. Knowl. Discov.* **26**, 310–331 (2013)
12. Hills, J., Lines, J., Baranauskas, E., Mapp, J., Bagnall, A.: Classification of time series by shapelet transformation. *Data Min. Knowl. Discov.* **28**, 851–881 (2014)
13. Houam, L., Hafiane, A., Boukrouche, A., Lespessailles, E., Jennane, R.: One dimensional local binary pattern for bone texture characterization. *Pattern Anal. Appl.* **17**, 179–193 (2014)
14. Jeong, Y., Jeong, M., Omitaomu, O.: Weighted dynamic time warping for time series classification. *Pattern Recognit.* **44**, 2231–2240 (2011)
15. Keogh, E.: Data mining and information retrieval in time series/multimedia databases. In: 14th ACM International Conference on Multimedia (2006)
16. Keogh, E.: Exact indexing of dynamic time warping. In: International Conference on Very Large Data Bases (2002)
17. Liao, T.W.: Clustering of time series data—A survey. *Pattern Recognit.* **38**, 1837–1874 (2005)

18. Lines, J., Bagnall, A.: Time series classification with ensembles of elastic distance measures. *Data Min. Knowl. Discov.* **29**, 565–592 (2015)
19. Maharaj, E.A.: Clusters of time series. *J. Classif.* **17**, 297–314 (2000)
20. Marteau, P.: Time warp edit distance with stiffness adjustment for time series matching. *IEEE Trans. Pattern Anal. Mach. Intell.* **31**, 306–318 (2009)
21. Ratanamahatana, C., Keogh, E.: Three myths about dynamic time warping data mining, In: *Proceedings of 5th SDM* (2005)
22. Rodriguez, J., Alonso, C., Maestro, J.: Support vector machines of interval-based features for time series classification. *Knowl. Based Syst.* **18**, 171–178 (2005)
23. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. Acoust. Speech Signal Process.* **26**, 43–49 (1978)
24. Senin, P., Malinchik, S.: SAX-VSM: interpretable time series classification using sax and vector space model. In: *Proceedings of 13th IEEE ICDM* (2013)
25. Stefan, A., Athitsos, V., Das, G.: The Move-Split-Merge metric for time series. *IEEE Trans. Knowl. Data Eng.* **25**, 1425–1438 (2013)
26. Ye, L., Keogh, E.: Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. *Data Mining Knowl. Discov.* **22**, 149–182 (2011)



Intelligent Agents in a Blockchain-Based Electronic Voting System

Michał Pawlak, Aneta Poniszewska-Marańda^(✉) , and Jakub Guziur

Institute of Information Technology, Lodz University of Technology, Lodz, Poland
{michal.pawlak, jakub.guziur}@edu.p.lodz.pl,
aneta.poniszewska-maranda@p.lodz.pl

Abstract. There exist many approaches to electronic voting. Each of them has its own advantages and disadvantages. However, most of them suffer from lack of transparency and auditability. Recently developed blockchain technology may provide a solution to these issues. This paper presents a use of multi-agent system idea and intelligent agents in Auditable Blockchain Voting System (ABVS), which is a blockchain-based supervised and non-remote internet voting system intended to be end-to-end verifiable and auditable.

Keywords: E-voting · Blockchain · E-voting system
Intelligent agent · Multi-agent system

1 Introduction

Voting is the most basic and important process of democracy. For this reason, it is protected by complex security measures to ensure it is fair and valid. However, even today, the voting process is not free from frauds and manipulations [1, 2]. Furthermore, modern voting systems are slow due to a necessity of collecting ballots from different locations before they can be counted. Results of such voting are not verifiable because voters do not have means of verifying whether their votes were included in the results or were tampered with. Finally, voting is a very expensive process due to costs of ballots, staff and overall organization.

In order to solve these issues, many electronic voting systems were proposed [3]. Electronic voting, or e-voting, is a broad term that covers many approaches and implementations [12–14]. In simple terms, it may be defined as employment of electronic means or information technologies to conduct a voting process [3, 11]. In general, an e-voting system can be divided into six phases, which are: (i) registration; (ii) authentication; (iii) authorization; (iv) vote casting; (v) vote counting; (vi) vote verification [8, 10]. Using electronic voting systems provides many benefits, for instance: fraud prevention; results processing acceleration; human involvement and error reduction; reduction of a number of spoiled ballots due to better ballot presentation; decrease of voting overhead and associated costs; increase voting availability and potential for more direct democracy due to a possibility of a remote voting [9].

Unfortunately, e-voting systems suffer from numerous issues and must face many challenges. The most important and significant include: (i) lack of transparency and understanding of e-voting solutions that results in a lack of trust, which is crucial for any voting system; (ii) lack of widely accepted standards, which also decreases trust; (iii) risk of fraud and manipulation by privileged insiders or system providers; (iv) increased costs of required infrastructure, regarding power supply and communication technology [9, 15].

Introduced in 2008 blockchain technology may provide a solution to some of the mentioned issues. The technology has many properties, which are desirable in e-voting systems. The first one is ability to provide a base for a platform for public data storage and verification. This would enable common voters to follow and audit the voting results without dependency on dedicated institutions and officials. Appearance of blockchain technology created an opportunity for development of new e-voting solutions. In some countries such research is already ongoing [4, 5]. The main example of such country is Estonia, which utilizes blockchain in national health, judicial, legislative, security and commercial code systems and is currently researching application of blockchain in its e-voting procedure [5]. In 2017 and 2018 respectively, South Korea and Sierra Leone conducted a successful electronic voting with systems based on blockchain technology [4, 6].

This paper presents the use of intelligent agents and multi-agent system concepts for a blockchain-based electronic voting system. The main goal of this solution is to provide a transparent and auditable e-voting system. The idea of using intelligent agents in an offline e-voting system was presented in [16].

The paper is structured as follows: Sect. 2 presents an outline of Auditable Blockchain Voting System (ABVS) while Sect. 3 deals with a model of multi-agent system for security protection in ABVS.

2 Auditable Blockchain Voting System

Auditable Blockchain Voting System (ABVS) is a voting system described in detail in [19]. It was designed as a non-remote and supervised voting system based on blockchain technology and which utilizes the Internet connection for transmitting the votes. ABVS architecture is made of three components:

- super-node,
- network of trusted nodes,
- polling stations.

The *super-node* represents national electoral authorities (National Electoral Commission in case of Poland). This component is responsible for starting the election and providing other nodes with setup information. Furthermore, the super-node participates in consensus algorithm and block mining like a trusted node.

Trusted nodes are public institutions selected and approved by election officials, which will provide computational power and storage capability for an

ABVS blockchain network. Their task is to collect votes and append them to a blockchain. Moreover they must verify correctness of the whole chain by participating in a consensus algorithm. Finally, they provide backup in case of damage or loss of a main chain.

Polling stations are lightweight voting applications representing individual voting districts. They are used as interfaces for voters to cast their votes in a form of a blockchain transaction. Each application must be verified and signed by the election officials.

As mentioned, all presented components are connected in a blockchain-based peer-to-peer network. A general schema of the ABVS network of nodes is presented in Fig. 1.

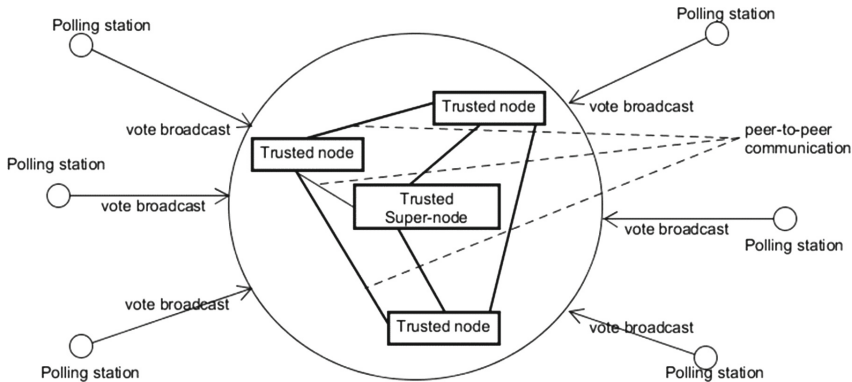


Fig. 1. Schema of the ABVS block-chain network

Voting process in ABVS consists of three phases:

- initiation phase,
- voting phase,
- counting and verification phase.

In the *initiation phase* trusted public institutions are selected to create the ABVS network of nodes. Next, election equipment must be prepared and election officials must verify and sign it. Finally, Vote Identification Tokens (VITs) must be generated and distributed between voting districts. VITs are used for authorization and identification of votes in the following phase.

The *voting phase* begins with installation of the ABVS equipment at electoral districts. When an election starts, voters identify themselves at their voting district. This ensures that only authenticated and authorized voters participate in the election. Subsequently, the voters randomly select VIT stored on any medium which allows random selection without showing its content in advance. Finally, the voters can proceed to polling stations, where they enter their respective VITs and cast their votes. The votes are broadcasted to the ABVS network

where they are verified and added to the ABVS blockchain. Furthermore, the voters are provided with Voter-Verified Paper Audit Trails (VVPATs) which serve as additional safeguard and are submitted into ballot boxes as in traditional voting.

The *counting and verification phase* is initiated after time designated for the election elapses. The ABVS equipment and software is deactivated and finalisation of the election starts. The ABVS blockchain is validated and votes contained within are counted. The voters can use their VITs to identify their votes and verify whether they were correctly included in the results. In case of any discrepancies, corresponding VVPATs are located and used as reference. After the time for finalization has passed, the results are approved and the final results are released to the public. Figure 2 presents a schema of the ABVS process.

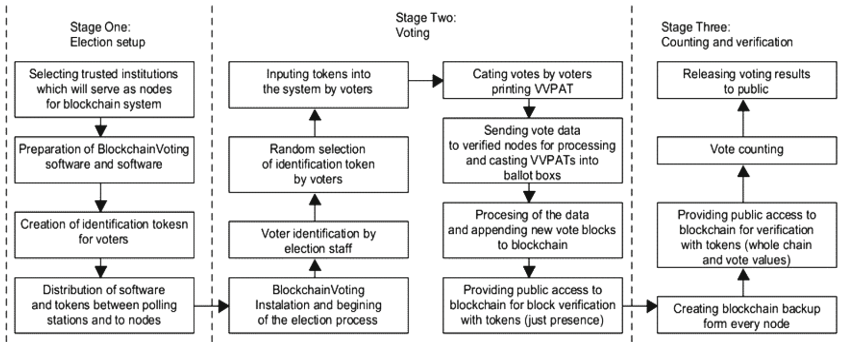


Fig. 2. Scheme of the ABVS voting process

3 Multi-agent System in Auditable Blockchain Voting System

Described in the previous section Auditable Blockchain Voting System may be improved with a use of intelligent agents [17, 18], which could potentially increase its security. The idea is based around implementation of two types of intelligent agents operating in the second phase of the ABVS process. These types consist of:

- authorization-configuration agent,
- voting agent.

The *authorization-configuration agent* would be tasked with authorization and configuration of the voting applications at the polling stations. Furthermore, the agent would be responsible for conducting transactions between the voting applications and the trusted-nodes. Through these transactions, the voters would

receive voting ballots, which would be delivered in a form of *voting agents*. The voting agents would enable the voters to cast their votes and would be responsible for transmitting the votes to the trusted nodes.

The authorization-configuration agent conducts its functions in three phases:

1. authorization phase,
2. configuration phase,
3. voting phase.

The *authorization phase* would start after installation of the ABVS software at the polling stations. Each of the installed applications would send authorisation requests to its nearest trusted node. The node would respond by sending authorization-configuration agent, which upon arrival, would verify correctness of the installation and authorize the application, to enable it to participate in the voting.

Subsequently, in the *configuration phase*, the agent would perform configuration of the application by collecting information regarding identification of a given polling station from a supervising official. A schema of the authorisation and the configuration phases is presented in Fig. 3.

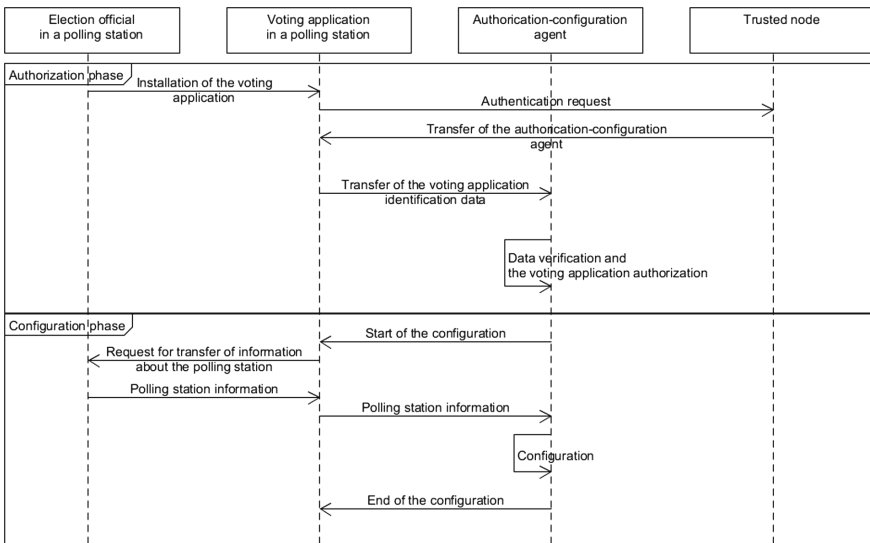


Fig. 3. Authorisation and configuration phases in ABVS agent system

In the *voting phase*, the agent functions as an intermediary between the voters and the trusted nodes. The agent would compose and sent a request for a voting agent to the nearest trusted node, when the voter enters VIT number into the voting application. Furthermore, when the received voting agent completes its task then the authorization-configuration agent would ensure deletion

of the voting agent. A schema of the voting phase from the point of view of the authorization-configuration agent is presented in Fig. 4.

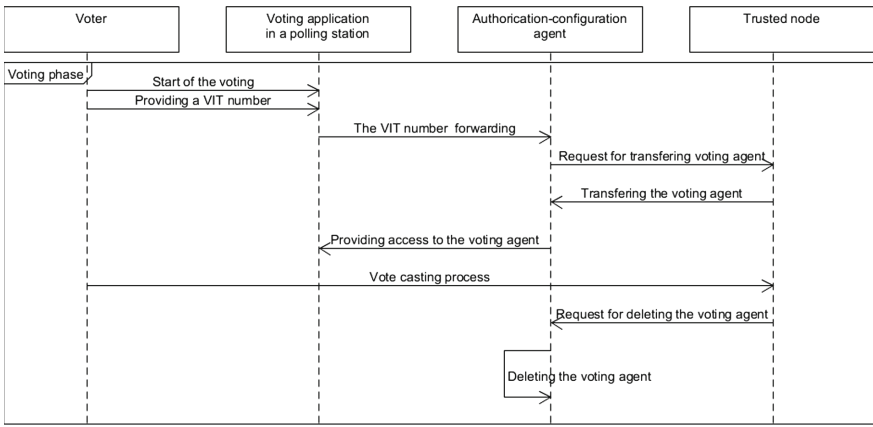


Fig. 4. Voting phase from the point of view of the authorization-configuration agent

The *voting agent* would be tasked with managing the voting process. The agent would provide the voter with a ballot relevant to the given election (via the voting application interface) and transmit the vote with a necessary voting metadata, including: timestamp, VIT, polling station identification data, etc. The voting agent would be removed after the vote has been cast and its place would be taken by the next voting agent when the next voter starts the process. Figure 5 presents a schema of the voting process form the point of view of the voting agent.

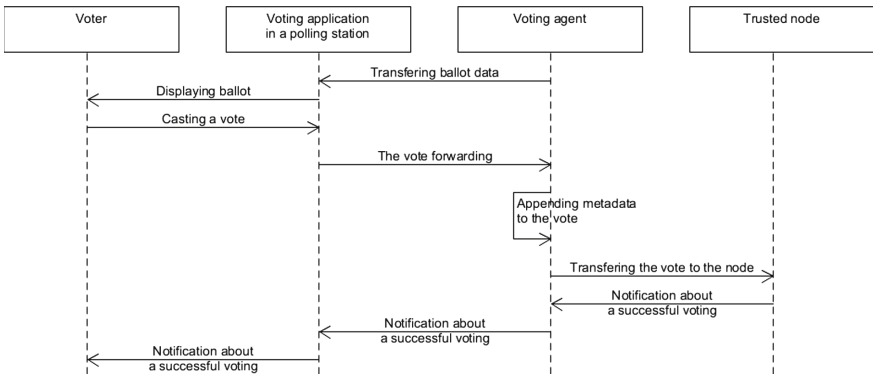


Fig. 5. Voting process from the point of view of the voting agent

4 Conclusions

The presented agent-based approach to electronic voting with ABVS voting system has two main advantages. Firstly, it may significantly increase security of the whole system by reducing the voting applications in the polling station to a role of intermediaries between the voters and the ABVS blockchain network. The responsibility of processing and transmitting the votes would be transferred onto intelligent agents. Furthermore, any attempt of unauthorized modification of the system would be easily detected due to the agents being distributed only by the trusted nodes. Secondly, the agent-based solution would allow using computational resources available at the polling stations for vote processing, which would reduce load on the trusted nodes.

The proposed solution may be implemented using a technology called “smart contracts”, which are digital agreements, scripts and applications originating in the Ethereum blockchain network [7]. Smart contracts enable automatic execution of contract terms without third parties involvement. This technology can be used for implementation of intelligent agents and transmitting them as a form of a blockchain transaction between the voting applications and the trusted nodes.

Overall, this agent-based blockchain based e-voting solution seems to have a lot of research potential. In the future, a working prototype should be implemented and thoroughly tested.

References

1. De Faveri, C., Moreira A., Araújo, J.: Towards security modeling of e-voting systems. In: Proceedings of IEEE 24th International Requirements Engineering Conference Workshops (REW), Beijing, China (2016)
2. Lehoucq, F.: Electoral fraud: causes, types, and consequences. *Ann. Rev. Polit. Sci.* **6**(1), 233–256 (2003)
3. Willemson, J.: Bits or paper: which should get to carry your vote? *J. Inf. Secur. Appl.* **38**, 124–131 (2018)
4. Ojo, A., Adebayo, S.: Blockchain as a next generation government information infrastructure: a review of initiatives in D5 countries. In: Ojo, A., Millard, J. (eds.) *Government 3.0 – Next Generation Government Technology Infrastructure and Services*. PAIT, vol. 32, pp. 283–298. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63743-3_11
5. Enterprise Estonia: Factsheet on Estonian blockchain technology (in English) (2012). <https://e-estonia.com/wp-content/uploads/facts-a4-v03-blockchain.pdf>. Accessed 8 Feb 2018
6. Akwei, I.: Sierra Leone is first country in the world to use blockchain technology to vote, March 2018. <https://face2faceafrica.com/article/sierra-leone-first-country-world-use-blockchain-technology-vote>. Accessed 22 Apr 2018
7. Ethereum Foundation: Ethereum Project. Ethereum Foundation, August 2014. <https://www.ethereum.org/>. Accessed 20 Apr 2018
8. Naidu, P.S., Kharat, R., Tekade, R., Mendhe, P., Magade, V.: E-voting system using visual cryptography & secure multi-party computation. In: Proceedings of International Conference on Computing Communication Control and Automation, India (2016)

9. Wolf, P., Nackerdien, R., Tuccinardi, D.: Introducing electronic voting: essential considerations. International Institute for Democracy and Electoral Assistance, 1 December 2011. <https://www.idea.int/publications/catalogue/introducing-electronic-voting-essential-considerations>. Accessed 22 Jan 2018
10. Ochoa, X., Peláez, E.: Affordable and secure electronic voting for university elections: the SAVE case study. In: Proceedings of 4th International Conference on eDemocracy & eGovernment (ICEDEG), Quito, Ecuador (2017)
11. Caarls, S.: E-voting handbook: Key steps in the implementation of e-enabled elections. Council of Europe, November 2010. <https://www.coe.int/t/dgap/goodgovernance/Activities/E-voting>. Accessed Jan 2018
12. Fouard, L., Duclos, M., Lafourcade, P.: Survey on Electronic Voting Schemes. Project AVOTÉ, University of Grenoble (2017)
13. United Nations Development Programm: Feasibility study on Internet Voting for the Central Electoral Commission of the Republic of Moldova: Report and preliminary roadmap. Central Electoral Commission of the Republic of Moldova, Chisinau (2016)
14. State Electoral Office of Estonia: General Framework of Electronic Voting and Implementation thereof at National Elections in Estonia, 20 June 2017. <https://www.valimised.ee/sites/default/files/uploads/eng/IVXV-UK-1.0-eng.pdf>
15. Noizat, P.: Blockchain electronic vote. In: Handbook of Digital Currency, pp. 453–460. Elsevier Inc. (2015)
16. Sandikkaya, M.T., Orencik, B.: Agent-based offline electronic voting. In: Proceedings of 30th Annual International Computer Software and Applications Conference, Chicago, IL, USA (2006)
17. Opalinski, A.: Integrating web site services into application through user interface. *J. Appl. Comput. Sci.* **22**(1), 137–153 (2014). ISSN 1507–0360
18. Poniszewska-Maranda, A., Gebel, L.: Retrieval and processing of information with the use of multi-agent system. *J. Appl. Comput. Sci.* **24**(2), 17–37 (2016). ISSN 1507–0360
19. Pawlak, M., Guziur, J., Poniszewska-Marańda, A.: Voting process with blockchain technology: auditable blockchain voting system. In: Xhafa, F., Barolli, L., Greguš, M. (eds.) INCoS 2018. LNDECT, vol. 23, pp. 233–244. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-98557-2_21



Signal Reconstruction Using Evolvable Recurrent Neural Networks

Nadia Masood Khan^(✉) and Gul Muhammad Khan

Electrical Engineering Department, University of Engineering and Technology,
Peshawar, Pakistan
{nadiakhan, gk502}@uetpeshawar.edu.pk

Abstract. Accurate reconstruction of under-sampled data plays an important role in wireless transmission of signals. A novel approach to reconstruct randomly missing data based on interpolation and machine learning technique i.e. Cartesian genetic programming evolved recurrent neural network (CGPRNN) is proposed in this research. Although feed-forward Neural networks have been very successful in signal processing fields in general with recurrent neural networks having an edge where system with memory is priority. Recurrent neural networks not only provide non-linearity but also non-Markovian state information. The proposed method is used for reconstruction of lost samples in audio signal which are non-stationary in nature through accurate predication. Simulation results are presented to validate the performance of CGPRNN for accurate reconstruction of distorted signal. The error rate of 12% for 25% missing data and 18% for 50% distorted data has been achieved where the system has low confidence in its predication.

Keywords: Signal reconstruction · Predication · Missing audio samples
Recurrent neural network (RNN) · Cartesian genetic programming (CGP)

1 Introduction

Missing data is part of almost all research areas nowadays i.e. medical imaging and diagnosis, data loss in wireless sensor networks (WSNs) including sensor data, voice data and information transmitted through a communication network. These problems can arise due to different reasons such as low signal to noise ratio (SNR), mishandling and less number of samples. The data may be lost because of external interference. It is very crucial to reconstruct the missing data especially the large scale missing data. Interpolation technique applied together with a machine learning algorithm is nearly the most accurate and stable method for solving missing data problems. Jaques et al. [1] have proposed specialized de-noising auto encoder: the Multimodal Autoencoder (MMAE) to handle missing data from sensors and enhance accuracy of emotion recognition systems. Rzepka et al. [2] have explored shifted Slepian function for reconstructing a continuous time band limited signal from its non-uniform samples. Compressive sensing (CS) is a signal processing method proposed for reduction in usage of resources for signal transmission and storage. CS samples signal at a rate far below the Nyquist sampling rate and hence compress the signal while sampling. However, the algorithms used in past for reconstruction of signal is highly complex and computationally expensive. In this research, lost and damaged samples are estimated in

time domain and original signal is reconstructed. The estimation algorithm is Cartesian genetic programming evolved recurrent neural network (CGPRNN) [3] which proved to perform better than other neuro-evolutionary algorithms.

2 Literature Survey

In signal processing field, reconstruction of the signal from its non-uniformly distributed samples is very important. The proposed research focuses on the importance of using system with memory for reconstruction of a signal. The input signal to CGPRNN reconstruction algorithm is a low pass signal non-uniformly sampled below Nyquist sampling rate. Wang et al. [4] have proposed a sampling theorem in function spaces associated with the linear canonical transform (LCT) for the reconstruction of signal. According to Shannon sampling theorem [5], continuous time signal can be reconstructed from uniform discrete time samples when the signal is band limited. In many practical applications, sampling occurs at non-uniform time instances because of signal lost due to environmental disturbances during transmission. Importance is given to non-uniform sampling as all biomedical i.e. Electrocardiogram (ECG), electroencephalogram (EEG), Electromyogram (EMG) and biological signals are non-uniform in nature. Lagrange interpolation [6] is used for reconstruction of a signal from uniform samples. Maymon et al. [7] presented perfect reconstruction from equally spaced uniform samples through sinc interpolation. Missing portion of data distorted due to impairments such as impulsive noise, clipping, and packet loss in audio signal is recovered using audio inpainting framework in [8]. Maher [9] have proposed spectral extrapolation of missing or corrupted samples in a digital audio data stream using a sinusoidal representation.

Under sampled k space data is used to accurately reconstruct magnetic resonance (MR) images and have improved reconstruction error on the order of 4 to 18 dB [10]. Time series inputs are required from sensors to generate inputs for designing a reconstruction algorithm. DeepSense explored by Yao et al. [11] integrates convolutional and recurrent neural networks for finding exact temporal relationship to model signal dynamics. Cands et al. [12] proposes signal reconstruction from incomplete frequency samples by solving convex optimization problem. Han et al. [13] used non-negative hidden Markov model to estimate missing values of audio signal in time-frequency domain. They have used 12 clips of real-world music recordings and achieved average performance of 15.32 dB. Gabor regression model proposed in [14] for dealing with missing data problem in audio signal has achieved 10 dB SNR for 36.5% and 5.94 dB for 37.5% missing data. Boufounos [15] presented Compressive Sampling Matching Pursuit (CoSaMP) and has reported 12 dB SNR.

3 Ensemble of Cartesian Genetic Programming and Recurrent Neural Network (CGPRNN)

Cartesian genetic programming (CGP) [16] encoding scheme is used to represent the RNN structure and connectivity. RNN uses output of the previous time step in the training phase allowing past information to be retained through a feedback loop. The

knowledge from the previous data helps in better training of network. The performance error is provided in run time to the network to adjust weights for better results. Feed forward networks are the most simplest type of neural network in which information flows in forward direction. Whereas RNN have a different architecture where the information flows through the system constitute a cycle and can store previous data in the internal memory of the network. RNN can thus be referred to as a system with memory ideal for non-Markovian systems providing better results than feed forward networks [3]. RNNs are especially useful when we have sequential data because each neuron or unit can use its internal memory to maintain information about the previous input. The proposed architecture can reduce computational complexity and produce variable length networks with optimal number of neurons.

4 Experimental Setup

In the experiments, training has been performed on 10% of dataset and the remaining 90% is used for testing. The training has been done in several situations, varying the number of iterations and network nodes. CGPRNN is dependent on system inputs as well as recurrent inputs and can efficiently learn from the data when some of the inputs are noisy. After training, CGPRNN is used to predict the missing values in time domain. Figure 1 represents the CGPRNN system lay-out. Neural network inputs are passed through a feed forward Cartesian Genetic Programming Artificial Neural Network (CGPANN) which generates output at time step $t-1$. The hidden layer of the network contains the memory of the network keeping a track of all the previous time steps. The output at time $t-1$ is calculated solely based on the inputs at time $t-1$. The output ($t-1$) is multiplied with recurrent weights and is fed from the output layer as a recurrent input to the network making a closed loop. The feed back loop adjusts the weights based on the error rate and improves the performance of the network. In this method, missing sample prediction is done on the assumption that previous samples appears at time $t-1$ and the prediction target of interest is at time t given by network output $Y(t)$. The experiments are repeated several times with pseudo random initialization of the parameters in CGPRNN. To indicate missing values in the data, we consider zero imputation in which missing values in each time series are replaced by zero [17]. Signal reconstruction algorithm based on CGP evolved RNN has been proposed to handle audio signals with missing samples. The trained CGPRNN is used for predicating the target missing samples. The audio signals used in the experimentation is recorded using personal computer (PC) sound recorder and is saved as waveform audio (.wav) format having sampling frequency of 44.1 kHz. We have considered application of CGPRNN for two cases:

1. 50% missing data
2. 25% missing data.

Moving window approach has been used for first case i.e. 50% missing data and jumping window mechanism has been used for 25% missing data problem which are explained in detail in the following sections.

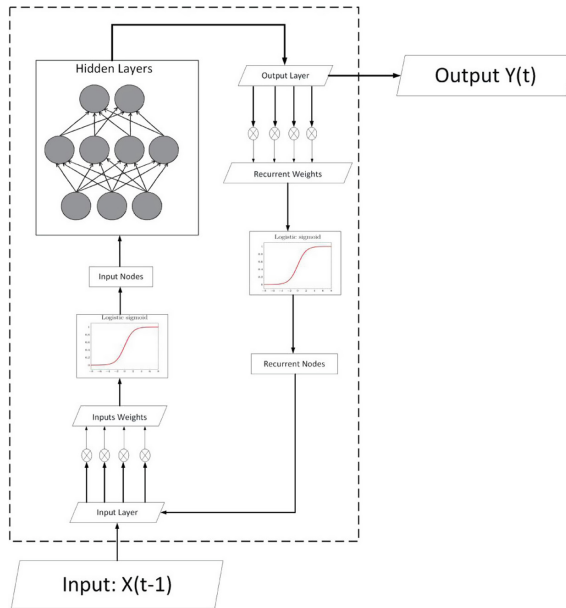


Fig. 1. CGPRNN system architecture

4.1 Moving/Sliding Window Approach

Moving window approach is used when there are consecutive missing samples. Half of the signal samples lost are considered as 50% missing data. As shown in Fig. 2A, first five inputs are used to predict the 6th sample. The window then moves forward and predicts the next missing sample based on previous four inputs and one predicted sample. In similar manner, whole segment of missing data is generated. Sliding window mechanism allows an unlimited number of samples to be estimated.

4.2 Jumping Window Approach

Jumping window approach is applied on 25% missing data case. As 25% missing data does not have consecutive missing samples so jumping window predict one value and then jump to predict the next missing sample as shown in Fig. 2B.

One sample lost out of 4 samples is considered as 25% missing data. The original 3 samples are used to estimate the lost sample.

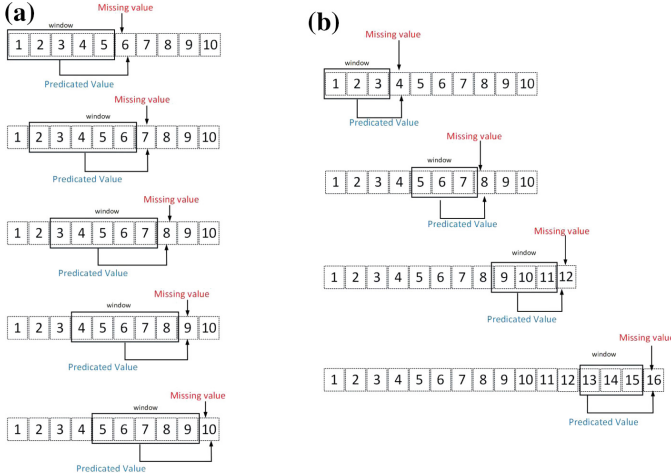


Fig. 2 A. Sliding window mechanism. B. Jumping window mechanism

5 Result and Analysis

In our experiments, we have used two criteria for performance evaluation. The first evaluation criterion is the signal to noise ratio (SNR), which is computed as follows,

$$SNR = 20 \log_{10} \frac{\sum_{i=1}^k |S_i|}{\sum_{i=1}^k |S_i^T - S_i^P|}$$

SNR is ratio of signal to noise, noise is represented as difference between the predicted value and the actual value (target) of missing sample. S_i represents the total signal, S_i^T represents target value and S_i^P shows the predicted value of missing sample. Results achieved are reasonably competitive when SNR is high which shows signal strength is better than noise. The second evaluation criteria is mean absolute percentage error (MAPE) which is computed using the following equation:

$$MAPE = \frac{1}{n} \sum_{t=1}^n \left| \frac{e_t}{y_t} \right| * 100\%$$

e_t shows the error between original and estimated signal sample and y_t depicts the actual data. Lower MAPE value represents better estimation.

The SNR obtained after estimation of missing samples for the case of 25% missing data is shown in Table 1. The highest SNR of **40.67** dB is achieved for audio signal 9 having highest standard deviation of 8.21 and overall 50-node network have performed well showing SNR of 37.54 dB, which is good because less number of nodes means less evolution time. Average SNR of **31.54** is obtained when 25% data is missing. The mean absolute percentage error of **12.54%** is recorded for the 25% missing samples as evident from Table 2. The moving window mechanism is used for 50% missing data

Table 1. Average SNR in dB for 25% missing data

S. No.	Number of nodes					Average	STDEV
	50	100	150	200	250		
Audio signal 2	37.81	42.97	34.46	41.59	43.35	40.04	3.81
Audio signal 3	37.61	43.49	34.59	41.99	42.40	40.02	3.77
Audio signal 4	36.88	21.85	20.99	15.12	19.88	22.94	8.21
Audio signal 5	32.71	19.41	17.81	22.24	20.88	22.61	5.88
Audio signal 6	36.81	26.12	20.75	29.32	26.14	27.83	5.89
Audio signal 7	38.16	25.50	18.71	21.85	22.14	25.27	7.60
Audio signal 8	33.00	31.19	35.79	32.92	31.63	32.91	1.79
Audio signal 9	47.34	40.22	40.63	36.27	38.88	40.67	4.10
Average	37.54	31.34	27.97	30.16	30.66	31.54	

Table 2. MAPE values of 25% missing data

S. No.	Number of nodes					Average	STDEV
	50	100	150	200	250		
Audio signal 2	10.16	8.97	11.50	9.25	8.98	9.77	1.08
Audio signal 3	8.12	6.37	10.11	7.48	4.86	7.39	1.96
Audio signal 4	17.27	16.14	15.91	23.80	8.42	16.31	5.46
Audio signal 5	18.58	16.87	17.88	15.17	7.22	15.14	4.61
Audio signal 6	14.39	13.02	16.40	14.79	7.06	13.13	3.60
Audio signal 7	19.57	18.88	20.82	21.09	8.46	17.76	5.28
Audio signal 8	14.96	11.76	10.58	11.04	8.85	11.44	2.24
Audio signal 9	9.54	10.67	9.80	12.80	8.24	10.21	1.69
Average	14.07	12.83	14.12	14.43	7.76	12.54	

problem and SNR reported is shown in Table 3. Audio signal 4 having high standard deviation of 4.31 indicates spread out of data points over a wider range of values and the proposed system still shown best SNR of 31.99 dB. Overall 200-node network have performed well (27.62) in estimating multiple missing data. Table 4 shows MAPE values obtained for 50% missing data. High deviation from mean value in case of audio signal 4 has shown optimal error of 26.51. Average MAPE of **18.70%** and average SNR of **27.39** dB is recorded using CGPRNN for 50% missing data. As complexity of problem is unknown so different number of nodes are evaluated to make sure that the network is not stuck in local optimum but has achieved global optimum. From experimentation it is confirmed that same network is achieved irrespective of initial number of nodes so computational cost remains same. Comparison of proposed CGPRNN with other algorithms is shown in Table 5 which depicts SNR improvement achieved using different algorithms on similar types of audio datasets. Higher SNR and lower MAPE value represents better quality of reconstructed signal. From Table 5, it is evident that CGPRNN outperforms all other contemporary techniques in terms of SNR.

Table 3. Average SNR in dB 50% missing data

S. No.	Number of nodes					Average	STDEV
	50	100	150	200	250		
Audio signal 2	20.41	20.05	20.17	20.47	21.77	20.57	0.69
Audio signal 3	26.73	26.52	26.83	27.22	27.93	27.04	0.56
Audio signal 4	33.92	33.49	34.54	33.70	24.30	31.99	4.31
Audio signal 5	25.11	24.99	24.80	25.10	26.78	25.36	0.80
Audio signal 6	28.75	28.71	28.82	29.13	29.27	28.94	0.25
Audio signal 7	25.98	26.04	26.07	26.19	26.26	26.11	0.11
Audio signal 8	29.83	29.11	29.60	30.03	31.35	29.99	0.84
Audio signal 9	29.22	29.18	29.27	29.13	28.99	29.16	0.11
Average	27.49	27.26	27.51	27.62	27.08	27.39	

Table 4. MAPE values of 50% missing data

S. No.	Number of nodes					Average	STDEV
	50	100	150	200	250		
Audio signal 2	21.90	22.62	22.40	21.85	19.38	21.63	1.30
Audio signal 3	11.08	11.26	11.00	10.64	10.05	10.81	0.48
Audio signal 4	24.11	24.09	24.36	24.73	35.26	26.51	4.90
Audio signal 5	22.08	22.15	22.66	22.66	21.90	22.29	0.35
Audio signal 6	19.27	18.67	18.69	18.80	19.89	19.07	0.52
Audio signal 7	22.22	22.08	22.32	22.64	23.58	22.57	0.60
Audio signal 8	13.03	13.38	13.19	13.02	12.63	13.05	0.28
Audio signal 9	13.74	13.85	13.75	13.61	13.36	13.66	0.19
Average	18.43	18.51	18.55	18.49	19.51	18.70	

Table 5. Comparison with other techniques

S. No	Technique	SNR Improvement/Gain (dB)
1	Gabor regression model [14]	10 (36.5% missing data)
2	Gabor regression model [14]	5.94 (37.5% missing data)
3	Matched sign pursuit [15]	12 (Random noise)
4	Non-negative hidden Markov model [13]	15.32 (Random missing data)
5	Proposed CGPRNN	31.54 (25% missing data)
6	Proposed CGPRNN	27.39 (50% missing data)

6 Conclusion and Future Work

The intelligent signal processing method presented here uses a combination of Cartesian genetic programming and recurrent neural networks to devise an efficient non-markovian system for accurate signal reconstruction. Missing data is a part of almost all research area, and there are several ways for restoration of missing points.

An advantage of using the proposed approach is that it can reconstruct an under sampled and band limited signal in time domain ideally which makes it much more compatible than other traditional algorithms. Simulation results shows that CGPRNN has achieved accuracy of 81% when half of the signal data is missing and 87% when 25% data was missing. In future work, we will try to design deep recurrent neural network evolved by Cartesian genetic programming for further improving the performance of missing data prediction and also exploration of current algorithm on high frequency video data streams.

References

1. Jaques, N., Taylor, S., Sano, A., Picard, R.: Multimodal autoencoder: a deep learning approach to filling in missing sensor data and enabling better mood prediction. In: 2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII), pp. 202–208. IEEE (2017)
2. Rzepka, D., Miśkiewicz, M.: Fast reconstruction of non-uniformly sampled bandlimited signal using Slepian functions. In: 2014 Proceedings of the 22nd European Signal Processing Conference (EU-SIPCO), pp. 741–745. IEEE (2014)
3. Khan, G.M., Zafari, F., Mahmud, S.A.: Very short term load forecasting using Cartesian genetic programming evolved recurrent neural networks (CGPRNN). In: 2013 12th International Conference on Machine Learning and Applications (ICMLA), vol. 2, pp. 152–155. IEEE (2013)
4. Wang, J., Ren, S., Chen, Z., Wang, W.: Periodically nonuniform sampling and reconstruction of signals in function spaces associated with the linear canonical transform. *IEEE Commun. Lett.* **22**, 756–759 (2018)
5. Shannon, C.E.: Communication in the presence of noise. *Proc. IRE* **37**(1), 10–21 (1949)
6. Sauer, T., Xu, Y.: On multivariate lagrange interpolation. *Math. Comput.* **64**(211), 1147–1170 (1995)
7. Maymon, S., Oppenheim, A.V.: Sinc interpolation of nonuniform samples. *IEEE Trans. Sig. Process.* **59**(10), 4745–4758 (2011)
8. Adler, A., Emiya, V., Jafari, M.G., Elad, M., Gribonval, R., Plumbley, M.D.: Audio inpainting. *IEEE Trans. Audio Speech Lang. Process.* **20**(3), 922–932 (2012)
9. Maher, R.C.: A method for extrapolation of missing digital audio data. *J. Audio Eng. Soc.* **42** (5), 350–357 (1994)
10. Ravishankar, S., Bresler, Y.: MR image reconstruction from highly under sampled k-space data by dictionary learning. *IEEE Trans. Med. Imaging* **30**(5), 1028–1041 (2011)
11. Yao, S., Hu, S., Zhao, Y., Zhang, A. and Abdelzaher, T.: Deepsense: a unified deep learning framework for time-series mobile sensing data processing. In: Proceedings of the 26th International Conference on World Wide Web, pp. 351–360. International World Wide Web Conferences Steering Committee (2017)
12. Candès, E.J., Romberg, J., Tao, T.: Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inf. Theor.* **52**(2), 489–509 (2006)
13. Han, J., Mysore, G.J., Pardo, B.: Audio imputation using the non-negative hidden Markov model. In: Theis, F., Cichocki, A., Yeredor, A., Zibulevsky, M. (eds.) *LVA/ICA 2012*. LNCS, vol. 7191, pp. 347–355. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28551-6_43

14. Wolfe, P.J., Godsill, S.J.: Interpolation of missing data values for audio signal restoration using a Gabor regression model. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2005, vol. 5, pp. v–517. IEEE (2005)
15. Boufounos, P.T.: Greedy sparse signal reconstruction from sign measurements. In: 2009 Conference Record of the Forty-Third Asilomar Conference on Signals, Systems and Computers, pp. 1305–1309. IEEE (2009)
16. Miller, J.F.: Cartesian genetic programming. In: Miller, J. (ed.) Cartesian Genetic Programming. Natural Computing Series. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-17310-3_2
17. Bennett, D.A.: How can I deal with missing data in my study? *Aust. N.Z. J. Public Health* **25** (5), 464–469 (2001)



A Cluster-Based Prototype Reduction for Online Classification

Kemilly Dearo Garcia^{1,2}, André C. P. L. F. de Carvalho² ,
and João Mendes-Moreira^{3,4} 

¹ University of Twente, Enschede, Netherlands

² ICMC, University of São Paulo, São Paulo, Brazil
kemilly.dearo@usp.br, andre@icmc.usp.br

³ Faculty of Engineering, University of Porto, Porto, Portugal
jmoreira@fe.up.pt

⁴ LIAAD-INESC TEC, Porto, Portugal

Abstract. Data stream is a challenging research topic in which data can continuously arrive with a probability distribution that may change over time. Depending on the changes in the data distribution, different phenomena can occur, for example, a concept drift. A concept drift occurs when the concepts associated with a dataset change when new data arrive. This paper proposes a new method based on k -Nearest Neighbors that implements a sliding window requiring less instances stored for training than existing methods. For such, a clustering approach is used to summarize data by placing labeled instances considered similar in the same cluster. Besides, instances close to the uncertainty border of existing classes are also stored, in a sliding window, to adapt the model to concept drift. The proposed method is experimentally compared with state-of-the-art classifiers from the data stream literature, regarding accuracy and processing time. According to the experimental results, the proposed method has better accuracy and less time consumption when fewer information about the concepts are stored in a single sliding window.

Keywords: k NN Prototyping · Data stream · Online clustering

1 Introduction

In real world data analysis, data can continuously arrive in streams, with a probability distribution that can change over time. These data are known as data streams. Depending on the changes in the data distribution, different phenomena can occur, like concept drift [8]. In these situations, it is important to adapt the

This work was partially funded by the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia within project POCI-01-0145-FEDER-016883.

classification model to the current stream, otherwise its predictive performance can decrease over time.

Several algorithms proposed for data stream mining are based on online learning [4, 7–9]. Some of them are based on the k NN (k -Nearest Neighbor) algorithm. In the data stream mining, the k NN algorithm maintains a sliding window with a certain amount of labeled data, which is used as its training data.

Other algorithms from the literature deal with concept drift by explicitly detecting changes in parts of the stream, comparing the current concept with previous concepts from time to time [9]. Some of them continuously calculate the model classification error. For such, they assume that the label of the data arriving in the stream is available.

However, there is a cost associated with the data labeling process that can become prohibitive or unfeasible when data arrive in high speed or volume. In online classification, the labeling of incoming instances can have a high cost [11]. The lack of the label make the measure of classification error a difficult task.

Despite its simplicity, k NN has been largely used in literature, because it is nonparametric, favoring its use in scenarios with few available information and with known concepts changing over time [4]. However, the use of a sliding window may ignore instances with relevant information about persistent concepts. Furthermore, the size of a sliding window affects its efficient use.

This article proposes SWC (Sliding Window Clusters), a method based on k NN that implements a sliding window whose number of instances stored can be reduced. SWC summarizes data streams by creating a set of clusters, each one representing similar labeled instances. Instances close to decision border of each cluster are also stored, so they can be used to adapt the model to concept drift.

An experimental evaluation shows that SWC can increase the predictive performance and reduce both computational and time consumption than related methods based on k NN and sliding window.

This paper is structured as follows. Section 2, presents previous related works using k NN and sliding window. Data stream and concept drift are introduced in Sect. 3. The proposed method is described in Sect. 4. Sections 5 presents the experimental setup and analyses the results obtained. Finally, Sect. 6 has main conclusions and points out future work directions.

2 Related Work

This section briefly presents previous works using k NN for data stream classification with concept drift. These works use variations of sliding window to store the training instances.

One alternative of online learning is to randomly select instances to maintain or discard in the sliding window. This is the case of the method PAW (Probabilistic Approximate Window) method [4], a probabilistic measure used to decide which instance will be discarded from the sliding window when a new instance arrive. Thus, the size of the window is variant and represents a mix of outdated

and recent relevant instances. The $k\text{NN}_W$ method combines the PAW method couplet with the $k\text{NN}$ classifier.

Another related method, ADWIN (ADaptive sliding WINdowing) [2], is a concept drift tracker able to monitor changes in data streams. The algorithm automatically grows the sliding window when no change is detected in the stream. When a change is detected, the algorithm shrinks the sliding window and forgets the sub-window that is outdated. In the combination of $k\text{NN}$ with PAW and ADWIN [4], $k\text{NN}_{WA}$, ADWIN is used to keep only the data related to the most recent concept from the stream, the rest of instances are discarded.

A deficiency of updating instances using a sliding window is the possibility to forget old but relevant information. To avoid losing relevant information, another method, named SAM (Self Adjusting Memory) [9], to adapt a model to concept drift by explicitly separating current and past information. SAM uses two memory structures to store information, one based on short-term-memory and the other on long-term-memory. The short-term-memory contains data associated with the current concept and the long-term-memory maintains knowledge (old models) from past concepts. SAM is couplet with a $k\text{NN}$ classifier.

The implementations and variations of $k\text{NN}$ for data stream mining are available in the MOA framework. Due to memory and computational limitations, the implementations use a fixed size window of 1000 labeled instances.

3 Problem Formalization

A possible unbounded amount of data can sequentially arrive in a data stream. These data can often undergo changes in their distribution over time, which may require the adaptation of the current model context [8].

Formally, a data stream is a sequence of instances, potentially infinity that can be represented by [7]:

$$D_{tr} = \{(X_1, y_1), (X_2, y_2), \dots, (X_{tr}, y_{tr})\}$$

where X_{tr} is an instance arriving in time tr and y_{tr} is the target class. Each instance needs to be processed only once due to finite resources.

Concept drift is a change in the distribution probability of target classes [8]. Formally, a distribution P in a given time tr conditioned by the instance X and label y can suffer changes affecting the conditional probability $P_{tr+1}(X, y)$. As a result, a model built during time tr could be outdated in time $tr + 1$.

$$X : P_{tr}(X, y) \neq P_{tr+1}(X, y)$$

4 Methodology

In data stream mining, an ideal classifier should be able to learn the current concept in feasible time without forgetting relevant past information [4].

The proposed method is described in Algorithm 1. Instead of storing all instances that fit in a sliding window (for representing both old and current

Algorithm 1. SWC: Online Window Update

```

1: input:  $X_{tr}$ ,  $W$ ,  $T$ ,  $\rho$ 
2: output:  $W$ 
3:  $rand \leftarrow random(0, 1)$ 
4: if  $rand \leq \rho$  then
5:   for all  $w$  in  $W$  do
6:     Let  $X_{tr}$  be the nearest to  $w$  ( $w \in W$ )
7:      $dist \leftarrow EuclidianDistance(X_{tr}, w)$ 
8:     if  $dist < w_{radius}$  then
9:        $W \leftarrow UpdateCluster(X_{tr}, w)$ 
10:    else
11:      if  $dist \leq T$  then
12:         $W \leftarrow W \cup X_{tr}$ 
return  $W$ 

```

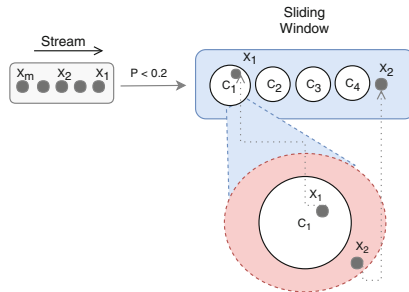


Fig. 1. Instances X_1 and X_2 are stored within the sliding window. The first instance, X_1 , is closer to cluster C_1 and inside its radius. The second instance, X_2 , is outside cluster area, but close to the uncertainty border.

concepts), SWC stores compressed information about concepts and instances close to uncertainty border of each class. As the previous methods, SWC is combined with the k NN classifier in the MOA framework [3].

A more detailed description of how SWC works is presented next. Initially, all instances arriving from the stream are stored in the form of clusters. The clusters are created using the CluStream algorithm [1]. A constrain implying that each cluster must contain only instances from the same class was included.

As data arrive in the stream, a parameter based on probability, ρ , is used to decide if a new instance, X_{tr} , will be incorporated to the model W . If X_{tr} is inside a radius from a existing cluster, the instance is incorporated to this cluster. However, if X_{tr} is outside, but is close to a uncertainty border, X_{tr} is incorporated to the model alone, outside the existing clusters. For such, the uncertainty border is defined as the area outside the radius of a cluster, but inside a given threshold.

As is illustrated in Fig. 1, if the instance, X_1 , is inside the radius of the closest cluster, then it will be incorporated to the existing cluster, however if the instance, X_2 , is closer to a uncertainty border, it is stored alone.

It must be observed that not all instances in the stream are included into the sliding window. For each instance arriving in the stream, SWC randomly decides if the instance will be learned or not. A similar procedure is used in [4], which uses a probability $\rho = 0.5$. SWC uses a lower probability, consider that the learning process can be done with a lower probability of $\rho = 0.2$, without significant predictive performance loss, but with a lower processing cost.

5 Experimental Evaluation

This section experimentally compares SWC with other methods implemented in the MOA framework that use k NN with sliding window, namely k NN, k NN_W, k NN_{WA} and SAM. The experimental evaluated used was Interleaved Test-Train to incremental learning [4].

5.1 Datasets

Table 1 describes the datasets used in the experiments. Before the streaming, in a offline phase, all methods started with a batch of labeled data representing 10% of the each dataset. The remaining data arrived in the stream. Real and artificial datasets were used.

Table 1. Characteristics of datasets evaluated.

Datasets	Samples	Features	Class
SEA	5.000/50.000 (total)	3	2
Mixed drift	60.000/600.000 (total)	2	15
Rotating hyperplane	20.000/200.000 (total)	10	2
Forest cover type	58.101/581.012 (total)	54	7
Airlines	53.938/539.383 (total)	4	2
Moving RBF	20.000/200.000 (total)	10	5

Artificial Datasets

The SEA Concepts Dataset [10] has four concepts. A concept drift occurs at each 15.000 instances, with different thresholds for the concept.

The Rotating Hyperplane dataset is based on a hyperplane of d -dimensional space which is continuously changing in position and orientation. It is available in the MOA framework and was used in [4,9].

Moving RBF is a dataset, generated by MOA framework, based on Gaussian distributions with random initial positions, weights and standard deviations. Over time, this Gaussian distributions suffer changes. This dataset is used by [4,9].

Mixed Drift [4,9] is a mix of tree datasets: Interchanging RBF, Moving Squares and Transient Chessboard. Data from each dataset are alternatively presented in the stream.

Real World Datasets

The Forest Cover Type [6] data set is a well known benchmark for the evaluation of algorithms for data stream mining, being constantly used to validate proposed methods [4, 9, 11].

The Airlines dataset has data from US flight control [11]. It has two classes, one indicating that a flight will be delayed, and the other that the flight will arrive on time.

5.2 Results and Discussion

The proposed method, SWC, is compared with the methods k NN, k NN_W, k NN_{WA} and SAM. For all methods, one nearest neighbor ($k = 1$) is adopted. The remaining parameters use default values, including a fixed window size ($w = 1000$).

The ρ parameter, chance of updating the model, in the SWC method is defined for an acceptable trade-off between accuracy and time cost. A parameter of threshold $T = 1.1$, uncertainty border, is also defined for each cluster. The threshold is multiplied by the radius of each cluster and indicates how much the cluster can expand. Both parameters were explained in Sect. 4.

Experiments were performed to decide the value of ρ and for SWC. Figure 2 shows that there is an increase of accuracy with $\rho = 0.5$, meaning that a instance has 50% of chance to be learned by the model. However, the selected value was $\rho = 0.2$, which results in a better balance between accuracy and time cost.

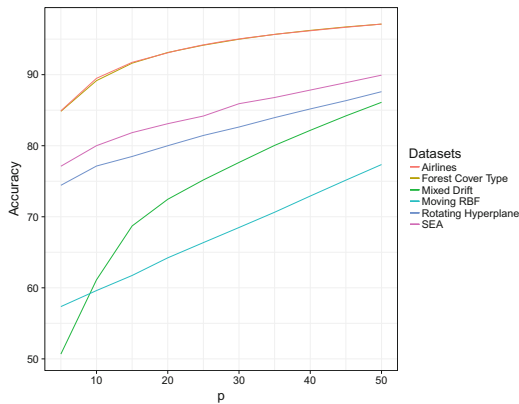


Fig. 2. SWC accuracy performance of all datasets varying ρ value in 5% to 50%.

Table 2 shows the average accuracy and total time cost. It must be observed that accuracy is the measure of instances correctly classified over test/train interleaved evaluation [4].

The results shows that SWC is competitive with state-of-the-art SAM and is considerably faster. The method baseline k NN presented the worst performance, which was expected, once it does not learn over time. However, it is a good baseline to measure how much time each other method take to learn new instances.

Both methods k NN_W and k NN_{WA} present similar accuracy rates. However, k NN_{WA} has a higher cost, due the use of ADWIN.

Finally, although SAM and SWC obtained predictive accuracies similar to SWC, for some cases, SWC was better. Besides, SWC is faster, due to the use of only one sliding window with compressed concepts and relevant instances.

Table 2. Accuracy and time cost (in seconds) for each method.

Dataset	k NN	k NN _W	k NN _{WA}	SAM	SWC
SEA	77.24 (6)	77.25 (9)	77.25 (10)	80.53 (18)	83.49 (8)
Mixed Drift	16.82 (68)	53.62 (94)	53.62 (102)	80.53 (2724)	72.46 (73)
Rotating Hyperplane	50.00 (63)	66.42 (88)	68.42 (91)	70.27 (318)	80.17 (70)
Forest Cover Type	23.46 (614)	54.56 (898)	55.56 (1024)	89.84 (3422)	93.12 (394)
Airlines	54.37 (91)	52.53 (163)	52.53 (146)	88.37 (1530)	93.07 (120)
Moving RBF	26.07 (62)	59.98 (89)	59.97 (100)	69.92 (1788)	64.22 (71)

To assess their statistical significance, a Friedman rank sum test combined with Nemenyi post-hoc test [5], both with a significance level of 5%, was applied to the experimental results. A $p - value = 0.000441$ was obtained in the Friedman test, showing a significant difference between the five methods. Additionally, the Nemenyi post-hoc test, Table 3, showed meaningful statistical differences between the following pair of methods: SWC > k NN. There is no significant difference between all remaining pairs. However we emphasize that SWC > k NN_W and SWC > k NN_{WA} have relatively low p-values (less than 10%).

Table 3. P-values obtained for the multiple comparison post-hoc Nemenyi test.

	k NN	k NN _W	k NN _{WA}	SAM
k NN _W	0.8536	-	-	-
k NN _{WA}	0.7591	0.9998	-	-
SAM	0.0090	0.1506	0.2201	-
SWC	0.0024	0.0621	0.0987	0.9962

6 Conclusion and Future Work

This paper presented a new method, SWC, based on k -Nearest Neighbors that implements a sliding window that stores less training instances than related methods. SWC stores in a sliding window clusters and instances close to uncertainty border of each class. The clusters are compressed stable concepts and the instances are possible drifts of these concepts.

Considering accuracy performance, time and storage cost, SWC was experimentally compared with state-of-the-art related methods. According to the experimental results SWC presented higher predictive performance, with lower processing and memory cost than the compared methods.

As future work, the authors want to distinguish concept drift from novelty detection and study an efficient alternative to discard outdated information. Besides, they intend to include an unsupervised concept drift tracker.

References

1. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: Proceedings of 29th International Conference on Very Large Data Bases, VLDB 2003, 9–12 September 2003, Berlin, Germany, pp. 81–92 (2003)
2. Bifet, A., Gavaldà, R.: Learning from time-changing data with adaptive windowing. In: Proceedings of the Seventh SIAM International Conference on Data Mining, 26–28 April 2007, Minneapolis, Minnesota, USA, pp. 443–448 (2007)
3. Bifet, A., Holmes, G., Kirkby, R., Pfahringer, B.: MOA: massive online analysis. *J. Mach. Learn. Res.* **11**, 1601–1604 (2010)
4. Bifet, A., Pfahringer, B., Read, J., Holmes, G.: Efficient data stream classification via probabilistic adaptive windows. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC 2013, Coimbra, Portugal, 18–22 March 2013, pp. 801–806 (2013)
5. Demsar, J.: Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **7**, 1–30 (2006)
6. Dheeru, D., Karra Taniskidou, E.: UCI machine learning repository (2017). <http://archive.ics.uci.edu/ml>
7. Faria, E.R., Gama, J., Carvalho, A.C.P.L.F.: Novelty detection algorithm for data streams multi-class problems. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC '13, Coimbra, Portugal, 18–22 March 2013, pp. 795–800 (2013)
8. Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. *ACM Comput. Surv.* **46**(4), 44:1–44:37 (2014)
9. Losing, V., Hammer, B., Wersing, H.: KNN classifier with self adjusting memory for heterogeneous concept drift. In: IEEE 16th International Conference on Data Mining, ICDM 2016, 12–15 December 2016, Barcelona, Spain, pp. 291–300 (2016). <https://doi.org/10.1109/ICDM.2016.0040>
10. 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, San Francisco, CA, USA, 26–29 August 2001, pp. 377–382 (2001)
11. Zliobaite, I., Bifet, A., Pfahringer, B., Holmes, G.: Active learning with drifting streaming data. *IEEE Trans. Neural Netw. Learning Syst.* **25**(1), 27–39 (2014)



Reusable Big Data System for Industrial Data Mining - A Case Study on Anomaly Detection in Chemical Plants

Reuben Borrison^{1(✉)}, Benjamin Klöpper^{1(✉)}, Moncef Chioua^{1(✉)},
Marcel Dix^{1(✉)}, and Barbara Sprick^{2(✉)}

¹ ABB Corporate Research Center Germany, Ladenburg, Germany
{reuben.borrison,benjamin.kloeppe,moncef.chioua,marcel.dix}@de.abb.com

² SRH Hochschule Heidelberg, Heidelberg, Germany
Barbara.Sprick@srh.de

Abstract. Industrial data mining projects in general and big data mining projects in particular suffer from long project execution. The resulting high costs render many interesting use cases otherwise economically unattractive. This contribution shows on the example of anomaly detection for process plants, how the major obstacles - namely the inefficient development tools for Big Data Frameworks like Apache Hadoop and Spark and the lack of reuse of software artifacts across different projects can be overcome. This is achieved by selecting an application case that shares considerable commonalities across different projects and providing a supported project workflow implemented in a scalable and extensible big data architecture.

Keywords: Big data · Data mining · Software architecture

1 Introduction

Despite the current hype, (big) data mining is hardly affecting the industrial practice of process plant operations. Industrial data mining projects in general suffer from long project execution time making many otherwise interesting projects economically unattractive. The objective of the public-funded project FEE^{1,2} is the development of assistance systems to support plant operators. To achieve these goals, the project team analysed heterogeneous data from process plants using big data technologies, for instance, to warn plant operators at an

¹ The underlying project of this contribution was sponsored by the German Federal Ministry of Education and Research (Bundesministerium für Bildung und Forschung, BMBF), under the sponsorship reference number 01IS14006. The authors are responsible for the contents of this contribution.

² Development partners of the project are University of Kassel, Technical University of Dresden and RapidMiner GmbH. Reuben Borrison did his master thesis at the ABB Corporate Research Center Germany. Barbara Sprick was his University Supervisor.

early stage of a critical situation. In critical situations currently, operators may be overloaded with information (e.g. alarm flood) regarding process dynamics, which may result in loss of control of the situation. This can be catastrophic in terms of damage to assets, costly production downtime, and the endangerment of human life and the environment. Early detection and warning of critical situations should allow the operator to make proactive instead of reactive actions to prevent catastrophe.

A major challenge in providing such solutions is the time required to explore, understand and prepare the data required to experiment with different data modeling techniques that can provide the desired assistance functionality. Two major drivers behind these challenges, experienced while providing such solutions, are the inefficient development tools for big data mining stacks like Hadoop and Apache Spark and the lack of reuse of software artifacts across different projects. This paper shows how software artifacts can be created in a Big Data system with a clearly defined scope that applies to many different projects with a common goal. The main contribution of the paper, therefore, consists of the architecture and implementation of a big data processing system that realizes an analytics workflow for anomaly detection that can be reused across different plants.

1.1 Illustrative Example

This section introduces an illustrative example with two main personas: Tim, a plant engineer and Bob, a data scientist. Tim is in charge of smooth operation in a chemical production site. The site consists of several plants for which plant operators handle the day to day operation. Repeatedly after analyzing a difficult situation that lead to issues like lost production, increased energy consumption, quality problems or even plant shutdowns, Tim realizes that he could have prevented the situation if he had been aware early enough. After reading about big data and data mining, Tim approaches Bob and asks if Bob can provide help. Bob immediately realizes that the problem is a perfect match for anomaly detection and starts working. Bob is always on the cutting edge of data mining and big data and uses notebook technologies like Jupyter Notebooks or Apache Zeppelin in combination with Java libraries to work on the problem on a big data cluster (e.g. c.f. [6] for such a setup). He explores and cleanses the data, formats it to match the requirements of the selected anomaly detection algorithm and evaluates the results. At the end Bob is able to provide an algorithm that is able to detect anomalies reliably with not too many false positives. Tim is happy and Tim and Bob prepare a presentation for plant engineering conference. Everyone is impressed with the results and other chemical plants approach Bob to provide them with a similar anomaly detection solution. When receiving first data samples from the new potential customers, Bob discovers that his code and his notebooks are tailored specifically to the properties of Tim's plant and that he needs to start over again for each and every customer. He realizes the need for an adaptable system that can be reused across different plants in order to make anomaly detection for chemical industries more efficient.

1.2 Structure of the Paper

The paper starts with describing the problem setting and the resulting requirements, Sect. 3 gives an overview of related work, Sect. 4 describes the resulting application workflow, system architecture and the selected technology for its implementation, and Sect. 5 evaluates the systems with experimental results and user study. The paper ends with the conclusion and discussed further work.

2 Problem Setting and Requirements Collection Process

Our study of a big data architecture for anomaly detection in chemical plants started with interviewing the intended users for the application: data scientists and plant engineers. The intended data scientists were presented with a set of questions in the form of an online survey. Their responses were thus collected and the requirements analyzed from them. For plant engineers, telephone interviews were organized, where their responses to a set of questions were recorded and requirements analyzed from the same.

The dataset used in this study was historic data from two chemical plants. This was done so as to investigate different formats of datasets used across the industry. This was also helpful in deciding how we can cater to different formats of datasets in the system. The chemical plants taken into consideration are continuous processing plants, which, as the name suggests, runs in an uninterrupted fashion and is operated in more or less constant or optimal state. The sensors located at the process plant generate data which is captured and stored at a particular time interval. The time interval, or sampling rate, of the datasets considered in this study was 1 min. The number of sensors in the dataset were approximately 1300 which was a subset of the sensor located in a specific section of the plant. One of the chemical plants involved in the current project had approximately 60000 sensors which generated around 300 GB of data per year [9]. Also the data generated during the process is both numeric and non-numeric. Numeric data included sensor data and non-numeric data included alarm and event data from the plant. For the case of anomaly detection only numeric data was required and hence any non-numeric sensor data was considered as a missing value. The characteristics of the data are also shown in the Table 1.

After getting the responses from the users and datasets from chemical plants, the next task was to deduce the functional and non-functional requirements for this investigation which are concisely shown in Table 1. In total there were four high level functional requirements for the system. First, the data mining processes required to perform anomaly detection on industrial continuous process data must be implemented within the system. Second, the flow of execution of the anomaly detection methods should be automated wherever possible. Third, the results obtained in each step of anomaly detection workflow should be made available to the user to be reused later. Forth, the changes made by the user to the data should be logged, so that they can be traced by the user at later stages.

Besides the data requirements, the non-functional requirements from both intended users overlapped more or less. The data scientists wanted a flexible,

extensible, visual and a scalable system. Besides having an independent system, having a scalable and a visual system was also required by the plant engineer. The user wanted flexibility in terms of choosing between a programming environment and an automated tool at any given time. The extensibility of the system was required in terms of adding and modifying existing data processing techniques according to user needs. The scalability of the system was required in terms of performance of the system, such that the performance should be improved on the addition of additional resources. The users wanted that the results relevant to them should be visualized using the relevant visualization techniques. The plant engineer also wanted that the system should be independently available at the premises of the process plant and should be able to assess the results of anomaly detection and monitor the incoming anomaly detection results without the help of data scientists.

3 Related Work

3.1 Big Data Architectures

The NIST defines big data as data sets that require scalable architectures for efficient storage, manipulation and analysis [8]. The need for scalable architectures is driven by the data characteristics volume, velocity, variety, and variability. Furthermore, big data systems are composed by distributed, horizontally coupled and independent resources that perform operation on data in massively distributed way (“shared-nothing architecture”).

Li [10] suggests a pattern for two data pipelines that produce reproducible and consistent results and can be productized. Two separate pipelines are suggested ranging for the (analytics) back-end from raw data, over ETL and model training to evaluation of the models and from raw data, over ETL, production model and prediction in the production case. In order to productionize especially the ETL process should be common between the back-end and production system. While the focus of the data pipelines discussed by Li is more on high latency analytics, assuming an ETL process in the production pipeline, the guidance to harmonize the back-end and production part of big data analytics is very valuable and can be applied in other scenarios as well. The Lambda Architecture proposed in [11] is a system designed to handle the challenges of Big Data applications. The main concept of the Lambda Architecture is to keep an immutable collection of the original data and to use separate tools for specific tasks. This is in contrast to traditional multi-purpose databases that try to perform incremental calculations on the original data. The approach of the Lambda Architecture offers several advantages. Because the original data is not changed, this layout offers a high fault tolerance, because there exists always a clean backup. However, calculating queries on the complete data set can also be slow and is ineffective. Thus, so-called batch views are calculated that preprocess the data. Batch views are usually realized in a classic Big Data environment like Hadoop, which offers parallelization and high reliability as integrated design features. Batch views are stored in a serving layer, which handles query request. Updating the batch

views might take some time so that queries to the serving layer might be already outdated as new data continuously arrives. Therefore, a speed layer is added to handle real-time analysis of arriving data until it is processed by the batch layer.

3.2 Big Data Mining Tools

In the early days of big data, analytics code had to be provided not only in the high-level programming language Java, but also in programming paradigm Map-Reduce [7]. Data scientists or data mining experts are usually not proficient in object oriented programming and parallel computing paradigms which lead to intermediate languages like Pig Latin [12] or SQL style languages [1]. This approach works well in application context with long-living, repeated big data workflows like social media or web/search data analysis. For applications that require an initial data exploration and rather short running analysis steps these languages are not a good match. The gap is addressed by data mining notebooks [6] like Jupyter Notebooks or Apache Zeppelin which enable an interactive analysis based on scripting languages like Python (in combination with e.g. PySpark) or R. These tools are not perfect for reuse of repetitive tasks that are usually done in the exact same way and do not support well handling several projects and data sets. The idea of the reusable Big Data system follows a entirely different approach to address the shortcomings of the development tools by providing a standardized workflow based on the CRISP DM process favored by data scientist. This enables the data scientist to leverage robust and tested big data procedures and out-of-the-box data visualisation. The integration of Notebook technologies and the extensibility of the system provides the still required flexibility.

4 Developed System

4.1 Application Workflow

Interviews with data scientists revealed that they use Cross Industry Standard process for Data Mining (CRISP-DM, [5]) more often to investigate anomaly detection problems like the one addressed in this study. Therefore, CRISP-DM was used to define the application workflow. Also, after going through the requirements and the anomaly detection steps a concise workflow was derived (see Fig. 1). The workflow starts by creating a project, which is followed by importing the raw data. Importing raw data can also be considered as matching the data collection phase of CRISP-DM. The import step ensures the homogeneity of the format for subsequent steps. The imported data is then passed for data profiling, where on retrieval of the data profiling results, data preparation steps can be performed.

The data profiling results gives metadata information about the data into consideration such as number of sensors, spread of the data and co-related sensors. Through the user interface, the user can take certain actions on the data

Table 1. Requirements

Functional requirements		
FR1	CRISP-DM	Support all CRISP-DM steps with pre-defined functionality
FR2	Automation	Automation of standard profiling and modeling steps
FR3	Iterative	Preserving results from each step of data mining process
FR4	Logging	Logging of user actions to make them reversible
Non-functional requirements		
NFR1	Visual	The system should visualize the data set, its properties and the anomaly detection results
NFR2	Scalable	The system should be scalable in response to growing data
NFR3	Flexible	Support custom coding of data scientist for exploration and cleaning
NFR4	Extensible	Easy to add new data preparation and modeling methods
NFR5	Independent	Independent screens for plant engineers and data scientists
Data properties and requirements		
DR2	Sampling	Provided data is equidistantly sampled. Usually per minute
DR5	Volume	Large volume - 365 * 24 * 60 * 1300 data points
DR3	Veracity	The data contains outlier, sensor failures, and missing values
DR1	Variety	The data contains numerical and non-numerical values and different sensor types (temperatures, level, pressures, etc.)
DR4	Redundancy	The data contains redundant and highly correlated sensor

such as selective or complete removal of sensor data. The selective removal of data is done based on the data quality issues such as missing values and outliers. The data can also be interpolated for missing values and outliers. These actions can be performed by clicking on the name of the sensor in the data profiling screen which brings a pop up as shown in Fig. 2. The pop up shows the data quality issues along with summary stats in the data recorded by a particular sensor. The screen is divided in four parts (in clockwise order from top left): Outliers, Missing values, Summary stats and Co-related sensors. The cleaning operations specified by the data scientist are stored by the system in a JSON format and is used for the data preparation phase to prepare the dataset.

Some of the data preparation steps required plant load change information from the plant engineer. Plant load change information describes when the amount of input raw material to the continuous process is increased or decreased.

The prepared dataset can be used for the following training process of the anomaly detection which can be run in the background in case of a large dataset. Once the anomalies are detected, the data scientist can verify the results with the help of known anomalies in the dataset, or he can get a feedback from the plant engineer by sharing the screen that show these results with him, where he can give direct feedback. Based on the feedback or the verification of results the data may be sent back again for profiling and preparation or can be used for online anomaly detection for incoming data.

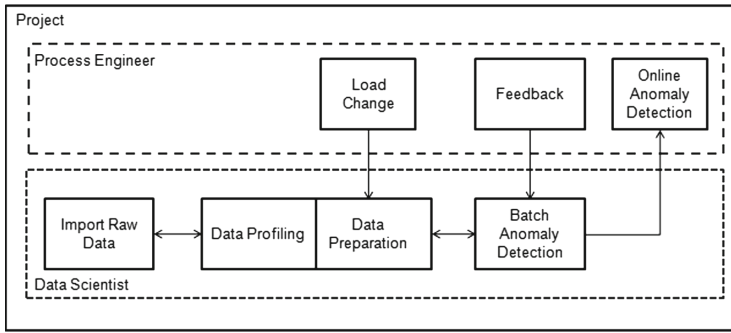


Fig. 1. Application workflow



Fig. 2. Data profiling result view (per sensor)

The sample of user interface showing the detected anomalies is shown in Fig. 3. The user interface visualizes the anomalies in a concise fashion at the top of the screen. Grouping of anomalies in this fashion gives a user-friendly view for analyzing the anomalies over several days as the information about anomalies in several sensors can be grouped together and their effect visualized in a single grid through color coding. The line chart in the middle of the screen can be used to visualize the anomalies in a sensor and also to visualize the nearest neighbors of the anomalies. This visualization can be controlled by selecting different sensors in the sensor list below the line chart. If displaying data of a single sensor, the anomalies are marked with a red band on the chart with the anomaly score marked at the top of the band.

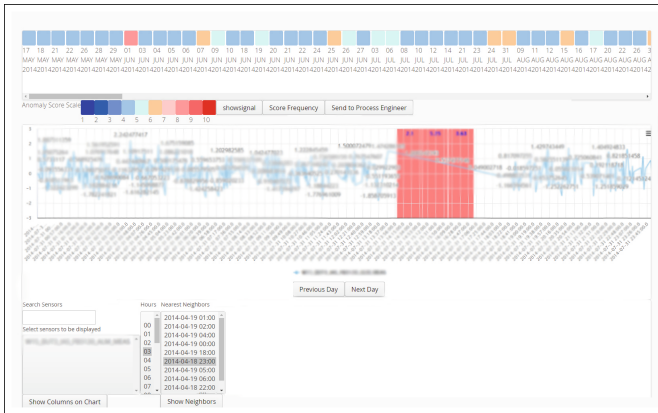


Fig. 3. Anomaly detection result view

4.2 Component Architecture

The anomaly detection for chemical plants requires both the analysis of massive amounts of historical data (in training of the algorithm) and fast processing based on intermediate results (namely the anomaly detection model). Thus, we decided to use the Lambda Architecture as a blue print for our system, Fig. 4. As discussed in Sect. 3, the lambda architecture has primarily three layers: Batch layer, Speed layer and Serving layer. Figure 4 shows the modified Lambda architecture proposed for the anomaly detection application to be used in chemical plants. In the lambda architecture new data is fed separately to the batch layer and the speed layer. In the modified architecture, a persistent publish subscribe messaging system is added before the batch and speed layers. The messaging system helps the architecture to eliminate the need to write additional code for batch and speed layer. Instead, a single connection API to the historian server is required for the retrieval of process data. This follows Li's suggestion to harmonize backend (data mining environment, batch layer) and production (anomaly detection). The batch layer in the original lambda architecture stores batch processing frame for the processing of historical data. The proposed modified lambda architecture consists of a distributed storage together with the distributed database for the storage of raw data files and historical data in a structured fashion respectively. The distributed batch processing framework has a REST server on top of it which helps in submitting the commands to the processing framework through a user interface. In case of a new project, historical raw data files are stored in the distributed storage which in turn are ingested into the distributed database with the help of batch processing framework. The batch processing framework on receiving a job request through the REST server processes the data according to the job received and stores the results to the serving layer. The serving layer in the traditional lambda architecture consists of storing results from batch computations and speed layer. In the modified archi-

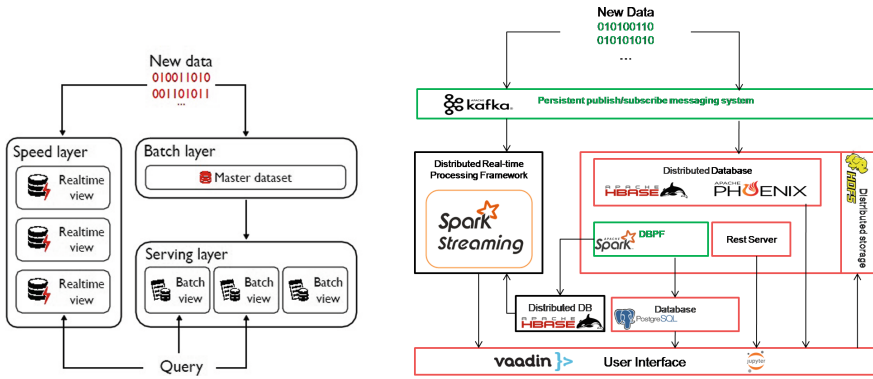


Fig. 4. Left: Lambda architecture [11] Right: Modified lambda architecture

architecture, the serving layer stores intermediate results in a distributed database from batch processing which are relevant for the speed layer and thus can be accessed through the speed layer. In addition to the distributed database, a traditional relational database (RDBMS) is used to store the anomaly detection results which can be queried by the user interface to display the result. Decision for an RDBMS was made because the results for anomaly detection were not large and could be therefore retrieved swiftly. The speed layer of the lambda architecture processes the incoming data as does the modified lambda architecture but just takes intermediate results of anomaly detection from the serving layer. It then appends the results to the RDBMS and passes on the results to the user interface for visualization.

4.3 Main Technology Decisions

The section describes about the technology decisions for the architecture, described in the previous section, which were made based on the requirements and without any technology comparison as it was out of scope of the study. However, technologies were chosen with NIST Big Data Definition cf. Sect. 3 in mind and thus the focus was on distributed shared-nothing technologies in order to support scalability w.r.t. to amount of data.

Apache Kafka³ was used as the publish/subscribe messaging system as it fulfills the requirement of fault tolerance and scalability. It can also handle velocity of the incoming data and both speed and batch layer can subscribe to it.

In the batch layer, the distributed database used is Apache HBase along with Apache Phoenix. Apache HBase is a columnar store which gives user a random access to the columns. This allows a quick access to timeseries sensor data in the chemical process data. Apache HBase is further masked by Apache Phoenix

³ Web-sites of open source used in the system: <http://kafka.apache.org/>, <https://hbase.apache.org/>, <https://phoenix.apache.org/>, <https://spark.apache.org/>, <https://github.com/spark-jobserver>, <https://vaadin.com/>.

which provides a SQL mask over HBase. Apache phoenix uses HBase as the database but takes SQL queries as the input and displays the data stored in HBase in the form of SQL tables. These technologies allow a quicker access to the data without increasing the learning curve of the end user. Apache Spark is used along with spark job server as the distributed batch processing framework and REST server respectively. Spark provides a swift in-memory distributed fault tolerant computation and therefore is faster than traditional Map-Reduce which stores intermediate computation results on the physical storage. HDFS is used as the distributed storage which can be used with Spark to process raw data files. For the processing in the speed layer, Spark Streaming is the selected technology as its micro-batch semantic matches the use case well and enables to share the same analytics code between batch and speed layer, following Li's (cf. Sect. 3) recommendation to harmonize back-end and production.

For storing the final results in the serving layer, Postgres has been used mainly because it has a support for storing data in JSON format within the relational tables. Postgres also serves as a backend to store the data of various user actions and to keep track of the spark jobs sent for execution to the spark job server. Serving layer consists of HBase for storing intermediate results of anomaly detection.

A user interface was required for both data scientist and plant engineer. While plant engineer wanted a visual interface for the anomalies in the chemical plant, data scientist needed it not only for anomaly visualization but also for carrying out a variety of data analysis tasks. Hence Vaadin, a Java web development framework, was chosen for user interface development. Data Scientists also had the requirement to be flexible in choosing the user interface or programming language to carry out the data analysis tasks. Jupyter notebooks was therefore chosen and embedded with the user interface. The advantage of Jupiter notebooks is that one can embed any programming language kernel (compiler /interpreter of a programming language) out of many kernels offered with the notebook.

5 Evaluation

5.1 Scalability Tests

The software architecture proposed in this study was built such that it is scalable. One point to note is that the scalability characteristics of an application also depends on its implementation. Therefore, this section discusses the result of horizontally scaling, the computation cluster, the execution time of data profiling, anomaly detection (Local Outlier Factor algorithm [3]) and the imported raw data into HBase using Phoenix. Data preparation was not considered for scalability tests as the execution time of data preparation job depends on the number of actions given by the user to be performed on the data. However, the parallelization approach is the same for data profiling, data preparation and modeling and the results from the two other phases can be easily transferred to the data preparation phase.

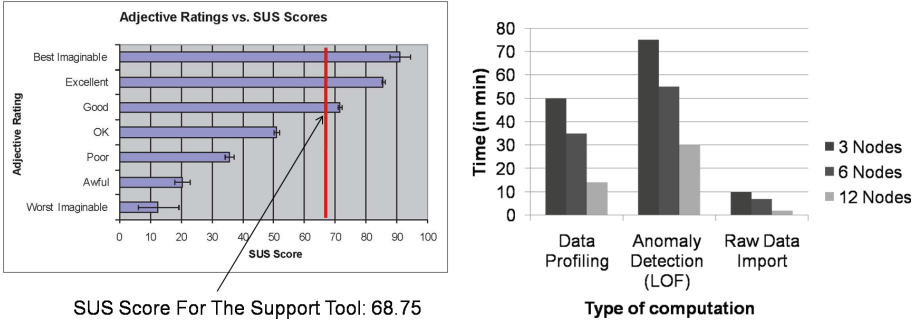


Fig. 5. Left: Usability test result, right: scalability results

The dataset taken into consideration for scalability tests consisted of 1300 sensors and 87860 time samples. The compute cluster consisted of 12 nodes where nodes are the number of machines. Each machine in the cluster had 8GB of RAM and an Octa-core processor. Figure 5 describes the results of horizontal scaling of data profiling, anomaly detection and import of raw data. The results are in execution times with different worker nodes for data profiling, anomaly detection and raw data import into Phoenix. The three results clearly show a reduction in the execution time for each task with the addition of nodes to the compute cluster.

5.2 Usability Tests

Brooke [4] describes system usability scale as a quick and dirty way to evaluate how usable any particular system is. It is quick as the template for SUS already exists and cheaper as it is a short quiz of ten questions that do not require much resources. The users are required to grade a question from 1 to 5 where 1 means they strongly disagree and 5 means they strongly agree. The system developed in the study was tested with the data scientists involved, by giving them two distinct datasets. After the data scientists completed their evaluation of the system, they were provided with the SUS questions and their responses were recorded. Finally, the scores from all the users were averaged to give an average system usability score which came out to be 68.75. To interpret the average scores an adjective rating is provided by Bangor [2] who conducted a study using several SUS scores for different systems and provided a guide to interpret the average SUS scores through adjective ratings. This interpretation has been used to interpret the average scores obtained for support tool built in this study. The adjective rating for the score obtained therefore came to be good which also suggests there is a room for improvements. These adjective ratings are shown in Fig. 5. The usability improvements, therefore, have been added as a part of future work.

6 Conclusion

The example of the presented big data system with the scope of anomaly detection for chemical plants shows that it is possible to facilitate reuse in industrial data mining by appropriate choice of application types and state-of-the-art big data tools without sacrificing the flexibility data scientists need, especially in the phase of data exploration and data understanding.



Selecting application types, that share enough commonalities between specific projects and have a large enough number of potential applications is in our opinion the key to develop such big data systems. Offering such systems that support data scientist in the standard activities and that foster reuse across project is a key enabler for increasing data mining efficiency and thus ultimately for sustainable and successful application of data mining in industrial contexts. Plant anomaly detection is an example for such a type of application, and the system described in this contribution helps to address the main challenges in industrial data mining. It is our strong belief that the approach can be replicated in other applications as well. This will be addressed by future work.

References

1. Armbrust, M., et al.: Spark SQL: relational data processing in spark. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp. 1383–1394. ACM (2015)
2. Bangor, A., Kortum, P., Miller, J.: Determining what individual SUS scores mean: adding an adjective rating scale. *J. Usability Stud.* **4**(3), 114–123 (2009)
3. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, pp. 1–12 (2000). <https://doi.org/10.1145/335191.335388>
4. Brooke, J.: SUS - a quick and dirty usability scale. *Usability Eval. Ind.* **189**(194), 4–7 (1996)
5. Chapman, P., et al.: Crisp-Dm 1.0. CRISP-DM Consortium, p. 76 (2000)
6. Chrimes, D., Moa, B., Zamani, H., Kuo, M.H.: Interactive healthcare big data analytics platform under simulated performance. In: 2nd International Conference on Big Data Intelligence, pp. 811–818. IEEE (2016)
7. Dean, J., Ghemawat, S.: Mapreduce: simplified data processing on large clusters. *Commun. ACM* **51**(1), 107–113 (2008)
8. NIST Big Data Public Working Group: NIST big data interoperability framework: volume 1, definitions. Technical report, National Institute of Standards and Technology (2015)
9. Klöpper, B., et al.: Defining software architectures for big data enabled operator support systems. In: 2016 IEEE 14th International Conference on Industrial Informatics (INDIN), pp. 1288–1292. IEEE (2016)
10. Li, M.: Three best practices for building successful data pipelines. In: Big Data Now 2015. O'Reilly, Sebastopol (2015)
11. Marz, N., Warren, J.: Big Data: Principles and Best Practices of Scalable Realtime Data Systems. Manning Publications Co., Shelter Island (2015)
12. Olston, C., Reed, B., Srivastava, U., Kumar, R., Tomkins, A.: Pig latin: a not-so-foreign language for data processing. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1099–1110. ACM (2008)



Unsupervised Domain Adaptation for Human Activity Recognition

Paulo Barbosa¹, Kemilly Dearo Garcia^{2,4}, João Mendes-Moreira^{1,3} ,
and André C. P. L. F. de Carvalho² 

¹ Faculty of Engineering, University of Porto, Porto, Portugal
up201607828@g.uporto.pt

² ICMC, University of São Paulo, São Paulo, Brazil
kemilly.dearo@usp.br, andre@icmc.usp.br

³ LIAAD-INESC TEC, Porto, Portugal
joao.mendes.moreira@inesctec.pt

⁴ University of Twente, Enschede, Netherlands

Abstract. Human Activity Recognition has been primarily investigated as a machine learning classification task forcing it to handle with two main limitations. First, it must assume that the testing data has an equal distribution with the training sample. However, the inherent structure of an activity recognition systems is fertile in distribution changes over time, for instance, a specific person can perform physical activities differently from others, and even sensors are prone to malfunction. Secondly, to model the pattern of activities carried out by each user, a significant amount of data is needed. This is impractical especially in the actual era of Big Data with effortless access to public repositories. In order to deal with these problems, this paper investigates the use of Transfer Learning, specifically Unsupervised Domain Adaptation, within human activity recognition systems. The yielded experiment results reveal a useful transfer of knowledge and more importantly the convenience of transfer learning within human activity recognition. Apart from the delineated experiments, our work also contributes to the field of transfer learning in general through an exhaustive survey on transfer learning for human activity recognition based on wearables.

Keywords: Human activity recognition · Transfer learning
Unsupervised domain adaptation

This work was partially funded by the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia within project POCI-01-0145-FEDER-016883.

1 Introduction

Human Activity Recognition (HAR) is a machine learning task that induces human activity models able to classify human activities. Though, HAR has relevant applications, including in health care, it still has several challenges that need to be overcome, including the subject sensitivity and model adaptability. The predictive performance of a classification model in human activities is largely affected by the physiological features of each person. For instance, young people perform ambulation activities different from elder or disabled people [1]. Although a specialised model could be developed for each individual, the cost would be much higher than to induce more generic models, with high predictive performance for different user profiles.

When HAR systems are used for classification of human activities, they usually assume the data distribution is stationary. However this is not necessarily true. Users tend to perform activities differently over time, as they get older or face temporary movement restrictions. Besides, sensors may need frequent calibration and are prone to malfunction.

In this work, we provide a literature revision in Transfer Learning (TL) to leverage the performance of HAR systems. TL is a Machine Learning (ML) sub-area that focuses on the re-utilisation of knowledge between ML tasks, by allowing the dataset distributions and even their training and testing sets be different.

The primary goal we expect to achieve by applying TL in HAR is through different, but related, datasets, boost the system's performance of the target domain. As a side effect of achieving this goal, we expect to reduce the effort of capturing labelled data increasing the efficiency of a HAR system.

2 Transfer Learning in HAR

HAR relies on the deployment of pervasive but non-intrusive sensors in the environment surrounding humans. HAR techniques are able to help understanding the context in which a person is involved, especially when help is needed, as in health care systems. Most typical ML models assume that both training and test datasets have the same distribution and feature space. This assumption is indeed a significant limitation since it's proven [2] that real problems suffer distribution mismatch over time and the test error is in proportion to the given difference.

2.1 Transfer Learning

The learning paradigm of TL uses past-knowledge, whether through datasets or model parameters, to leverage the performance of a different, but related, problem. In fact, this paradigm is inspired on our biological learning process, that the more experience we have better prepared are we when confronting novel conditions [3].

Essentially, a TL problem is composed of two categories of datasets, source and target. The primary goal is to, using a source dataset, leverage the performance in a target datasets. These two datasets have to be different whether regarding the source and target domains or both tasks. The knowledge of the particular cause of the difference is crucial, since it dictates which TL approach can be applied. A dataset domain consists of two elements, a feature space and a marginal probability distribution, while its task is composed of two elements, a label space and an objective predictive function. A distribution mismatch between datasets is considered a dataset-shift problem.

2.2 Dataset Shift in HAR

According to a survey of the related literature, the most likely shift-cause agents within HAR are the sensor and the user.

Sensor misplacement is likely to happen since it can sometimes be intrusive to the user causing him to change the position voluntarily. Furthermore, if even the sensor is placed in a comfortable position, it can be misplaced involuntarily during the activities, especially the ambulant ones. For this specific problem, we emphasize the works by Calatroni et al. [4], Khan et al. [5], Rokni [6], Krishnan [7], Kurz et al. [8], Roggen et al. [9], and Zhao et al. [10].

Regarding the user, the age and the physiology are the main causes for different activity executions and must be thought of when training a HAR classification model, to prevent training a model with users different from the target user. For user-related shift problems, various works have been proposed, such as the works of Deng et al. [11], Zhao et al. [12], Chen et al. [13], Hachiya et al. [14], Diethe et al. [15], and Fallahzadeh [16].

There are other shift causes, such as the environment itself, which are typical in smart homes HAR, where models are trained with laboratory data that do not accurately represent a real-world scenario. As research works have shown, laboratory environments generally restrict and influence subject activity patterns [17].

3 Unsupervised Domain Adaptation

Unsupervised Domain Adaptation is one specific scenario of TL recurrent in HAR contexts because it assumes the availability of labelled data only from the source domain and unlabelled data from the target domain, the same feature space and label space, and a distribution mismatch (e.g. different users, sensor's positions, surrounding environments). We firstly notice four different approaches to tackle this field: instance-weighting; self-labelling, feature representation; and cluster-based [19]. An instance weighting approach assigns weights to the source samples depending on their similarity with the target samples. A self-labelling approach focuses on adapting classifiers trained from the source samples to the new target domain. An approach of feature representation constructs an abstract representation of the data, while a cluster-based one assumes that from high-density regions we can perceive the similarity between the two domains.

3.1 Feature-Representation Employed Techniques

Transfer Component Analysis (TCA) [20] recurs to a dimensionality reduction method, Maximum Mean Discrepancy Embedding (MMDE) [21], that learns a low-dimensional latent space common to both domains. This way, it embeds both source and target domains into a shared low-dimensional latent space using a non-linear mapping function to learn then a corresponding kernel matrix. The kernel matrix is defined on all data by minimising the distance, measured with the Maximum Mean Discrepancy metric, between the projected two domains while maximising the embedded data variance. Having found a transformation matrix, a classifier can finally be trained with it to predict the target domain accurately.

Subspace Alignment (SA) [22] first generates the subspaces by representing the source and target domains into subspaces spanned by eigenvectors. Then, it focuses on the alignment of the two subspaces to decrease the discrepancy between both the source and target domains.

3.2 Instance-Weighting Employed Techniques

Nearest Neighbour Weighting (NNeW) [23] applies a Voronoi tessellation of the space to help to determine the weights for each source sample. Hence, the weight on each sample is dependent on the number of target neighbour samples. As the final step, it is then applied Laplace smoothing for regularising the weights to avoid some getting exaggeratedly biased towards the test set.

Kernel Mean Matching (KMM) [24] directly estimates source weights without performing density estimation, as most algorithms in this scenario. Maximum Mean Discrepancy is the technique used to define the weights in a way that the divergence between the source and target domain is minimised.

4 Experiments

Employing two HAR datasets, PAMAP2 [25] and ANSAMO [26], we evaluated the four TL techniques (TCA, SA, KMM, NNeW) in various HAR scenarios having different mismatches of distribution between the source and target datasets. In total, we delineated five different experiment scenarios (AA-AGE, AA-POS, AP-ENV, PA-ENV, PP-POS), where we explore the user's age, sensor misplacement, and environment as shift-causes.

As for the employed techniques, we adopted three different learning approaches, TL, supervised and semi-supervised for comparison purposes. The two latter approaches are necessary since both serve as baseline methods helping to identify negative transfer in each TL approach.

Each TL technique had to be linked with a classifier, since the TL techniques' responsibility was only adjoining the source and target distributions, leaving classification task to the classifier. Therefore we incorporated each TL technique with four different classifiers (Logistic Regression, Bernoulli Naive Bayes,

Table 1. Average accuracy obtained in each experiment scenario per technique.

	AA-AGE	AA-POS	AP-ENV	PA-ENV	PP-POS
SUP	0,52	0,25	0,65	0,71	<u>0,57</u>
SEMI	0,30	0,24	0,49	0,46	0,33
KMM	0,58	0,35	0,72	0,77	0,50
NNeW	0,57	0,34	0,68	0,78	0,55
SA	0,54	0,24	0,68	0,76	<u>0,57</u>
TCA	0,35	0,27	0,27	0,25	0,19
ENS	<u>0,60</u>	<u>0,36</u>	0,65	<u>0,83</u>	0,54

Decision Tree, Linear Support Vector Machine) to find the highest performance combination. Having found the best combinations, we posteriorly implemented a majority voting ENSemble (ENS) composed of KMM with Decision Tree, NNeW with Logistic Regression, and SA with Logistic Regression.

Regarding the semi-supervised approaches, we used two different iterative techniques, Label Propagation [27] and Label Spreading [28], where a model is trained with both the labelled source and unlabelled target samples.

As for the supervised approaches, we adopted the same four classifiers of TL, where a model is taught only with the original source dataset and tested with the target sample. These two approaches are necessary to help identify negative transfer in the TL results.

4.1 Age, Position, and Environment as Shift-Causes

The AA-AGE scenario focuses on exploring the users' age as the leading cause for distribution mismatch between datasets. Using only the ANSAMO dataset, the younger users compose this scenario's source dataset to help classify the activities of the older people unlabelled target dataset. In total we implemented 3 experiments, each trying to classify from different positions (ankle, chest, and wrist) 6 activities (bending, hopping, sitting, ascending stairs, and walking).

Regarding the AA-POS and PP-POS scenarios, the goal was to explore the misplacement of sensors as the leading cause for distribution mismatch between datasets. Each experiment consists of using a source dataset with data of a particular position (e.g. wrist) to help identify the activities of a target dataset captured from a different position (e.g. ankle). Two scenarios were devised, one using the ANSAMO, named AA-POS, and the other the PAMAP2 dataset, named PP-POS. Since the ANSAMO dataset had data captured from four different positions (ankle, chest, waist, wrist), and the PAMAP from three (ankle, chest, wrist), we devised 18 unique experiments. As for the activities, the AA-POS scenario had 7 (bending, hopping, running, sitting, ascending stairs, descending stairs, and walking) and PP-POS had 8 (lying, rope jumping, running, sitting, ascending stairs, descending stairs, standing, and walking).

As for the AP-ENV and PA-ENV scenarios, the goal was to explore the environment as the primary cause for distribution mismatch between datasets. Two scenarios were delineated, AP-ENV and PA-ENV, in which the former uses the ANSAMO dataset and PAMAP2 as source and target datasets, respectively. The latter employs the PAMAP2 dataset to leverage the activity recognition in the ANSAMO dataset. Every implemented experiment classifies 5 activities (running, sitting, ascending stairs, descending stairs, and walking) in 3 different positions (ankle, chest, wrist).

4.2 Results and Discussion

Altogether, we tested 22 different approaches in 5 experiment scenarios. As depicted in Table 1, in 3 of the 5 scenarios, the TL approaches had a significant transfer of knowledge particularly in the age and environment shift-cause scenarios. The ensemble approach, as expected, obtained the highest score gaining 8% in AA-AGE, 11% in AA-POS, and 12% in PA-ENV comparing with the supervised approach.

Furthermore, by comparing all the yielded results between the ensemble and baseline methods through Friedman Sum Rank test we are able to observe a significant difference between the three methods for the significance level of 1% (p-value = 6.707e-07, p-value < 0.01). Additionally, with a Nemenyi post-hoc test [29] we are also able to observe that there is a highly significant difference between the ensemble and the semi-supervised approach and between the ensemble and the supervised approach for a significance level of 5% (p = 3.3e-07 and p = 0.022, respectively).

Moreover, Fig. 1 has the comparative performance between ensemble and supervised models, and between ensemble and semi-supervised models. In overall, the results show the usefulness of applying TL in scenarios where there is a considerable distribution mismatch. While the ensemble has a median positive

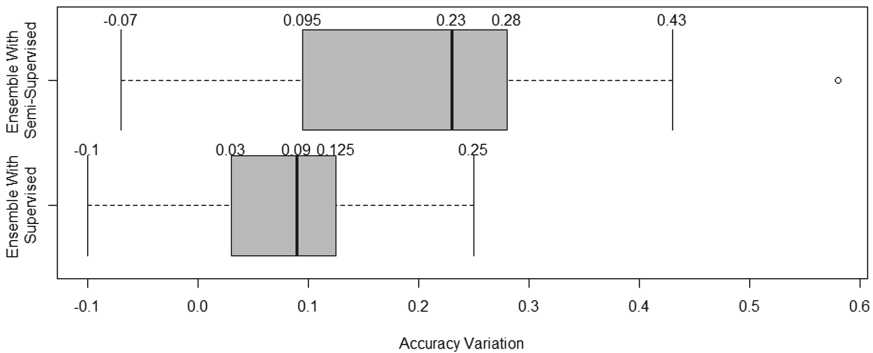


Fig. 1. Accuracy’s variation between the ensemble technique and the supervised approach, and between the ensemble technique and the semi-supervised approach.

variance relative to the semi-supervised of 23%, it has a minor variation relative to the supervised approach with 9%.

5 Conclusions

Overall, we accomplished the delineated objectives. Our work not only contributes to the unsupervised domain adaptation field but to TL as well through the extensive performed survey. The yielded results indicate a highly significant difference between the TL ensemble approach with the semi-supervised approach and a less but significant difference with the supervised approach.

As for future work, we envision the study of this same domain but with an online learning approach. Having obtained in experiments negative transfer by some TL techniques, especially with TCA, it would be better employing multiple source datasets, which would likely have higher success since it increases the chance of discovering useful transferable knowledge between the source and target datasets.

References

1. Malbut-Shennan, K., Young, A.: The physiology of physical performance and training in old age. *Coron. Artery Dis.* **10**(1), 37–42 (1999)
2. Sun, B., Feng, J., Saenko, K.: Return of frustratingly easy domain adaptation. In: AAAI, vol. 6, no. 7 (2016)
3. Thrun, S., Pratt, L. (eds.) *Learning to Learn*. Springer, Heidelberg (2012)
4. Calatroni, A., Roggen, D., Tröster, G.: Automatic transfer of activity recognition capabilities between body-worn motion sensors: training newcomers to recognize locomotion. In: 8th International Conference on Networked Sensing Systems (2011)
5. Khan, M.A.A.H., Roy, N.: TransAct: transfer learning enabled activity recognition. In: 2017 IEEE International Conference on Pervasive Computing and Communications Workshops, pp. 545–550 (2017)
6. Rokni, S.A., Ghasemzadeh, H.: Autonomous training of activity recognition algorithms in mobile sensors: a transfer learning approach in context-invariant views. *IEEE Trans. Mob. Comput.* **17**(8), 1764–1777 (2018)
7. Krishnan, N.C.: A computational framework for wearable accelerometer-based. Dissertation, Ph.D. thesis, Arizona State University (2010)
8. Kurz, M., et al.: Real-time transfer and evaluation of activity recognition capabilities in an opportunistic system. In: The Third International Conference on Adaptive and Self-Adaptive Systems and Applications, pp. 73–78 (2011)
9. Roggen, D.: The adarc pattern analysis architecture for adaptive human activity recognition systems. *J. Ambient. Intell. Hum. Ized Comput.* **4**(2), 169–186 (2013)
10. Zhao, Z.: Cross-mobile elm based activity recognition. *Int. J. Eng. Ind.* **1**(1), 30–38 (2010)
11. Deng, W.-Y., Zheng, Q.-H., Wang, Z.-M.: Cross-person activity recognition using reduced kernel extreme learning machine. *Neural Netw.* **53**, 1–7 (2014)
12. Zhao, Z., et al.: Cross-people mobile-phone based activity recognition. In: IJCAI, vol. 11, no. 3 (2011)

13. Chen, H., Cui, S., Li, S.: Application of Transfer Learning Approaches in Multimodal Wearable Human Activity Recognition. arXiv preprint [arXiv:1707.02412](https://arxiv.org/abs/1707.02412) (2017)
14. Hachiya, H., Sugiyama, M., Ueda, N.: Importance-weighted least-squares probabilistic classifier for covariate shift adaptation with application to human activity recognition. *Neurocomputing* **80**, 93–101 (2012)
15. Diethe, T., Twomey, N., Flach, P.: Active transfer learning for activity recognition. In: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning* (2016)
16. Fallahzadeh, R., Ghasemzadeh, H.: Personalization without user interruption: boosting activity recognition in new subjects using unlabelled data. In: *Proceedings of the 8th International Conference on Cyber-Physical Systems*. ACM (2017)
17. Laasonen, K., Raento, M., Toivonen, H.: Adaptive on-device location recognition. In: Ferscha, A., Mattern, F. (eds.) *Pervasive 2004*. LNCS, vol. 3001, pp. 287–304. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24646-6_21
18. Blanke, U., Schiele, B.: Remember and transfer what you have learned-recognizing composite activities based on activity spotting. In: *2010 International Symposium on Wearable Computers (ISWC)*. IEEE (2010)
19. Margolis, A.: A literature review of domain adaptation with unlabeled data. Technical report, pp. 1–42 (2011)
20. Pan, S.J.: Domain adaptation via transfer component analysis. *IEEE Trans. Neural Netw.* **22**(2), 199–210 (2011)
21. Pan, S.J., Kwok, J.T., Yang, Q.: Transfer learning via dimensionality reduction. In: *AAAI*, vol. 8 (2008)
22. Fernando, B., et al.: Subspace alignment for domain adaptation. arXiv preprint [arXiv:1409.5241](https://arxiv.org/abs/1409.5241) (2014)
23. Loog, M.: Nearest neighbor-based importance weighting. In: *2012 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE (2012)
24. Huang, J., et al.: Correcting sample selection bias by unlabeled data. In: *Advances in Neural Information Processing Systems* (2007)
25. Santoyo-Ramón, J.A., Casilari, E., Cano-García, J.M.: Analysis of a smartphone-based architecture with multiple mobility sensors for fall detection with supervised learning. *Sensors* **18**(4), 1155 (2018)
26. Reiss, A., Stricker, D.: Creating and benchmarking a new dataset for physical activity monitoring. In: *Proceedings of the 5th International Conference on Pervasive Technologies Related to Assistive Environments*. ACM (2012)
27. Zhu, X., Ghahramani, Z.: Learning from labeled and unlabeled data with label propagation, p. 1 (2002)
28. Zhou, D., et al.: Learning with local and global consistency. In: *Advances in Neural Information Processing Systems* (2004)
29. Demsar, J.: Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **7**(Jan), 1–30 (2006)



Data Set Partitioning in Evolutionary Instance Selection

Mirosław Kordos¹(✉), Łukasz Czepielik¹, and Marcin Blachnik²

¹ Department of Computer Science and Automatics, University of Bielsko-Biala,
Willowa 2, Bielsko-Biala, Poland

`mkordos@ath.bielsko.pl`

² Department of Applied Informatics, Silesian University of Technology,
Krasińskiego 8, Katowice, Poland

`marcin.blachnik@polsl.pl`

Abstract. Evolutionary instance selection outperforms in most cases non-evolutionary methods, also for function approximation tasks considered in this work. However, as the number of instances encoded into the chromosome grows, finding the optimal subset becomes more difficult, especially that running the optimization too long leads to over-fitting. A solution to that problem, which we evaluate in this work is to reduce the search space by clustering the dataset, run the instance selection algorithm for each cluster and combine the results. We also address the issue of properly processing the instances close to the cluster boundaries, as this is where the drop of accuracy can appear. The method is experimentally verified on several regression datasets with thousands of instances.

1 Introduction

Data preprocessing is frequently the most important step in data mining, even more important than choice of the learning model and its parameters, as even the best model cannot produce good outcome if the data quality is poor. An important step of data pre-processing is data selection, this is feature selection and instance selection. So far much more research has been conducted on feature selection than on instance selection and a few papers proposed joint approach [1]. There are two objectives of data selection: to improve the prediction accuracy and to reduce the training dataset size to make the model learning faster and data analysis easier. In classification tasks the accuracy is usually expressed by the percentage of correctly classified instances and in regression tasks by *rmse* (root mean square error).

Most of non-evolutionary methods of instance selection belong to two groups. The first group is used prior to the model learning and it decides upon the removal of particular instances by examining their local neighborhood. A number of such methods mostly for classification tasks and also a few for regression were proposed in the literature. A comprehensive review can be found in [2] and [3].

The second group embeds instance selection into the model learning by some modifications or additions to the learning algorithm [4].

The purpose of instance selection methods can be divided into three categories: noise filters, data condensation methods and methods joining both tasks. The noise filters are designed to remove only those instances, which constitute noise in the data. Removing them only slightly reduces the dataset size (usually below 10%), but it improves prediction accuracy.

Data condensation methods are frequently used as the second step, after noise has been removed. Their purpose is to find and remove the irrelevant instances, this is instances that can be removed from the training set T without impacting the prediction accuracy of a model trained on this set. In case of the condensation methods, usually stronger data reduction causes also decrease in prediction accuracy. However, that can still be useful if the reduction is very strong and the accuracy loss is minimal, because this will allow to have a better insight into the data properties and to make some preliminary experiments on the reduced dataset, so that they run quickly, what is especially important for the learning models with high computational cost.

In evolutionary methods based on genetic algorithms the dataset is usually represented by a chromosome, where each position (allele) corresponds to one instance; 0 at this position means the instance will be rejected and 1 means it will be selected. Then the optimization runs and the fitness function, which is a weighted sum of the model accuracy and the training dataset reduction is evaluated. That is the general principle and particular solutions may implement it in various ways.

One advantage of evolutionary-based methods is that there is no need to define the exact rules for the instance removal. The other advantage is that most cases better results in terms of accuracy-reduction balance can be obtained [5,6].

In evolutionary instance selection, which we discussed in Sect. 2, the final objectives are: minimization of training dataset size (number of instances) and minimization prediction error (*rmse* in case of regression) on the test set. However, the test set is unknown during the optimization and thus the second objective cannot be minimized directly, instead the error on the training set is minimized. Thus there is the same problem as with learning the predictive models: running the optimization on the training set for too many iterations leads to over-fitting and thus increases of error on the test set.

There is no simple general solution to the problem of uncertainty in genetic optimization as shown in [7] as the fitness function evaluation is different as well between problems as in different phases of the optimization in a single problem. The problem gets more serious for larger datasets and in this paper we address this issue by partitioning the training dataset. Then we run the optimization on the parts, and merge the results properly, as discussed in Sect. 3. The experimental evaluation presented in Sect. 4 proves that this approach allows for obtaining better results in terms of *rmse* - data reduction balance.

2 Evolutionary Instance Selection

There have been already some propositions in the literature to use genetic algorithms for instance selection, which confirmed that these approaches can find better solutions than the classical methods based on the local distances or neighborhood [5, 6, 8–11].

The way in which genetic algorithms work is with very high probability well known to the reader so we will not explain this, especially that all the details can be easily found in literature. In our application a single objective genetic algorithm is used [12, 13].

Each individual in the genetic population encodes the entire training set and the value at each chromosome position indicates whether the instance is present (value=1) or not (value=0). In the sample chromosome below, the dataset consists of 20 instances; the first instance is selected, the second rejected, the next three instances are selected, next two rejected and so on.

1|1|0|1|1|1|1|0|0|1|1|1|1|0|1|1|1|0|0|1|1|0|1|1|1|1|1|1|

A prediction model is used during the optimization to calculate the error on the training set. (We call it an inner prediction model to distinguish it from the final prediction model, which is trained on the reduced dataset and predicts the outputs for the test set.) Thus, if there are e.g. 96 individuals and the optimization runs for 25 iterations, the error on the training set has to be evaluated 2400 times. In order to significantly reduce computational cost of calculating the error, we use k-NN with pre-calculated and sorted distance matrices as the inner prediction model. We calculate and sort the distances between each pair of instances in the training set only once before the optimization starts. Then during the optimization we read the output values of the k nearest selected neighbors from the sorted arrays. If a given nearest neighbor is rejected in the current individual, then we go to the next one until we read from the sorted array the outputs of k selected instances. Then we predict the output value as the average of the k selected nearest neighbor outputs and calculate the *rmse* over the training dataset. This is really very fast, making the computational cost of this method comparable to the cost of the non-evolutionary instance selection methods.

Most of evolutionary approaches to instance selection define the fitness function in a similar way, as a weighted sum of the achieved reduction and inverse of the error on the training set. Thus to obtain a set of solutions the optimization has to be performed several times with various weights α . These two objectives are composed into the following fitness function:

$$fitness = \left(\alpha * \frac{avgRMSE}{RMSE} + (1 - \alpha) * \frac{avgNumInstances}{numInstances} \right)^v \quad (1)$$

where exponent v usually gradually increases as the optimization progresses (e.g. from $v = 2$ to $v = 4$) and each individual is selected as a parent in the crossover process with a probability proportional to its fitness.

Another solution is to use multi-objective evolutionary algorithms [14], where such a fitness function does not need to be defined, but only the criteria: *rmse* and instance reduction [16, 17].

We use multi-point multi-parent crossover with M parents and $M - 1$ crossover points. M parents are randomly selected proportionally to their fitness for each child; each of them gives its fragment of genetic material into one segment of the child, between two successive crossover points. After producing offspring with the crossover operator (with probability of 100%), we sort all parents and children together and P best individuals are advanced to the next iteration. We use the mutation probability of 0.001. Such parameters were selected based on our previous work [5]. In the experiments with we used two α values: 0.9 and 0.8. We used population size $P = 96$, because it is within the range of optimal population size and our servers had 48 CPU cores so parallel calculations scaled efficiently ($2 * 48 = 96$).

3 The Proposed Improvement

As it was mentioned, one of the final objectives is the *rmse* on the test set, but the fact the objective that we optimize directly is *rmse* on the training set, because during the optimization the test set is unknown and this requires early stopping, before the over-fitting occurs. For big datasets, there is a problem with determining the optimal stopping point. We observed that in some areas of the dataset the optimization converges faster than in others. Goldberg wrote that genetic algorithms work “by building short, low order, and highly fit schemata (blocks), which are recombined (crossed over), and re-sampled to form strings of potentially higher fitness” [12]. In this way, the over-fitting already may start occurring within such a block, while in other parts of the chromosome still more optimization is needed.

Even in tasks, where the objectives are directly optimized and thus there is no risk of over-fitting, partitioning the search space can be useful, because the already created blocks can be disturbed in the process of creating new ones and some researchers have already tried this approach. In [19] a differential evolutionary algorithm with space partitioning was implemented by dividing the search variables into groups of partitions, so each partition contained a certain number of variables and was manipulated as a subspace in the search process. In [18] the search space partitioning was applied to multi-objective genetic algorithms.

We compared the results of the evolutionary based instance selection with the instance selection performed by the regression version of the DROP3-RT algorithm [20] (which according to the comparative study in [3] was one of the best classical instance selection algorithms). For smaller datasets the differences were huge: the evolutionary method was much better, but for the dataset sizes of about 10.000 instances the differences begun to get much smaller and in some cases even the *rmse* of DROP3-RT was lower. Introducing the data partitioning and performing the instance selection separately inside each partition again allowed for gaining almost as significant advantage over the DROP-3 results as for the smaller datasets.

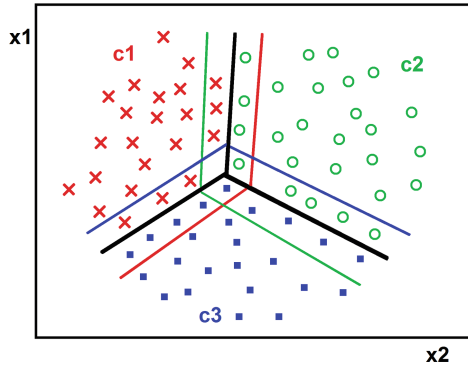


Fig. 1. The exact and the extended clusters (Color figure online)

We use the k-means clustering to partition the n instances into k ($k < n$) sets (clusters) S_1, S_2, \dots, S_k in order to minimize the within-cluster sum of squares (or variance):

$$\arg \min_{\mathbf{S}} \sum_{c=1}^k \sum_{\mathbf{x} \in S_c} \|\mathbf{x} - \boldsymbol{\mu}_c\|^2 = \arg \min_{\mathbf{S}} \sum_{c=1}^k |S_c| \text{Var} S_c \quad (2)$$

where c is the number of the cluster. As k-means is a well known algorithm and its description can be easily found in the literature [15] so we do not describe it here. However, it is worth mentioning, that we use also the output variable of the instances as one of the input variables to the clustering, as this proved to produce more adequate results for the purpose of instance selection.

Of course not each optimization problem can be partitioned in this way, because frequently there are mutual interactions between many positions in the chromosome. However, in the case of predicting the output of regression datasets it can be done, because there are no interactions between very remote instances.

Figure 1 shows a dataset with two attributes x_1 and x_2 partitioned into three clusters c_1, c_2 and c_3 with the thick black lines. The thin red lines represent the boundaries of the extended red cluster c_1 ; this is the red cluster itself and all instances from the other clusters that are the nearest neighbors for any instance from the red cluster.

Algorithm 1. The instance selection process

```

for  $m=1$  to 10 (crossvalidation) do
  Partition the  $m - th$  training  $T_m$  set into  $C$  clusters
  Optionally determine the extended clusters  $E$  for each of the  $C$  clusters
  for  $c=0 \dots C$  do
    Generate initial currentPopulation of  $P$  individuals
    calculate fitness  $f$  (Eq. 1) for currentPopulation individuals
    for  $i=0 \dots numIterations$  do
      apply the crossover operation to generate the newPopulation of  $P$  individuals
      calculate fitness  $f$  (Eq. 1) for newPopulation individuals
      sort together currentPopulation and newPopulation individuals by fitness
      select the best  $P$  individuals into currentPopulation
      apply the mutation operation
    end for
  end for
  Merge the selected instances from all  $C$  clusters to form the final reduced training set  $T_{mr}$ 
  calculate the  $reduction_m$  of the training set  $T_{mr}$ 
  calculate the  $rmse_m$  on the corresponding test set  $S_m$  using the predictor trained on  $T_{mr}$ 
end for
average the reduction and  $rmse$  over the  $m$  sets to get the final result

```

This approach usually allows for improving both criteria: obtaining higher reduction with slightly lower $rmse$, but definitely the reduction criterion gets more improved than the $rmse$ criterion. There is some “boundary effect”; this is we can observe some problems with the instances that are so close to the cluster borders that some of their nearest neighbors belong to another cluster. When we want to predict the output of such an instance using only the neighbors from its own cluster, the results are inadequate. For this reason we cannot further decrease $rmse$ using only that approach. Moreover, the decreasing prediction error inside the clusters is partially canceled out by the increasing error on the borders of the clusters. As a result the optimal size of the cluster in our experimental evaluation ranged from 500 to 1.500 instances.

In order to further improve the results, we propose to use a pair of datasets for each cluster: the dataset that forms the current cluster T_c and the extended dataset T_e which consists of the current cluster T_c and all k nearest neighbors of all instances from T_c , no matter to which cluster they belong, as shown in Fig. 1. We perform instance selection only within T_c , so only the instances from T_c are considered for removal. However, while predicting with k-NN the outputs of instances from T_c (which we need to obtain $rmse$), all the instances from T_e are considered as their potential neighbors. The method is shown in the pseudo-code 1.

4 Experiments and Results

The two criteria to assess the quality of the solution are: data reduction and $rmse$ on the test set. Because of limited space, we present here only results obtained with single-objective genetic algorithm and with 1-NN and optimal k

k-NN as the inner-evaluation model and as the final prediction model 1 (optimal k means k for which the lowest *rmse* was obtained on the whole training set in 10-fold crossvalidation). The source code of the software we created to perform the experiments (which also implements multi-objective instance selection algorithms and other prediction models) and the datasets can be found in supplementary materials to this paper at www.kordos.com/ideal2018.

As the dataset partitioning is useful for bigger datasets, we conducted the experiments on the 16 biggest benchmark regression datasets from the well known KEEL Repository [21]. The datasets were standardized to enable better comparison of the results in the tables. These datasets describe the real-world problems and are commonly used for evaluating different algorithms and thus provide much more reliable results than generating some artificial datasets (someone could argue that the artificial datasets were especially generated in such a way to show the desired results).

First we randomly changed the order of the instances in each dataset and then we used linear sampling in 10-fold crossvalidation to ensure that both of the methods (with and without data partitioning) use exactly the same training and test subsets - so that the comparison is made on exactly the same data and the results are not influenced by different sets. We used two-tailed Wilcoxon Signed-Rank Test and in each case the difference was statistically significant at $p < 0.05$. The results presented in the second and third table are the average *rmse* values obtained on the test sets in the crossvalidation and the corresponding average percentage of selected instances of the training set (retention rate).

The final prediction model and the inner evaluation model was k-NN with optimal k for each dataset. However, in the experiments, we limited the maximum k to 11. The rationale behind this was that only the most noisy datasets had optimal k value above 11 and after performing instance selection, which reduced the noise, the optimal k dropped to lower values, not exceeding 11. The number of clusters N_c used for each dataset is shown in Table 1. As our experiments showed the optimal number of instances in one cluster was between 500 and 1500 (Table 3).

If we used more instances, then the effect of the genetic algorithm not finding the optimal solution before over-fitting started to be visible. If we decreased the number of instances below the lower limit, the border effect started to decreasing the result quality. The precise optimization of the cluster size is not required and even not possible, as the differences between the algorithm performance with the cluster size of e.g. 400 and 700 are below the variability caused by the stochastic character of genetic algorithms.

Table 1. Datasets used in the experiments (obtained from the Keel repository) and their properties: **inst** - number of instances, **attr** - number of attributes, **oK** - the optimal k (the k in k -NN that gives the lowest $rmse$), **uK** - the k used the experiments in place of optimal k , **Nc** - the number of clusters.

Dataset	inst	attr	oK	uK	Nc	Dataset	inst	attr	oK	uK	Nc
wankara	1609	9	9	9	5	delta-elv	9516	6	35	11	20
plastic	1650	2	30	11	5	tic	9822	85	90	11	20
quake	2178	3	50	1	5	ailerons	13750	40	10	10	20
anacalt	4052	7	2	2	10	pole	14998	26	4	4	20
abalone	4177	8	13	11	10	elevators	16598	18	8	8	20
delta-ail	7128	5	17	11	15	california	20640	8	9	9	20
puma32h	8191	32	21	11	15	house	22784	16	11	11	20
compactiv	8192	21	2	2	15	mv	40767	10	9	9	40

Table 2. Experimental results: inner evaluation algorithm: 1-NN, final prediction algorithm: 1-NN, r0 - $rmse$ without instance selection, r1, r2 - $rmse$ with instance selection without data partitioning for $\alpha = 0.9$ and 0.8, r1p, r2p - $rmse$ with data partitioning for $\alpha = 0.9$ and 0.8 (smaller is better), c1, c2, c1p, c2p - corresponding retention rates (smaller is better)

Dataset	r0	r1	r1p	c1	c1p	r2	r2p	c2	c2p
wankara	0.225	0.222	0.209	0.679	0.623	0.290	0.266	0.147	0.136
plastic	0.617	0.542	0.513	0.718	0.674	0.607	0.589	0.202	0.192
quake	1.344	1.153	1.134	0.672	0.607	1.342	1.296	0.148	0.133
anacalt	0.227	0.232	0.211	0.482	0.470	0.282	0.280	0.194	0.171
abalone	0.915	0.768	0.747	0.697	0.655	0.872	0.844	0.163	0.154
delta-ail	0.716	0.630	0.629	0.618	0.593	0.744	0.740	0.141	0.135
puma32h	1.212	1.025	1.004	0.673	0.644	1.202	1.207	0.165	0.154
compactiv	0.254	0.295	0.295	0.434	0.414	0.307	0.307	0.236	0.216
delta-elv	0.828	0.708	0.710	0.663	0.623	0.838	0.819	0.136	0.128
tic	1.366	1.130	1.118	0.704	0.681	1.330	1.312	0.148	0.142
ailerons	0.657	0.585	0.580	0.596	0.583	0.701	0.688	0.142	0.139
pole	0.244	0.258	0.258	0.662	0.648	0.349	0.335	0.137	0.131
elevators	0.686	0.641	0.635	0.669	0.651	0.763	0.730	0.173	0.151
california	0.654	0.596	0.582	0.676	0.655	0.711	0.703	0.143	0.135
house	0.872	0.775	0.766	0.645	0.625	0.929	0.924	0.141	0.137
mv	0.210	0.199	0.199	0.728	0.721	0.302	0.300	0.164	0.160
average	0.689	0.610	0.599	0.645	0.617	0.723	0.709	0.161	0.151
Wilcox. p-value	0.00236			0.00044		0.00180		0.00044	

Table 3. Experimental results: inner evaluation algorithm: optimal-k k-NN, final prediction algorithm: optimal-k k-NN, symbols are explained at Table 2

Dataset	r0	r1	r1p	c1	c1p	r2	r2p	c2	c2p
wankara	0.167	0.183	0.171	0.447	0.420	0.217	0.217	0.154	0.146
plastic	0.468	0.452	0.452	0.427	0.398	0.466	0.466	0.219	0.198
quake	1.025	1.009	1.010	0.676	0.611	1.030	1.029	0.211	0.200
anacalt	0.212	0.212	0.211	0.480	0.459	0.274	0.273	0.167	0.163
abalone	0.702	0.708	0.705	0.681	0.640	0.756	0.755	0.141	0.132
delta-ail	0.560	0.575	0.572	0.692	0.585	0.605	0.607	0.167	0.148
puma32h	0.896	0.909	0.911	0.477	0.458	0.910	0.905	0.215	0.206
compactiv	0.231	0.277	0.267	0.507	0.463	0.288	0.285	0.258	0.242
delta-elv	0.610	0.622	0.625	0.700	0.613	0.622	0.624	0.230	0.202
tic	1.015	0.996	0.989	0.680	0.656	1.014	1.009	0.177	0.160
aileron	0.504	0.519	0.516	0.477	0.469	0.555	0.551	0.255	0.231
pole	0.214	0.236	0.233	0.487	0.478	0.254	0.252	0.303	0.302
elevators	0.559	0.577	0.572	0.465	0.451	0.625	0.620	0.207	0.205
california	0.527	0.549	0.551	0.527	0.516	0.579	0.580	0.336	0.324
house	0.687	0.708	0.708	0.563	0.538	0.727	0.721	0.269	0.261
mv	0.140	0.160	0.160	0.488	0.484	0.187	0.186	0.251	0.242
average	0.532	0.543	0.541	0.548	0.515	0.569	0.567	0.222	0.210
Wilcox. p-value	0.04236			0.00044		0.02382		0.00044	

5 Conclusions

We discussed the improvement of instance selection for regression tasks using genetic algorithms with clustering to partition the data space. The experiments showed that in the tested configuration this allows for stronger reduction of the data size and also for some reduction of *rmse*; we were able to reduce the dataset size by on average 5.5% and at the same time to reduce the *rmse* by about 1.5%. The next step in our research will be verification of this method for other learning models than k-NN and for multi-objective evolutionary instance selection. We believe that this approach can be useful also to other problems, where the search space is big and no significant interactions occur between remote parameters, especially in the cases where the optimization algorithm cannot run too long, because of possible over-fitting.

Acknowledgements. This work was supported by Polish National Science Center (NCN) grant “Evolutionary Methods in Data Selection” No. 2017/01/X/ST6/00202.

References

1. Tallón-Ballesteros, A.J., Riquelme, J.C.: Data cleansing meets feature selection: a supervised machine learning approach. In: Ferrández Vicente, J.M., Álvarez-Sánchez, J.R., de la Paz López, F., Toledo-Moreo, F.J., Adeli, H. (eds.) IWINAC 2015. LNCS, vol. 9108, pp. 369–378. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18833-1_39
2. Olvera-López, A., Carrasco-Ochoa, J., Martínez-Trinidad, F., Kittler, J.: A review of instance selection methods. *Artif. Intell. Rev.* **34**(2), 133–143 (2010)
3. Garcia, S., Derrac, J., Cano, J.R., Herrera, F.: Prototype selection for nearest neighbor classification: taxonomy and empirical study. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(3), 417–435 (2012)
4. Rusiecki, A., Kordos, M., Kamiński, T., Greń, K.: Training neural networks on noisy data. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2014. LNCS (LNAI), vol. 8467, pp. 131–142. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07173-2_13
5. Kordos, M.: Optimization of evolutionary instance selection. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2017. LNCS (LNAI), vol. 10245, pp. 359–369. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59063-9_32
6. Kordos, M., Wydrzyński, M., Lapa, K.: Obtaining pareto front in instance selection with ensembles and populations. In: Rutkowski, L., Scherer, R., Korytkowski, M., Pedrycz, W., Tadeusiewicz, R., Zurada, J.M. (eds.) ICAISC 2018. LNCS (LNAI), vol. 10841, pp. 438–448. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-91253-0_41
7. Merelo, J.J., et. al.: There is noisy lunch: a study of noise in evolutionary optimization problems. In: 7th International Joint Conference on Computational Intelligence (IJCCI), pp. 261–268 (2015)
8. Antonelli, M., Ducange, P., Marcelloni, F.: Genetic training instance selection in multiobjective evolutionary fuzzy systems: a coevolutionary approach. *IEEE Trans. Fuzzy Syst.* **20**(2), 276–290 (2012)
9. Tsai, C.-F., Eberle, W., Chu, C.-Y.: Genetic algorithms in feature and instance selection. *Knowl.-Based Syst.* **39**, 240–247 (2013)
10. Cano, J.R., Herrera, F., Lozano, M.: Instance selection using evolutionary algorithms: an experimental study. In: Pal, N.R., Jain, L. (eds.) *Advanced Techniques in Knowledge Discovery and Data Mining. Advanced Information and Knowledge Processing*, pp. 127–152. Springer, London (2005). https://doi.org/10.1007/1-84628-183-0_5
11. Derrac, J., et al.: Enhancing evolutionary instance selection algorithms by means of fuzzy rough set based feature selection. *Inf. Sci.* **186**, 73–92 (2012)
12. Goldberg, D.: *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, Boston (1989)
13. Lobo, F.G., Lima, C.F., Michalewicz, Z.: *Parameter Setting in Evolutionary Algorithms. Studies in Computational Intelligence*, vol. 54. Springer, Heidelberg (2007). <https://doi.org/10.1007/978-3-540-69432-8>
14. Deb, K.: *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley, Hoboken (2001)
15. Aggarwal, C.C., Reddy, C.K.: *Data Clustering: Algorithms and Applications*. Chapman and Hall/CRC, Boca Raton (2013)

16. Rosales-Pérez, A., García, S., Gonzalez, J.A.: An evolutionary multiobjective model and instance selection for support vector machines with pareto-based ensembles. *IEEE Trans. Evol. Comput.* **21**(6), 863–877 (2017)
17. Escalante, H.J., et al.: MOPG: a multi-objective evolutionary algorithm for prototype generation. *Pattern Anal. Appl.* **20**(1), 33–47 (2017)
18. Gong, D., Zhou, Y.: Multi-population genetic algorithms with space partition for multi-objective optimization problems. *IJCSNS Int. J. Comput. Sci. Netw. Secur.* **6**, 52–58 (2006)
19. Ali, F.A., Ahmed, N.N.: Differential evolution algorithm with space partitioning for large-scale optimization problems. *Intell. Syst. Appl.* **11**, 49–59 (2015)
20. Arnaiz-González, Á., Díez-Pastor, J.F., Rodríguez, J.J., García-Osorio, C.: Instance selection for regression: adapting DROP. *Neurocomputing* **201**, 66–81 (2016)
21. Alcalá-Fdez, J., et al.: KEEL Data-Mining Software Tool and Data Set Repository (2017). <http://sci2s.ugr.es/keel/datasets.php>



Identification of Individual Glandular Regions Using LCWT and Machine Learning Techniques

José Gabriel García¹(✉), Adrián Colomer¹, Valery Naranjo¹,
Francisco Peñaranda¹, and M. Á. Sales²

¹ Instituto de Investigación e Innovación en Bioingeniería (I3B),
Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain
jogarpa70i3b.upv.es

² Servicio de Anatomía Patológica, Hospital Clínico Universitario de Valencia,
Valencia, Spain

Abstract. A new approach for the segmentation of gland units in histological images is proposed with the aim of contributing to the improvement of the prostate cancer diagnosis. Clustering methods on several colour spaces are applied to each sample in order to generate a binary mask of the different tissue components. From the mask of lumen candidates, the Locally Constrained Watershed Transform (LCWT) is applied as a novel gland segmentation technique never before used in this type of images. 500 random gland candidates, both benign and pathological, are selected to evaluate the LCWT technique providing results of Dice coefficient of 0.85. Several shape and textural descriptors in combination with contextual features and a fractal analysis are applied, in a novel way, on different colour spaces achieving a total of 297 features to discern between artefacts and true glands. The most relevant features are then selected by an exhaustive statistical analysis in terms of independence between variables and dependence with the class. 3.200 artefacts, 3.195 benign glands and 3.000 pathological glands are obtained, from a data set of 1.468 images at 10x magnification. A careful strategy of data partition is implemented to robustly address the classification problem between artefacts and glands. Both linear and non-linear approaches are considered using machine learning techniques based on Support Vector Machines (SVM) and feedforward neural networks achieving values of sensitivity, specificity and accuracy of 0.92, 0.97 and 0.95, respectively.

Keywords: Machine learning · Multilayer perceptron
Support vector machine · Locally constrained watershed transform
Gland unit identification · Histological prostate image

1 Introduction

The definitive diagnostic procedure to detect prostate cancer is the visual examination of biopsy samples stained with hematoxylin and eosin (H&E). From these

digital images, pathologists assign a score according to the Gleason classification system [1], such that Gleason grades 1 and 2 correspond to benign samples, whereas grades 3, 4 and 5 correspond to malignant samples. Thereby, histological images are essential for the physicians to make a reliable and accurate diagnosis of the patient. However, this manual task is time-consuming, tedious and subjective, which results in high rates of discordance between different pathologists.

Several studies on computer-aided Gleason grading have been developed to help the pathologists and to reduce the subjectivity level. The standard procedure starts by segmenting some individual glands, and then, these segmented glands are classified into their corresponding grades through a feature extraction step [2–4]. However, a recent study [5] claimed that to achieve significant improvements in the Gleason score discrimination, it is essential to perform an initial accurate segmentation of individual glandular regions like the one shown in Fig. 1. Motivated by such statement, this paper focuses on the identification and segmentation of each gland unit.

The proposed model to segment glands previously requires the accurate detection of lumens, which can be often confused with an artefact by its white colour. Nevertheless, an artefact is not surrounded by cytoplasm and nuclei components [3], as can be observed in Fig. 1. In this study, all lumen candidates are segmented, and then, a classification stage is carried out by means of different machine learning techniques widely used in the literature [4], in order to discern between artefacts and true glands.

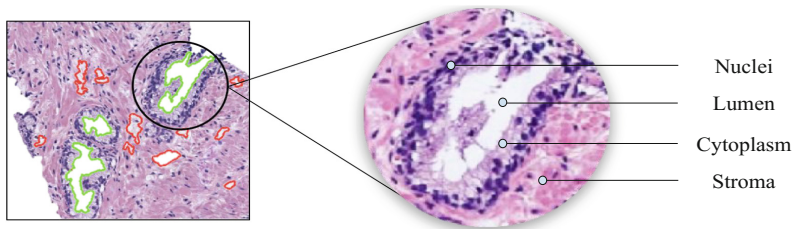


Fig. 1. The left image shows examples of lumens (green) and artefacts (red). The right image corresponds to an individual glandular region. (Color figure online)

2 Methodology

2.1 Material

The data set used in this paper includes 854 benign and 614 grade 3 pathological images, according to the Gleason scale, at 10x magnification with an image size of $1.024 \times 1.024 \times 3$ pixels, providing a total of 3.200 artefacts, 3.195 benign glands and 3.000 pathological glands. The samples come from a private database of the *Hospital Clínico Universitario de Valencia*, and they are characterised by a high variability both in shape and size of glands, as well as colour and sharpness of tissues, in order to provide a robust and consistent model.

2.2 Clustering

A clustering algorithm based on the *k-means* technique is carried out on several colour spaces. In particular, it is performed in Red, Green and Blue components from RGB colour spaces, Cyan from CMYK and Saturation from HSV, with the aim of grouping the pixels into four clusters: lumen, stroma, cytoplasm and nuclei, as shown in Fig. 2. In this way, each pixel is assigned to the cluster label with the closest centroid.

Notably, the resulting image of lumen candidates provides a large quantity of artefacts (see Fig. 2). For this reason, a filtering operation based on mathematical morphology, *area opening* [6], is used to remove the connected components whose number of pixels is smaller than a specific threshold $s = 40$ pixels.

The obtained masks are used as inputs in the following segmentation and feature extraction steps to discriminate between artefacts and glands.

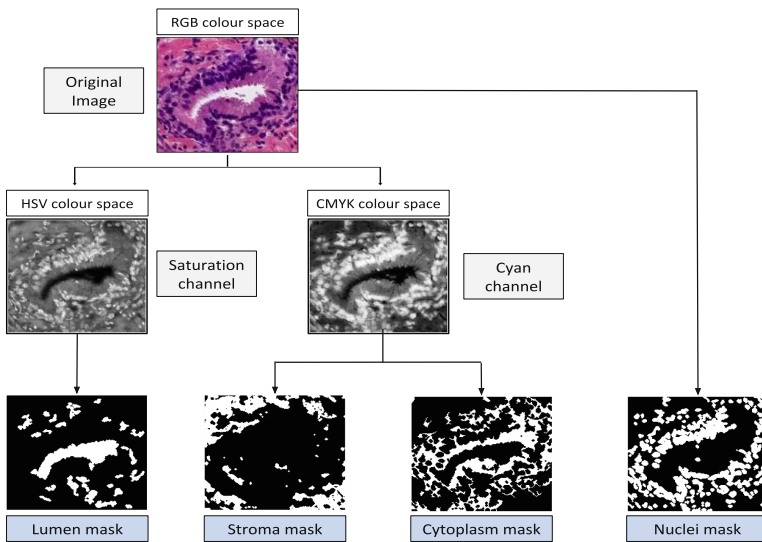


Fig. 2. Clustering procedure to obtain the different component masks by applying a transformation from RGB to CMYK and HSV colour spaces.

2.3 Gland Candidates Identification

Unlike the work presented in [2], where the gland boundaries are defined by the limits of the cytoplasm structures, in this paper a gland unit is described by its epithelial nuclei layer, similarly to the studies presented in [3] and [7] and according to the medical literature exposed in them. The three aforementioned segmentation approaches are shown in Fig. 3 to compare them.

One of the most remarkable novelties of this paper lies in the development of a robust segmentation method, based on the Locally Constrained Watershed

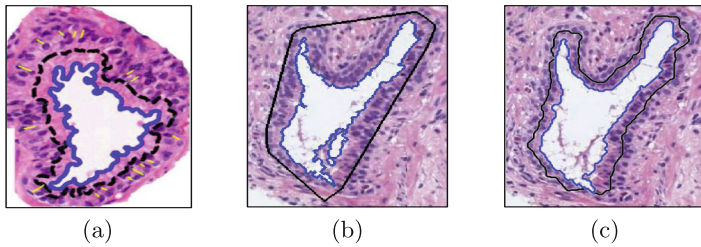


Fig. 3. Comparison of segmentation approaches proposed by different authors, where the blue line delimits the lumen; and the black line, the gland. (a) Example of segmented gland published in [2]. (b) Segmented gland with the code that the author proposes in [3]. (c) Same segmented gland with the method proposed in this study.

Transform (LCWT), able to address the delimitation of prostate glands in spite of its frontier (nuclei) remain open [8]. LCWT requires as inputs parameters: (1) internal markers, defined by the lumen candidates mask; (2) external markers, described by the pixels associated to the stroma mask; (3) input image, defined by the nuclei mask; and (4) Structurant Elements (SE) for both markers, which specifies the step size. The algorithm is based on the fact that each marker analyses the pixels of its neighbourhood trying to incorporate them into its region. The progress in the search for the integration of new pixels is carried out such that the higher the SE is, the more pixels are incorporated. On the other hand, the nuclei (input image) act as a restriction to the markers progress. Specifically, the expansion of a marker stops when the size of its SE is greater than the distance between adjacent nuclei. In that case, the external marker can not progress and a closed line of segmentation is defined by the distance between adjacent nuclei when both markers get into contact. Thus, the segmentation is performed over the nuclei epithelial layer around the gland (see Fig. 4).

The detailed process is applied to the images under study of $1.024 \times 1.024 \times 3$ pixels to obtain the segmentation of each gland candidate (see Fig. 5).

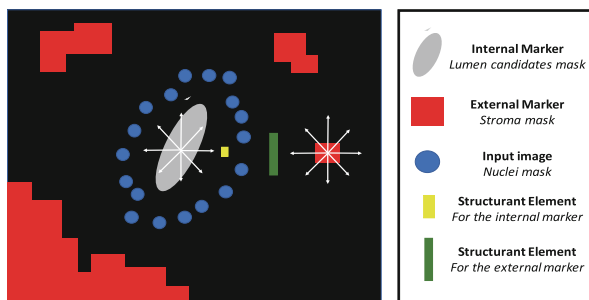


Fig. 4. Illustrative example to explain visually the performance of LCWT technique.

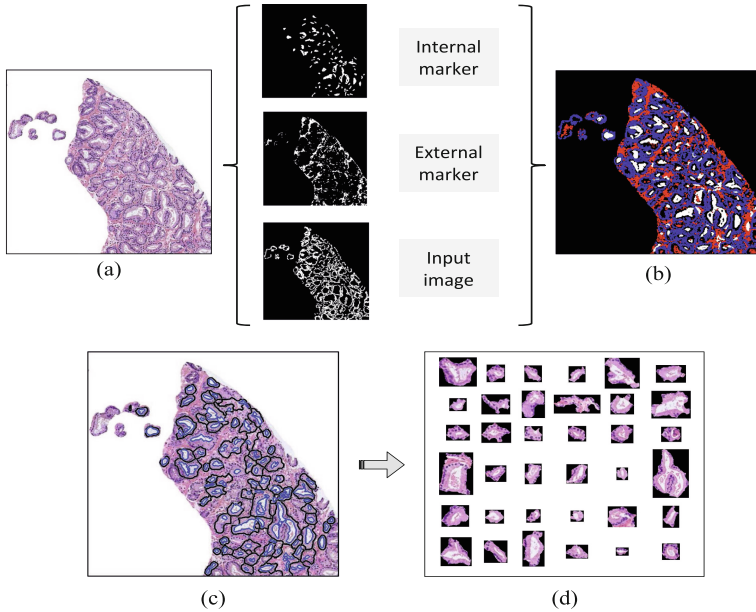


Fig. 5. (a) Original image. (b) Input parameters to the LCWT algorithm. (c) Image of $1.024 \times 1.024 \times 3$ pixels where all possible glands (black line) have been segmented from each lumen candidate (blue line). (d) All gland candidates automatically segmented by means the LCWT technique. (Color figure online)

2.4 Extraction and Selection of Features

Once all candidates to gland are segmented, it is necessary to implement a classification stage to discard the artefacts. In this way, a future grading Gleason could be more efficient because the characterisation of glandular patterns would be carried out from well-defined glands. Therefore, in this work, several descriptors are computed in order to distinguish between artefacts and glands.

On the one hand, shape descriptors are used to extract the structural and morphological information of the different components relative to the gland units and to the lumen, nuclei and cytoplasm structures. This approach was applied in some studies that also focused on the artefacts detection [7]. The metrics computed for the gland units and the lumens are: area, compactness, convex area, convex hull, eccentricity, ellipsoid diameter, extent, orientation, perimeter, roundness and solidity. The features related to the nuclei and the cytoplasm structures make reference to the density of pixels present in the segmented region.

On the other hand, in the same way that in [9], textural descriptors related to the grey-level co-occurrence matrix (GLCM) and Local Binary Patterns (LBP) are applied in order to extract the local texture information of the images. The computed metrics from the GLCM are: contrast, correlation, energy, homogeneity, mean and standard deviation. Regarding LBP, local histograms are used as feature vectors. In particular, 10-bin $LBP_{P,R}^{riu2}$ histograms uniformly invariant

to rotation transforms [10] and 10-bin $LBP_{P,R}^{riu2}$ histograms invariant against to local changes of contrast ($VAR_{P,R}$) [11].

Regarding the fractal analysis, in [12] the fractal dimension (FD) is used to analyse intensity and texture variations in certain regions of interest. In this work, the Hurst exponent is extracted over five directions (0° , 30° , 45° , 60° and 90°) with the aim of determining whether the data follow random or similar patterns. With the Hurst exponent, it is possible to analyse the complexity and the roughness character of the elements.

Finally, contextual features are also computed, similarly to [7], to consider the information around each gland candidate. In this way, relations of distances, shapes and other similarities between the individual gland and its context are taken into account.

It is important to remark that the textural descriptors and features based on fractals are applied to three colour spaces: cyan, hematoxylin and eosin. The two last channels are obtained from a method known as “colour deconvolution” that allows to separate the contributions of each staining in the H&E histopathological images [13].

From the 297 variables that compose the learning instance, an in-depth statistical analysis is performed in order to address the feature selection stage. This process allows taking into account only the most relevant features, in terms of discriminatory ability and independence between variables. First, a *Kolmogorov-Smirnov test* is used to check the normality of variables. A comparison of means using the *Student’s t-test* method or a comparison of medians using the *Mann-Whitney U test* method is performed depending on whether the variable under study follows a normal distribution $N(0, \sigma)$ or not. Thus, the independence of each variable concerning the class allows to determine the discriminatory capability of the variables. On the other hand, the independence between variables is assessed by calculating the correlation coefficient (R) to discard those variables with $p\text{-value} < 0.0001$ and $|R| > 0.95$. In this way, redundant information is avoided and finally, a total of 117 variables are obtained after the feature selection step applied over the whole data set.

2.5 Classification Strategy

An exhaustive data partition process is carried out in order to build robust models and to provide reliable results. To the best of the author knowledge’s, this work is the first that implements a type of data separation taking into account the medical history of the patient. In this study, all gland candidates that belong to a certain medical history must be contained in the same data set. Thereby, different samples of a certain patient can only be used to train or to test the models, but never for both in the same iteration. Considering these conditions, an external cross-validation technique with $k = 5$ folds is used to divide the samples into a test set and a training set. Moreover, the training set is subdivided into train and validation sets using an internal cross-validation with $v = 10$ folds. Thus, the method to separate the data guarantees that in each

of the five external iterations, the models are evaluated with samples belonging to new patients never “seen” before for the models.

Once the samples are divided into their corresponding folds, the classification problem is addressed from both linear and non-linear approaches applying popular machine learning methods such as Support Vector Machines (SVMs) with linear (LSVM) and quadratic (QSVM) kernels. SVMs are non-parametric binary classifiers that build a hyperplane to divide the input space maximising the distance of the support vectors from the different classes [11]. On the other hand, an alternative parametric approach is also considered in this work by using a Feedforward Neural Network (FNN), a.k.a. multilayer perceptron (MLP), with one hidden layer of fifteen neurons and a function that updates the weight and bias values according to the scaled conjugate gradient method.

3 Results and Discussion

Regarding the segmentation problem, 500 random glands are manually segmented in order to evaluate the LCWT automatic segmentation technique by means of similarity measures, such as the *Dice* and *Jaccard* coefficients. Some qualitative results are shown in Fig. 6. The hypothesis is that the results achieved from 500 glands can be inferred for all population (6.195 glands in this case).

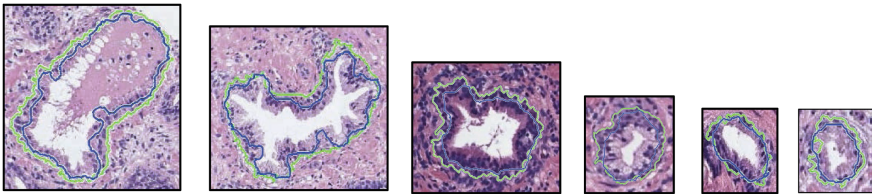


Fig. 6. Examples of benign and pathological glands of different sizes, manually and automatically segmented in green and blue, respectively. (Color figure online)

Attending to quantitative results, the LCWT method provides a *Dice* coefficient of 0.8491 and a *Jaccard* coefficient of 0.7378, from 500 randomly selected glands. The *Jaccard* index is compared with the results provided by other authors in the state of the art that also consider the problem of artefacts (see Table 1). Note the data set of images used to compare is different, so the results of the segmentation can’t definitively determine the best method.

Results of the classification problem are provided to determine whether, from the segmented gland candidates with the LCWT method, it is possible to successfully discern between artefacts and glands, based on the features detailed in Sect. 2.4. Different figures of merit to evaluate the robustness of the three learned models are shown in Table 2.

In general, all classifiers present a similar behaviour, but MLP provides a slight outperforming over the rest. Even though it is not possible to make a direct

Table 1. Comparison of segmentation methods with respect to the state of the art.

	Naik et al. [2]	Nguyen et al. [7]	LCWT method
<i>Jaccard</i> coefficient	0.43	0.66	0.7378 \pm 0.1080

Table 2. Mean and standard deviation of the classification results. (PPV: Positive Predictive Value, NPV: Negative Predictive Value; AUC: Area Under the ROC Curve).

	LSVM	QSVM	MLP
Sensitivity	0.9202 \pm 0.0107	0.9336 \pm 0.0112	0.9236 \pm 0.0187
Specificity	0.9687 \pm 0.0274	0.9581 \pm 0.0328	0.9696 \pm 0.0240
PPV	0.9404 \pm 0.0451	0.9202 \pm 0.0587	0.9410 \pm 0.0417
NPV	0.9564 \pm 0.0129	0.9633 \pm 0.0105	0.9589 \pm 0.0127
F1Score	0.9300 \pm 0.0270	0.9263 \pm 0.0332	0.9321 \pm 0.0294
Accuracy	0.9517 \pm 0.0209	0.9492 \pm 0.0232	0.9539 \pm 0.0212
AUC	0.9882 \pm 0.0074	0.9877 \pm 0.0090	0.9892 \pm 0.0080

comparison because there is no public data base, MLP results are examined in contrast to other studies of the state of the art, as shown in the Table 3.

Table 3. Comparison of classification results with respect to the state of the art.

	Naik et al. [2]	Nguyen et al. [7]	MLP classifier
Classification accuracy	0.78 \pm 0.09	0.93 \pm 0.04	0.9539 \pm 0.0212

4 Conclusions and Future Work

In this paper, a new method of individual glands segmentation is proposed in order to accurately detect the relevant information about the regions of interest. Several kinds of features are extracted to identify the characteristic patterns of the true glands. Finally, a classification stage based on machine learning techniques is carried out to demonstrate that, from the segmented gland candidates through the LCWT method, it is possible to discern between artefacts and true glands. The main lines of future research consist of addressing the grading Gleason by characterising the glands obtained from the LCWT segmentation method and performing the optimisation of the MLP classifier parameters.

Acknowledgements. This work has been funded by the Ministry of Economy, Industry and Competitiveness under the SICAP project (DPI2016-77869-C2-1-R). The work of Adrián Colomer has been supported by the Spanish FPI Grant BES-2014-067889. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

1. Gleason, D.F.: Histologic grading and clinical staging of prostatic carcinoma. In: *Urologic Pathology* (1977)
2. Naik, S., Doyle, S., Feldman, M., Tomaszewski, J., Madabhushi, A.: Gland segmentation and computerized gleason grading of prostate histology by integrating low-, high-level and domain specific information. In: *MIAAB Workshop*, pp. 1–8 (2007)
3. Nguyen, K., Sabata, B., Jain, A.K.: Prostate cancer grading: gland segmentation and structural features. *Pattern Recogn. Lett.* **33**(7), 951–961 (2012)
4. Kwak, J.T., Hewitt, S.M.: Multiview boosting digital pathology analysis of prostate cancer. *Comput. Methods Programs Biomed.* **142**, 91–99 (2017)
5. Ren, J., Sadimin, E., Foran, D.J., Qi, X.: Computer aided analysis of prostate histopathology images to support a refined gleason grading system. In: *SPIE Medical Imaging, International Society for Optics and Photonics*, p. 101331V (2017)
6. Soille, P.: *Morphological Image Analysis: Principles and Applications*. Springer, Berlin (2013)
7. Nguyen, K., Sarkar, A., Jain, A.K.: Structure and context in prostatic gland segmentation and classification. In: Ayache, N., Delingette, H., Golland, P., Mori, K. (eds.) *MICCAI 2012. LNCS*, vol. 7510, pp. 115–123. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33415-3_15
8. Beare, R.: A locally constrained watershed transform. *IEEE Trans. Pattern Anal. Mach. Intell.* **28**(7), 1063–1074 (2006)
9. Gertych, A., et al.: Machine learning approaches to analyze histological images of tissues from radical prostatectomies. *Comput. Med. Imaging Graph.* **46**, 197–208 (2015)
10. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(7), 971–987 (2002)
11. Guo, Z., Zhang, L., Zhang, D.: A completed modeling of local binary pattern operator for texture classification. *IEEE Trans. Image Process.* **19**(6), 1657–1663 (2010)
12. Huang, P., Lee, C.: Automatic classification for pathological prostate images based on fractal analysis. *IEEE Trans. Med. Imaging* **28**(7), 1037–1050 (2009)
13. Ruifrok, A.C., Johnston, D.A., et al.: Quantification of histochemical staining by color deconvolution. *Anal. Quant. Cytol. Histol.* **23**(4), 291–299 (2001)



Improving Time Series Prediction via Modification of Dynamic Weighted Majority in Ensemble Learning

Marek Lóderer^(✉), Peter Pavlík, and Viera Rozinajová

Faculty of Informatics and Information Technologies,
Slovak University of Technology, Ilkovičova 2, 841 04 Bratislava, Slovakia
{marek_loderer, xpavlikp, viera.rozinajova}@stuba.sk

Abstract. In this paper, we explore how the modified Dynamic Weighted Majority (DWM) method of ensemble learning can enhance time series prediction. DWM approach was originally introduced as a method to combine predictions of multiple classifiers. In our approach, we propose its modification to solve the regression problems which are based on using differing features to further improve the accuracy of the ensemble. The proposed method is then tested in the domain of energy consumption forecasting.

Keywords: Ensemble learning · Dynamic weighted majority method
Time series prediction

1 Introduction

Time series predictions have indispensable importance for the human society and they have a great impact on many domains as well as on our everyday human activities. Predictions are used in many areas like industry, energetics, business, banking, weather forecasting, research, etc. Especially in energetics, accurate forecasts of the future values are crucial. Identifying the underlying patterns in the data is usually not a trivial task. In the last decades, researchers have introduced several prediction methods (predictors) to solve this problem [8].

Some of the predictors use mathematical and statistical calculations, like Linear Regression [9], ARIMA models or Exponential Smoothing [19]. Others are based on Machine learning, e.g. Support Vector Regression, Neural Networks and Random Forest [6, 7].

Each of these predictors can be successful at describing certain types of time series patterns but may become less accurate over time if the time series contains changes in concept characteristics known as Concept Drift. To solve this problem various learning methods such as Bagging [2], Boosting [17], Stacking [20] and numerous hybrid approaches [7, 10, 15, 21] have been developed.

The strength of ensembles lies in the fact that even if some of its predictors fail to predict the new pattern correctly, others with ability to predict accurately in

these changed conditions can compensate the overall prediction error. Therefore, highly diverse ensembles are effective in lowering the error after a concept drift occurs [13].

Another benefit of ensemble is the ability to adapt to changes by dynamically combining base members (predictors) according to their recent performance which increases the overall accuracy of the prediction [3].

Ensemble learning based on Dynamic Weighted Majority strongly employs dynamic combination of predictors along with predictors reweighting which is described and discussed in this paper.

This paper is organized as follows. In Sect. 2 we introduce the main concept of ensemble learning and related terms. In Sect. 3 we present our proposed ensemble learning model based on modification of Dynamic Weighted Majority method for time series prediction. In Sect. 4 we present results of experimental evaluation and our conclusions can be found in Sect. 5.

2 Ensemble Learning

In general, the main principle of ensemble learning is based on a proper combination of results of different base models (predictors or classifiers) that can create a more accurate result in comparison to the result provided by the best individual model [12].

Ensemble learning consists of three main subprocesses: ensemble generation, ensemble pruning and ensemble integration. In ensemble generation, a set of diverse base prediction models are trained. The required level of diversity of base models can be achieved by three main approaches - data, parameter and structural diversity [16].

Ensemble pruning is used to eliminate redundant and high erroneous models to increase the overall accuracy of final prediction. The pruning can be performed by various ranking, search or partitioning-based methods [12]. This part of the ensemble learning is optional.

The last subprocess, ensemble integration, provides combination of results of prediction models. The combination is usually carried out as a linear combination, where the weights are calculated by numerous approaches e.g. a simple mean, an inverse value of prediction model performance or more complex weighted schemes based on optimization algorithms [5, 21].

Several types of ensemble based on Outperformance method [1] or Dynamic Weighted Majority method [10] combine outputs of currently generated predictors taking into account values of past weights and predictors errors. This additional information helps ensemble to overcome high fluctuation of weights in noisy and quickly changing data.

2.1 Dynamic Weighted Majority

As mentioned earlier, Dynamic Weighted Majority (DWM) is an ensemble method which uses more complex weighted schemes. It was first described by

Kolter and Maloof in 2003 [10]. It is based on an older Weighted Majority algorithm from Littlestone and Warmuth which gives individual experts (prediction methods of the ensemble) weights, modifies them according to their performance and generates final prediction by combining the predictions of the experts with consideration to their weights [11].

While original Weighted Majority algorithm works with a static set of experts, the Dynamic Weighted Majority can add or remove a number of experts based on their performance. Thanks to this added feature, the ensemble can successfully predict even in a changing environment with occurrence of Concept Drift [10].

The original algorithm works with a set of experts with corresponding weights. Each iteration of the algorithm starts by calculating a global prediction of the current ensemble. The global prediction is obtained by combining predictions of all experts proportionally to their weights. The algorithm obtains a prediction from each member of the ensemble and adds its weight to the sum for the corresponding output class. The class with the highest weight is then set as the global prediction of the ensemble.

If the prediction does not match the sample label, the weights of incorrect experts are lowered by predefined multiplicative factor from interval (0,1). If a weight of any expert is lower than a predefined threshold value, then the expert is removed from the ensemble.

A new expert is trained and added every time the global prediction is incorrect. At the end of each iteration, the weights of the experts are normalized to add up to 1. Otherwise, the resulting prediction would be biased.

2.2 Modification for Regression

The Dynamic Weighted Majority was originally created for classification but the base idea of keeping a dynamic set of experts is applicable for regression as well. However, changes must be made in the process of evaluating experts' performance, modification of their weights and experts' replacement.

When solving classification problems, evaluating the correctness of the prediction is quite straightforward. On the other hand, a result of regression is a number from a continuous interval where the accuracy of prediction has to be measured by certain metrics. That means we cannot easily decrease the weight of an expert by a constant factor when its result is incorrect.

Subsequently, due to the property of regression problems another step of the algorithm cannot be directly used - adding a new expert when the global prediction is incorrect. A possible solution is setting a threshold to specify the highest acceptable error of a prediction method. But setting the threshold is very domain-specific and often undesirable. A better solution for general use is a constant size of the ensemble which means a new expert is added to the ensemble only in case when another expert has been removed.

In 2016, a paper *Prediction of Power Load Demand Using Modified Dynamic Weighted Majority Method* by Radoslav Nemeč et al. applied the Dynamic Weighted Majority on the regression problem of predicting power load demand

[14]. In this paper, the problem of reducing expert weights was solved by introducing an error constant γ . The error constant reduces the weights of experts who achieve higher error and increases weights of experts with more precise results. The problem of adding new experts was solved by an constant size of the ensemble.

3 Proposed Algorithm

In this paper, we propose a general version of Dynamic Weighted Majority for regression and time series prediction. Our method is based on the modification for regression, which was mentioned in the previous chapter, but without the need for error constant γ , as it is very dependent on data. After removing this constant, we can use the error of prediction as a measure to determine how much we want to reduce the weight of an expert, as opposed to binary choice of reducing or not reducing the weight by a given constant. We believe this approach can increase the accuracy of the ensemble.

Algorithm 1. Dynamic Weighted Majority for regression

```

 $\{\vec{x}, y\}_n^1$  : training data composed of attribute vector and class label
 $\beta$  : expert weight lowering factor,  $0 \leq \beta > 1$ 
 $\theta$  : threshold weight value for expert removal
 $p$  : period of expert replacement
 $\{e, w\}_m^1$  : set of experts and their weights
 $A, \lambda \in \{1, \dots, c\}$  : global and local predictions
 $m$  : size of the expert set in ensemble
 $\varepsilon_{1, \dots, m}$  : prediction errors
1:  $e_m \leftarrow CreateExpertSet()$ 
2:  $w_{1, \dots, m} \leftarrow 1/m$ 
3: for  $i \leftarrow 1, \dots, n$  do
4:    $A \leftarrow 0$ 
5:   for  $j \leftarrow 1, \dots, m$  do
6:      $\lambda \leftarrow Predict(e_j, \vec{x}_i)$ 
7:      $\varepsilon_j \leftarrow MAPE(\lambda, y_i)$ 
8:      $A \leftarrow A + \lambda * w_j$ 
9:    $w \leftarrow LowerWeights(w, \varepsilon)$ 
10:  if  $i \bmod p = 0$  then
11:    for  $j \leftarrow 1, \dots, m$  do
12:      if  $w_j < \theta$  then
13:         $e, w \leftarrow ReplaceWithNewExpert()$ 
14:   $w \leftarrow NormalizeWeights(w)$ 
15:  output  $A$ 

```

The algorithm of the proposed DWM method starts by creating a set of m different experts with equal weights (lines 1–2). An iteration starts by obtaining

a prediction of the training sample for all experts. These local predictions are multiplied by the weight w_j of an expert j and added to global prediction Λ (lines 3–8). Subsequently, the prediction errors are calculated for each expert and saved into a vector. The prediction error of each expert is calculated by Mean Absolute Percentage Error metric (1)

$$\text{MAPE} = \frac{100}{n} \sum_{t=1}^n \left| \frac{A_t - F_t}{A_t} \right|, \quad (1)$$

where n is the number of samples, A_t is the actual value and F_t is the predicted value.

Although, various error metrics can be used [18]. These accuracy values are transformed into a vector of multipliers from interval $\langle \beta, 1 \rangle$ which is used to lower the weights of experts proportionally to their performance on the last sample (line 9).

In our implementation, we achieve this by using the transformation function (2) which assigns the lowest performing expert a multiplier of β and gradually increases the multipliers of other experts up to a theoretical maximum of 1 for perfect prediction:

$$\text{mult}_i = \left(1 - \frac{\varepsilon_i}{100} \right)^{\frac{1}{\log_{\beta} \left(1 - \frac{\max(\varepsilon)}{100} \right)}} \quad (2)$$

In case the expert replacement is allowed in this iteration (line 10), we check if any expert has weight lower than the threshold θ (lines 11–12). If that is the case, expert is removed from the ensemble and replaced by a different one with an initial starting weight (line 13). At the end of the iteration, weights of the experts are normalized so the sum is equal to 1 (line 14).

4 Evaluation

We evaluated the accuracy of our proposed DWM method on time series data containing electricity consumption measurements. We predict electricity consumption for the next 24 h and then we move the prediction window to the next day.

For testing of the proposed ensemble, a process of creating diverse experts is needed. We fulfill this requirement by creating structurally diverse experts based on different prediction methods. A pool of experts is created from commonly used time series prediction methods, namely: Autoregressive Integrated Moving Average (ARIMA), Random Forest (RF), Feed-forward Neural Network with a single hidden layer and lagged inputs (NN) and Support Vector Regression (SVR). Each method is trained on a window of four weeks training data prior to the prediction date. However, we do not use these methods directly on the time series data.

Before prediction, the time series is split into seasonal, trend and remainder component by the *Seasonal and Trend decomposition using Loess (STL)* [4].

Each expert in ensemble consists of one seasonal, trend and reminder component, where each component can be computed by different prediction method. By this approach we can increase the number of possible experts up to n^3 where n is the number of used base prediction methods.

The ensemble starts with a given number of random experts with equal weights. The weights are modified in each iteration by formula (2) and subsequently if a weight of any expert falls below the threshold θ , then it is replaced by another expert from currently uninvolved experts. In our implementation, we randomly pick one of the experts included in the ensemble and mutate it by changing one of its three components to create a new expert.

4.1 Data

The proposed ensemble model was evaluated on two electricity consumption datasets. Both datasets contain energy measurements from households as well as from enterprises. The first dataset consists of time series measurements with 60 min period from Toronto region, Canada. The data are collected by the Independent Electricity System Operator¹. In our experiments, we used the sliding window approach to perform daily predictions for whole year 2011.

The second dataset consists of time series measurements with 30 min period from Australian Energy Market Operator². In the experiment, we used aggregated data from the state Tasmania. Daily predictions were computed on data from year 2009.

4.2 Results

In our experiments we tested prediction accuracy of the proposed ensemble. Since the ensemble has several configuration parameters that strongly affects the prediction outcome, at first we experimentally estimated optimal values for these parameters. The parameter β was set to 0.65, parameter θ to 0.5 and p was 1.

The parameter estimation was calculated on one whole year of previous measurements in Toronto (year 2010) and Tasmania (year 2008) datasets. This one-year period of previous data was also used to eliminate potential prediction error caused by randomness of initial experts in the ensemble, and to select appropriate ones. Another important aspect influencing prediction accuracy of the ensemble is the number of experts in it.

To put the results of the proposed DWM ensemble with scaled multipliers into perspective, we also measured accuracy of the DWM ensemble with constant weights multipliers based on the work of Radoslav Nemeč et al. [14].

In our first experiment we evaluated the prediction accuracy based on the number of experts in the ensemble. An error metric MAPE was used to evaluate the prediction accuracy. Figures 1 and 2 show the development of the average

¹ <http://www.ieso.ca/>.

² <http://www.aemo.com.au>.

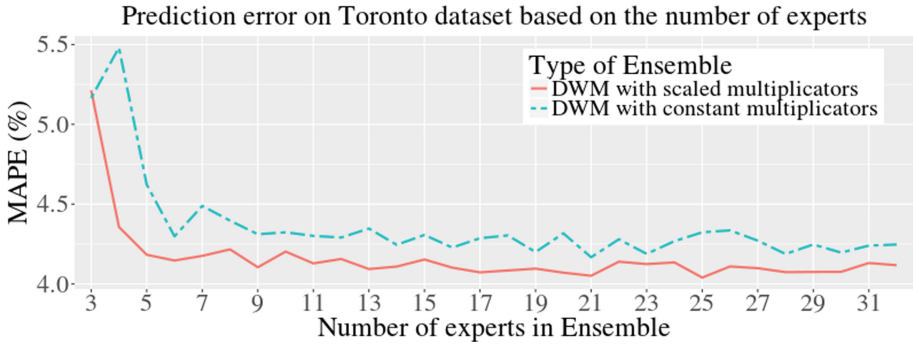


Fig. 1. Accuracy comparison of DWM ensembles with scaled and constant multipliers based on a number of experts in the ensemble measured on the electricity consumption dataset from Toronto region in Canada.

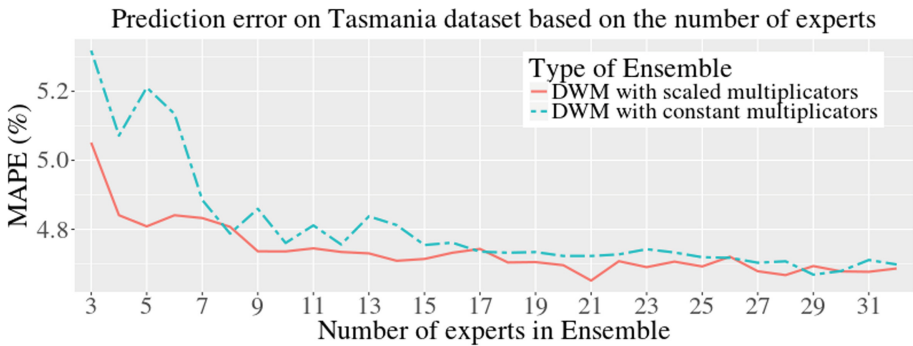


Fig. 2. Development of prediction accuracy of DWM ensembles with scaled and constant multipliers based on a number of experts in the ensemble evaluated on the Tasmania dataset.

daily prediction error of tested ensembles based on different number of experts. The results show that a reasonable number of experts in the ensemble is about 5 to 9. A higher number of experts improves results only slightly and it increases the computational complexity. The results also show that the proposed DWM ensemble with scaled multipliers obtained lower prediction error in comparison to the DWM ensemble with constant weights multipliers in almost all tested cases.

The second experiment was designed to show prediction accuracy of the tested DWM ensembles and 10 best experts. Results displayed in Figs. 3 and 4 show average daily prediction error measured on Toronto dataset for time period from 1.1.2011 to 31.12.2011 and Tasmania dataset from 1.1.2009 to 31.12.2009. In Toronto dataset, the number of experts in ensemble was set to 25. In case of Tasmania dataset, we used 21 experts.

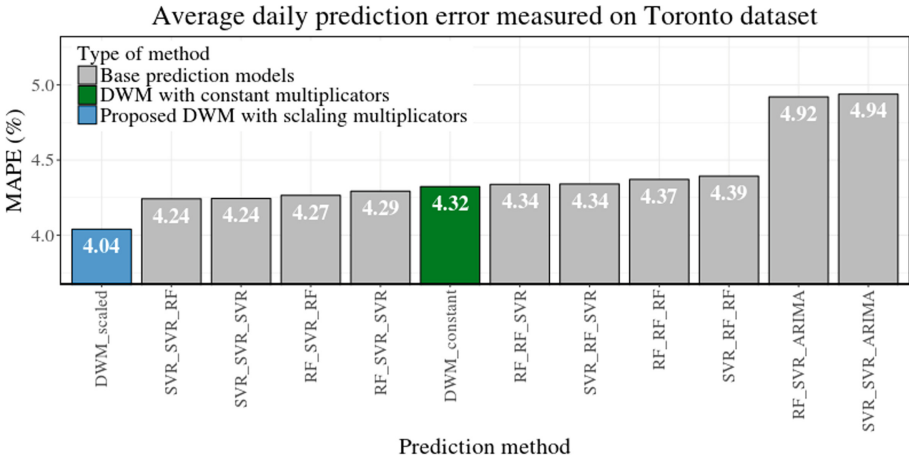


Fig. 3. Comparison of prediction error of DWM ensembles and ten best prediction methods (experts) on Toronto dataset in year 2011 displayed in ascending order.

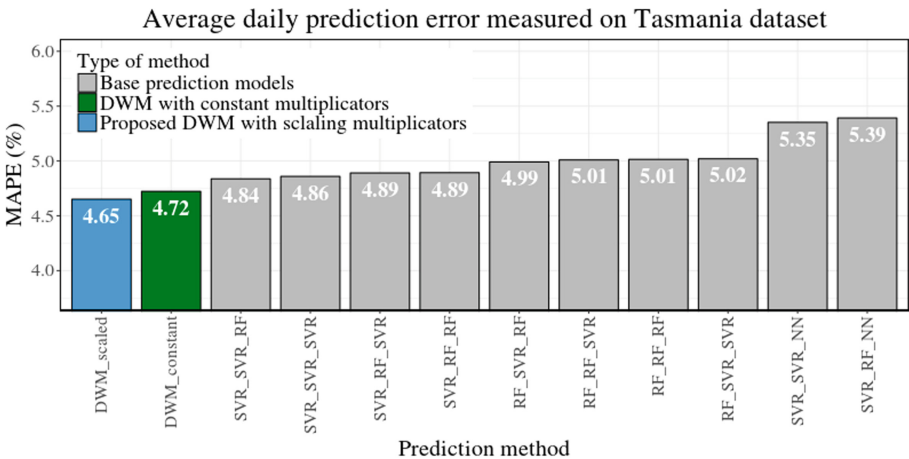


Fig. 4. Results representing prediction error of tested DWM ensembles and their best performing member methods (experts) on Tasmania dataset in year 2009 displayed in ascending order.

As mentioned previously, each expert is composed of seasonal, trend and reminder component of the time series that is predicted by individual prediction method and combined to create final prediction for the next day. The name of an expert consists of abbreviations of used prediction methods. The position of abbreviation in the expert name represents the seasonal, trend and reminder component.

According to the results the proposed DWM ensemble with scaled multipliers outperformed DWM ensemble with constant multipliers as well as

best expert on both datasets. In Tasmania dataset, several experts obtained even better prediction results than DWM ensemble with constant multipliers.

The results also show that the majority of the seasonal, trend and reminder components of the best predicting experts in both testcases were predicted mainly by Support Vector Regression and Random Forest.

5 Conclusion

Our proposed modification of Dynamic Weighted Majority with scaling multipliers for regression and time series prediction has proven to be a successful approach to combine multiple predictors (experts) into an accurate ensemble. It is especially useful if many predictors with various accuracies are available, as it is able to identify the best performing predictors and omit the underperforming ones even in a changing environment.

We also compared our proposed ensemble with another DWM ensemble on two publicly available electricity consumption datasets. According to the results our solution outperformed all base prediction methods as well as other tested ensemble in terms of prediction accuracy.

However, there is still room for improvement. The possible aim of future research is to find an optimal transformation of forecast errors into multipliers of experts' weights. Replacing the exponential scaling of errors shown above by other mathematical transformations would undoubtedly impact the precision of the ensemble and should be explored further. Another direction of future research could involve optimization of constant values β and θ used as the parameters of the ensemble.

Acknowledgment. This work was partially supported by the Slovak Research and Development Agency under the contract APVV-16-0213.

References

1. Adhikari, R., Agrawal, R.K.: Performance evaluation of weights selection schemes for linear combination of multiple forecasts. *Artif. Intell. Rev.* **42**(4), 529–548 (2012). <https://doi.org/10.1007/s10462-012-9361-z>
2. Breiman, L.: Bagging predictors. *Mach. Learn.* **24**(2), 123–140 (1996)
3. Cerqueira, V., Torgo, L., Oliveira, M., Pfahringer, B.: Dynamic and heterogeneous ensembles for time series forecasting. In: 2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA), pp. 242–251, October 2017. <https://doi.org/10.1109/DSAA.2017.26>
4. Cleveland, R.B., Cleveland, W.S., McRae, J.E., Terpenning, I.: STL: a seasonal-trend decomposition procedure based on loess. *J. Off. Stat.* **6**(1), 3–73 (1990)
5. Ezzeddine, A.B., et al.: Using biologically inspired computing to effectively improve prediction models. *Int. J. Hybrid Intell. Syst.* **13**(2), 99–112 (2016)
6. Górriz, J.M., Puntónet, C.G., Salmerón, M., de la Rosa, J.J.G.: A new model for time-series forecasting using radial basis functions and exogenous data. *Neural Comput. Appl.* **13**(2), 101–111 (2004). <https://doi.org/10.1007/s00521-004-0412-5>

7. Halaš, P., Lóderer, M., Rozinajová, V.: Prediction of electricity consumption using biologically inspired algorithms. In: 2017 IEEE 14th International Scientific Conference on Informatics, pp. 98–103, November 2017
8. Hyndman, R.J., Athanasopoulos, G.: *Forecasting: Principles and Practice*. OTexts (2014)
9. Kantikoon, V., Kinnares, V.: The estimation of electrical energy consumption in abnormal automatic meter reading system using multiple linear regression. In: IEEE International Conference on Electrical Machines and Systems (ICEMS), pp. 826–830 (2013)
10. Kolter, J.Z., Maloof, M.A.: Dynamic weighted majority: an ensemble method for drifting concepts. *J. Mach. Learn. Res.* **8**(Dec), 2755–2790 (2007)
11. Littlestone, N., Warmuth, M.: The weighted majority algorithm. *Inf. Comput.* **108**(2), 212–261 (1994)
12. Mendes-Moreira, J., Soares, C., Jorge, A.M., Sousa, J.F.D.: Ensemble approaches for regression. *ACM Comput. Surv.* **45**(1), 1–40 (2012)
13. Minku, L.L., White, A.P., Yao, X.: The impact of diversity on online ensemble learning in the presence of concept drift. *IEEE Trans. Knowl. Data Eng.* **22**(5), 730–742 (2010)
14. Nemeč, R., Rozinajová, V., Lóderer, M.: Prediction of power load demand using modified dynamic weighted majority method. In: Świątek, J., Tomczak, J.M. (eds.) *ICSS 2016. AISC*, vol. 539, pp. 36–49. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-48944-5_4
15. Qiu, X., Zhang, L., Ren, Y., Suganthan, P.N., Amaratunga, G.: Ensemble deep learning for regression and time series forecasting. In: 2014 IEEE Symposium on Computational Intelligence in Ensemble Learning (CIEL), pp. 1–6, December 2014. <https://doi.org/10.1109/CIEL.2014.7015739>
16. Ren, Y., Zhang, L., Suganthan, P.N.: Ensemble classification and regression—recent developments, applications and future directions [review article]. *IEEE Comput. Intell. Mag.* **11**, 41–53 (2016)
17. Schapire, R.E.: The boosting approach to machine learning: an overview. In: Denison, D.D., Hansen, M.H., Holmes, C.C., Mallick, B., Yu, B. (eds.) *Nonlinear Estimation and Classification*. LNS, vol. 171, pp. 149–171. Springer, New York (2003). https://doi.org/10.1007/978-0-387-21579-2_9
18. Shcherbakov, M., Brebels, A., Shcherbakova, N., Tyukov, A., Janovsky, T., Kamaev, V.: A survey of forecast error measures. *World Appl. Sci. J.* **24**, 171–176 (2013)
19. Taylor, J.W., McSharry, P.E.: Short-term load forecasting methods: an evaluation based on european data. *IEEE Trans. Power Syst.* **22**(4), 2213–2219 (2007)
20. Ting, K.M., Witten, I.H.: Issues in stacked generalization. CoRR abs/1105.5466 (2011). <http://arxiv.org/abs/1105.5466>
21. Xiao, L., Wang, J., Hou, R., Wu, J.: A combined model based on data pre-analysis and weight coefficients optimization for electrical load forecasting. *Energy* **82**, 524–549 (2015)



Generalized Low-Computational Cost Laplacian Eigenmaps

J. A. Salazar-Castro^{1,2,3(✉)}, D. F. Peña³, C. Basante³, C. Ortega³,
L. Cruz-Cruz², J. Revelo-Fuelagán³, X. P. Blanco-Valencia⁴,
G. Castellanos-Domínguez¹, and D. H. Peluffo-Ordóñez⁴

¹ Universidad Nacional, sede Manizales, Manizales, Colombia
joasalazarca@unal.edu.co

² Corporación Universitaria Autónoma de Nariño, Pasto, Colombia

³ Universidad de Nariño, Pasto, Colombia

⁴ SDAS Research Group, Yachay Tech, Urcuquí, Ecuador

Abstract. Dimensionality reduction (DR) is a methodology used in many fields linked to data processing, and may represent a preprocessing stage or be an essential element for the representation and classification of data. The main objective of DR is to obtain a new representation of the original data in a space of smaller dimension, such that more refined information is produced, as well as the time of the subsequent processing is decreased and/or visual representations more intelligible for human beings are generated. The spectral DR methods involve the calculation of an eigenvalue and eigenvector decomposition, which is usually high-computational-cost demanding, and, therefore, the task of obtaining a more dynamic and interactive user-machine integration is difficult. Therefore, for the design of an interactive IV system based on DR spectral methods, it is necessary to propose a strategy to reduce the computational cost required in the calculation of eigenvectors and eigenvalues. For this purpose, it is proposed to use locally linear submatrices and spectral embedding. This allows integrating natural intelligence with computational intelligence for the representation of data interactively, dynamically and at low computational cost. Additionally, an interactive model is proposed that allows the user to dynamically visualize the data through a weighted mixture.

Keywords: Dimensionality reduction · Generalized methodology
Kernel approximations · Low-computational cost
Multiple kernel learning · Spectral methods

1 Introduction

The aim of dimensionality reduction (DR) is to extract a lower dimensional, relevant information from high-dimensional data, being then a key stage within the design of pattern recognition and data mining systems. Indeed, when using adequate DR stages, the system performance can be enhanced as well as the

data visualization can become more intelligible. The range of DR methods is diverse, including classical approaches such as classical multidimensional scaling (CMDS), which is based on distance preservation criteria [1]. Recent methods of DR are focused on the data topology preservation [2, 3]. Mostly such a topology is driven by graph-based approaches where data are represented by a non-directed and weighted graph. In this connection, the weights of edge graphs are certain pairwise similarities between data points, the nodes are data points, and a non-negative similarity (also affinity) matrix holds the pairwise edge weights. Spectral methods such as Laplacian eigenmaps (LE) [4] and locally linear embedding (LLE) [5] were the pioneer ones to incorporate similarity-based formulations. Spectral approaches for DR have been widely used in several applications such as relevance analysis [6], dynamic data analysis [7] and feature extraction [8, 9]. Because of being graph-driven methods and involving then similarities, spectral approaches can be easily represented by kernels [10], which means that a wide range of methods can be set within a kernel framework [11]. At the moment to choose a method, aspects such as nature of data, complexity, aim to be reached and problem to be solved should be taken into consideration. In this regard, as mentioned above, there exists a variety of DR spectral methods making the selection of a method a nontrivial task. Also, some problems may require the combination of methods so that the properties of different methods are simultaneously taken into account to perform the DR process and the quality of resultant embedded space is improved.

In this work, we explore the possibility to extend LE to a generalised methodology of spectral dimension reduction, for this purpose the weight matrix of the graph is considered as a representation of data similarities, which can also be represented as a kernel approximation. In this way, the methodology allows the spectral decomposition of the Laplacian obtained considering the kernel approximations of conventional DR methods. The experiments are carried out over well-known data sets, namely an artificial **Spherical shell** and a **Swiss roll** toy set. The DR performance is quantified by a scaled version of the average agreement rate between K -ary neighborhoods as described in [12]. The rest of this paper is organized as follows: Sect. 2 introduces the proposed a generalized methodology for spectral dimensionality reduction based on Laplacian eigenmaps is outlined. In Sect. 3, we shows the integration of locally linear landmarks to reduced computational cost. Section 4 presents the experimental setup and Sect. 5 shows the results and discussion. Finally, Sect. 6 gathers some final remarks as conclusions and future work.

2 A Generalized Methodology for Spectral Dimensionality Reduction

One nonlinear dimensionality reduction method is known as Laplacian eigenmaps (LE) which is based on graph theory and it is responsible for performing the spectral decomposition of Laplacian, a matrix that lies in a difference between a similarity matrix and its grade matrix. This method was widely explained in

[4]. To sum up, LE generates a low-dimensionality representation from a high-dimensional data in order to preserve the highest relations of proximity between the nearest points. LE works through a mapping of close entry patterns on close output values [13]. The input for LE is a symmetric positive semidefinite matrix \mathbf{W} of size $N \times N$ which contains the information of the original space. One of the features that makes DR methods versatile is that this kind of matrices are susceptible to be represented with kernel approximations. Hence, a new spectral methodology, called Kernel Laplacian Eigenmaps (KLE) is introduced. The modification made in the classical approach consist in finding the same Laplacian \mathbf{L} that is used in LE but changing the similarity matrix \mathbf{W} by a kernel matrix \mathbf{K} , then a modification of the grade matrix $\mathbf{D} = \text{Diag}(\mathbf{W}\mathbf{1}_N)$ by $\mathbf{D} = \text{Diag}(\mathbf{K}\mathbf{1}_N)$ is made, in consequence, the new Laplacian will be $\mathbf{L} = \mathbf{D} - \mathbf{K}$. The objective function definition of the minimization problem of LE is not affected and the solution will be $\lambda \mathbf{D}\mathbf{f} = \mathbf{L}\mathbf{f}$. Figure 1 illustrates the general diagram of the proposed methodology.

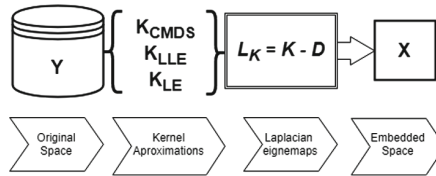


Fig. 1. A generalized methodology for spectral dimensionality reduction based on the LE method

3 Computational Cost Reduction Through Locally Linear Sub-matrixes (KLE + LLL)

In [14], an approximate solution for the computational cost problem is introduced in order to reduce time in the estimation of eigenvalues and eigenvectors, which are required by the spectral methods. The method is called locally linear submatrixes (LLL), where the selection of a number of sub matrixes $\tilde{\mathbf{Y}}_{D \times L}$ with $L \ll N$ is done in a process does a linear approximation of the input space in a local way around the sub matrixes in a nonlinear form from the global representation. Then, \mathbf{Z} is defined as a projection matrix which made a locally linear mapping given by $\mathbf{Y} \approx \tilde{\mathbf{Y}}\mathbf{Z}$. In this sense, the computational cost can be reduce by an alternative decomposition of eigenvalues and eigenvectors without a significant lost of information since the entire input space is considered for the construction of the similarity matrix. Since LLL assumed the existence of one local dependence between the sub matrixes points in the input space and in the embedded space, the projection matrix \mathbf{Z} can be used again in a linear mapping in low dimension as $\mathbf{X} \approx \tilde{\mathbf{X}}\mathbf{Z}$. In [14], it is possible to see the definition of the current spectral problem through an approximation by the relation parameter

d and L . Finally, whereas the solution is applied to LE (see Fig. 2), the affinity matrix \mathbf{A} is replaced by the Laplacian \mathbf{L} which contains one relationship of the kernel approximation \mathbf{K} of LE and matrix \mathbf{B} is replaced by the matrix \mathbf{D} of the same method. Then, the spectral problem of LLL applied to LE (LLL + LE) is expressed as $\operatorname{argmin}_x \operatorname{tr}(\mathbf{X}\mathbf{L}\mathbf{X}^T)$ s. t. $\mathbf{X}\mathbf{D}\mathbf{X}^T = \mathbf{I}$, and the approximate problem can be described by $\operatorname{argmin}_x \operatorname{tr}(\tilde{\mathbf{X}}\tilde{\mathbf{L}}\tilde{\mathbf{X}}^T)$ s. t. $\tilde{\mathbf{X}}\tilde{\mathbf{D}}\tilde{\mathbf{X}}^T = \mathbf{I}$, where $\tilde{\mathbf{L}} = \mathbf{Z}\mathbf{L}\mathbf{Z}^T$, $\tilde{\mathbf{D}} = \mathbf{Z}\mathbf{D}\mathbf{Z}^T \in \mathbb{R}^{L \times L}$ are the matrices in the reduced spectral problem. The solution of the approximate problem is defined by $\tilde{\mathbf{X}} = \tilde{\mathbf{U}}_d^T \tilde{\mathbf{D}}^{-\frac{1}{2}}$ and the trivial solution, $\tilde{\mathbf{L}}_1 = 0$, where $\lambda_1 = 0$, is discard.

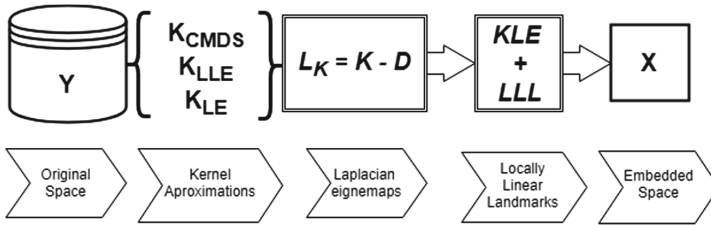


Fig. 2. The low-computational methodology based on locally linear landmarks

Algorithm 1. Pseudo-code for KLE+LLL implementation

1. **Verification 1:** Check whether the data consists of distances in pairs, if so skip to step 3.
2. **Verification 2:** If the data consists of vectors, **Do:**
 Compute all distances in pairs
End do
3. **Determine:** Choose the k neighborhoods or the ϵ spherical neighborhoods.
4. **Graph design:** Construct the corresponding graph and its proximity matrix.
5. **Obtaining the kernel K :** Build the K matrix using some kind of three kernel or a weighted mixture of these.
6. **Obtaining D :** Build the D matrix through $D = \text{Diag}(K\mathbf{1}_N)$.
7. **Computing the Laplaciano:** Computing $L = K - D$.
8. **Apply the LLL methodology:** Computing \tilde{L} .
9. **Eigenvalue decomposition:** Apply EVD to \tilde{L} .
10. **Obtaining the embedded space: Do**
 - a) Multiply the eigenvectors by $\tilde{D}^{1/2}$.
 - b) Transpose the result.
 - b) Determine the values associated with the smallest d eigenvalues without considering the last eigenvalue.

End

4 Experimental Setup

Kernels for DR: Three kernel approximations for spectral DR methods [10] are considered, CMDS, LLE and LE. CMDS kernel is the double centered distance matrix $\mathbf{D} \in \mathbb{R}^{N \times N}$, so $\mathbf{K}^{(1)} = \mathbf{K}_{CMDS} = -\frac{1}{2}(\mathbf{I}_N - \mathbf{1}_N \mathbf{1}_N^\top) \mathbf{D} (\mathbf{I}_N - \mathbf{1}_N \mathbf{1}_N^\top)$, where the ij entry of \mathbf{D} is given by $d_{ij} = \|\mathbf{y}_i - \mathbf{y}_j\|_2^2$ and $\|\cdot\|_2$ stands for Euclidean norm. A kernel for LLE can be approximated from a quadratic form in terms of the matrix \mathbf{W} holding linear coefficients that sum to 1 and optimally reconstruct observed data. Define a matrix $\mathbf{M} \in \mathbb{R}^{N \times N}$ as $\mathbf{M} = (\mathbf{I}_N - \mathbf{W})(\mathbf{I}_N - \mathbf{W}^\top)$ and λ_{max} as the largest eigenvalue of \mathbf{M} . Kernel matrix for LLE is in the form $\mathbf{K}^{(2)} = \mathbf{K}_{LLE} = \lambda_{max} \mathbf{I}_N - \mathbf{M}$. Since kernel PCA is a maximization problem of the covariance of the high-dimensional data represented by a kernel [11], LE can be expressed as the pseudo-inverse of the graph Laplacian \mathbf{L} , so $\mathbf{K}^{(3)} = \mathbf{K}_{LE} = \mathbf{L}^\dagger$, where $\mathbf{L} = \mathbf{D} - \mathbf{S}$, \mathbf{S} is a similarity matrix and $\mathbf{D} = \text{Diag}(\mathbf{S} \mathbf{1}_N)$ is the degree matrix. All previously mentioned kernels are widely described in [10]. The similarity matrix \mathbf{S} is formed in such a way that the relative bandwidth parameter is estimated keeping the entropy over neighbor distribution as roughly $\log K$ where K is the given number of neighbors as explained in [15]. The number of neighbors is established as $K = 30$.

Databases: Experiments are carried out over three conventional data sets. The first data set is an artificial spherical shell ($N = 1500$ data points and $D = 3$). The second data set is a toy set here called **Swiss roll** ($N = 3000$ data points and $D = 3$). Figure 3 depicts examples of the considered data sets.

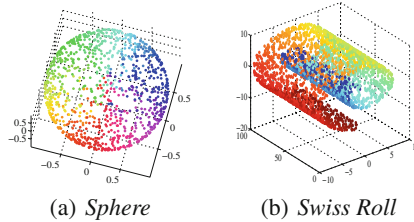


Fig. 3. The considered datasets.

Performance Measure: To quantify the performance of studied methods, the scaled version of the average agreement rate $R_{NX}(K)$ introduced in [12] is used, which is ranged within the interval $[0, 1]$. Since $R_{NX}(K)$ is calculated at each perplexity value from 2 to $N - 1$, a numerical indicator of the overall performance can be obtained by calculating its area under the curve (AUC). The AUC assesses the dimension reduction quality at all scales, with the most appropriate weights.

5 Results and Discussion

Aimed at depicting the methodology usability, Fig. 4 represents a comparison of the embedded spaces obtained with KLE and KPCA methods. It illustrates a two-dimensional representation of the original data which is obtained through the application of the mixture of three different kernels approximations of DR methods.

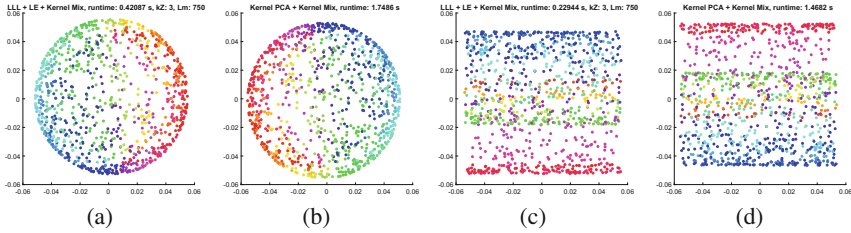


Fig. 4. Results for KLE (figures a and b) and KPCA (figures c and d) obtained under a weighted mix of the three kernel approaches and applying a computational cost reduction.

In addition, it can be observed the difference in the computation time of the spectral decomposition of the proposed methodology with respect to KPCA for one iteration with a number of $L = 750$ landmarks and a number $kZ = 3$. In order to demonstrate the computational cost reduction, values of $L = 750$ and $kZ = 5$ were selected but these values were obtained after running tests for different values of L between $[1, N]$, in steps of 50 points, and $kZ = [1, 50]$, in steps of 5 points. These results allow to measure the quality of the proposed approach in both, in a visual way where it is possible to highlight some patterns and trends that is not observable in the raw data and in a numeric way with the area under the curve given by the function $R_{NX}(k)$. This curves are created for each method which is applied to the databases mentioned previously.

Table 1 presents this reduction as a percentage measure. It shows an average time of computing the eigen-decomposition of KPCA and KLE in seconds. KLE presents different landmarks values L . The number of L and kZ are selected taking into account criteria of visual representation in low dimension, quality curve and computation time of the eigen-decomposition.

The results of AUC of classical approaches and the use of kernel approximations in KPCA and KLE are illustrated in Fig. 5. The information presented in the results provides a good indication of the proposed approach quality due to the approximation of KPCA. Thus, the methodology proposed is comparable in results and data representation. As can be seen, the quality curves present results comparable with KPCA and the conventionally applied method. In contrast, our methodology presents a lower computational cost in relation to KPCA.

Table 1. Percentage of computational cost reduction (CRP) for each database under $kZ = 5$ and with variation in the number of submatrices (L).

L	Sphere		CRP	Swiss Roll		CRP	MNIST		CRP
	KPCA	KLE		KPCA	KLE		KPCA	KLE	
100	29.4	0.21	99.29	28.84	0.21	99.27	33.4	0.45	98.65
400	29.4	0.31	98.95	28.84	0.21	99.27	33.4	0.67	97.99
700	29.4	0.28	99.05	28.84	0.29	98.99	33.4	0.85	97.46
1000	29.4	0.37	98.74	28.84	0.42	98.54	33.4	1.47	95.60
1300	29.4	0.51	98.27	28.84	0.53	98.16	33.4	1.97	94.10
1600	29.4	0.59	97.99	28.84	0.72	97.50	33.4	1.61	95.18
1900	29.4	0.74	97.48	28.84	0.93	96.78	33.4	1.85	94.46
2200	29.4	0.99	96.63	28.84	1.22	95.77	33.4	2.29	93.14
2500	29.4	1.19	95.95	28.84	1.41	95.11	33.4	2.71	91.89
2800	29.4	1.35	95.41	28.84	1.6	94.45	33.4	3.12	90.66
	Mean		97.78	Mean		97.39	Mean		94.91

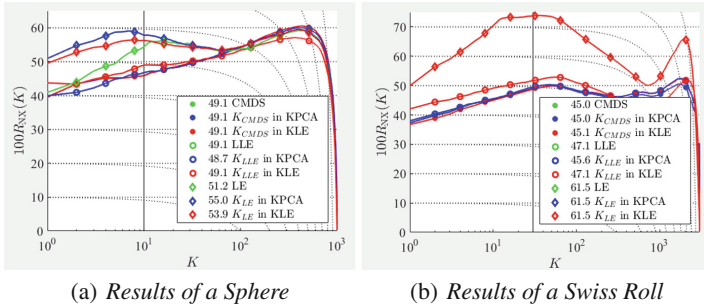


Fig. 5. Quality curves obtained for each conventionally applied method and for its kernel approaches applied to KLE and KPCA.

6 Conclusions and Future Work

In this work, a multiple kernel learning approach for dimensionality reduction tasks is presented. The core of this approach is the generalized kernel that is calculated by means of a linear combination of kernel matrices representing spectral dimensionality reduction methods, where the coefficients are obtained from a variable ranking based on a variance criterion. Proposed approach improves both data visualization and preservation by exploiting the representation ability of every single technique. Given the versatility of spectral DR methods to be represented by a kernel approximation, our methodology allows us to generate low dimensional representations from the inclusion of these Kernel approximations or the weighted mix of them. Although this methodology is less expensive to compute than KPCA, the spectral decomposition for the calculation of eigen-

vectors and eigenvalues still requires considerable processing time. To solve this, the original data set is approximated by estimating a number of submatrixes, represent a neighborhood locally, and a projection matrix that allows the data to be reconstructed. In this research, we estimated a sufficient number of submatrixes which are processed and this dramatically reduces the computational cost. The results show that when the number of submatrixes representing a certain number of neighborhoods is sufficient, the representation obtained is very approximate and of such good quality as that obtained with all the original space, but at a lower computational cost.

As future work, new multiple kernel learning approaches will be explored by combining kernel representations arising from other dimensionality reduction methods, aimed at reaching a good trade-off between preservation of data structure and intelligible data visualization considering factors for the estimation of the number of sub-matrixes and neighborhoods that allow the original space to be reduced with a non-significant loss of information.

Acknowledgment. The authors acknowledge to the research project “Desarrollo de una metodología de visualización interactiva y eficaz de información en Big Data” supported by Agreement No. 180 November 1st, 2016 by VIPRI from Universidad de Nariño. Authors thank the valuable support given by the SDAS Research Group (www.sdas-group.com).

References

1. Borg, I., Groenen, P.J.: *Modern Multidimensional Scaling: Theory and Applications*. Springer, New York (2005). <https://doi.org/10.1007/0-387-28981-X>
2. Salazar-Castro, J.A., et al.: Dimensionality reduction for interactive data visualization via a geo-desic approach. In: 2016 IEEE Latin American Conference on Computational Intelligence (LA-CCI), pp. 1–6. IEEE (2016)
3. Salazar-Castro, J.A., et al.: A novel color-based data visualization approach using a circular interaction model and dimensionality reduction. In: Huang, T., Lv, J., Sun, C., Tuzikov, A.V. (eds.) *ISNN 2018*. LNCS, vol. 10878, pp. 557–567. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-92537-0_64
4. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.* **15**(6), 1373–1396 (2003)
5. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. *Science* **290**(5500), 2323–2326 (2000)
6. Aggarwal, C.C.: Outlier analysis. In: Aggarwal, C.C. (ed.) *Data Mining*, pp. 237–263. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-319-14142-8_8
7. Langone, R., Alzate, C., Suykens, J.A.: Kernel spectral clustering with memory effect. *Phys. A: Stat. Mech. Appl.* **392**(10), 2588–2606 (2013)
8. Salazar-Castro, J., Rosas-Narváez, Y., Pantoja, A., Alvarado-Pérez, J.C., Peluffo-Ordóñez, D.H.: Interactive interface for efficient data visualization via a geometric approach. In: 2015 20th Symposium on Signal Processing, Images and Computer Vision (STSIVA), pp. 1–6. IEEE (2015)
9. Rosero-Montalvo, P., et al.: Interactive data visualization using dimensionality reduction and similarity-based representations. In: Beltrán-Castañón, C., Nyström, I., Famili, F. (eds.) *CIARP 2016*. LNCS, vol. 10125, pp. 334–342. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-52277-7_41

10. Ham, J., Lee, D.D., Mika, S., Schölkopf, B.: A Kernel view of the dimensionality reduction of manifolds. In: Proceedings of the Twenty-First International Conference on Machine Learning, p. 47. ACM (2004)
11. Peluffo-Ordóñez, D.H., Lee, J.A., Verleysen, M.: Generalized Kernel framework for unsupervised spectral methods of dimensionality reduction. In: 2014 IEEE Symposium on Computational Intelligence and Data Mining (CIDM), pp. 171–177. IEEE (2014)
12. Lee, J.A., Renard, E., Bernard, G., Dupont, P., Verleysen, M.: Type 1 and 2 mixtures of Kullback-Leibler divergences as cost functions in dimensionality reduction based on similarity preservation. *Neurocomputing* **112**, 92–108 (2013)
13. Saul, L.K., Weinberger, K.Q., Ham, J.H., Sha, F., Lee, D.D.: Spectral methods for dimensionality reduction. In: *Semisupervised Learning*, pp. 293–308 (2006)
14. Vladymyrov, M., Carreira-Perpiñán, M.Á.: Locally linear landmarks for large-scale manifold learning. In: Blockeel, H., Kersting, K., Nijssen, S., Železný, F. (eds.) *ECML PKDD 2013. LNCS (LNAI)*, vol. 8190, pp. 256–271. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40994-3_17
15. Cook, J., Sutskever, I., Mnih, A., Hinton, G.: Visualizing similarity data with a mixture of maps. In: *Artificial Intelligence and Statistics*, pp. 67–74 (2007)



Optimally Selected Minimal Learning Machine

Átilla N. Maia¹(✉), Madson L. D. Dias²(✉), João P. P. Gomes²,
and Ajalmar R. da Rocha Neto¹

¹ Graduate Program in Computer Science, Federal Institute of Ceará (IFCE),
Fortaleza, Ceará, Brazil

atilla.negreiros@ppgcc.ifce.edu.br, ajalmar@ifce.edu.br

² Department of Computer Science, Federal University of Ceará (UFC),
Fortaleza, Ceará, Brazil

{madson.dias, jpaulo}@lia.ufc.br

Abstract. This paper introduces a new approach to select reference points (RPs) to minimal learning machine (MLM) for classification tasks. A critical issue related to the training process in MLM is the selection of RPs, from which the distances are taken. In its original formulation, the MLM selects the RPs randomly from the data. We propose a new method called optimally selected minimal learning machine (OS-MLM) to select the RPs. Our proposal relies on the multiresponse sparse regression (MRSR) ranking method, which is used to sort the patterns in terms of relevance. After doing so, the leave-one-out (LOO) criterion is also used in order to select an appropriate number of reference points. Based on the simulations we carried out, one can see our proposal achieved a lower number of reference points with an equivalent, or even superior, accuracy with respect to the original MLM and its variants.

Keywords: Minimal Learning Machine · Reference points
Multiresponse sparse regression · Sparse models

1 Introduction

Minimal Learning Machine (MLM, [20]) is a recent supervised learning algorithm that can be used to handle classification and regression problems. This method has been employed to a diverse range of problems, such as fault detection in motors [4], ranking of documents [2] and location of mobile robots [13].

The basic idea behind MLM is the assumption about the existence of a mapping between the geometric configurations of points in the input space and the geometric configurations of respective points in the output space. This mapping is represented by two distance matrices (input and output), computed for all points in the training data set and a subset of it, named reference points. The learning of MLM is accomplished by determination of a regression linear model

Supported by Federal Institute of Ceará and Federal University of Ceará.

© Springer Nature Switzerland AG 2018

H. Yin et al. (Eds.): IDEAL 2018, LNCS 11314, pp. 670–678, 2018.

https://doi.org/10.1007/978-3-030-03493-1_70

between two distance matrices. Thus, given a point in the input space, the MLM can compute the location of this point in the output space through the learned regression model [5]. In other words, the MLM training algorithm consists in obtaining a multi-response linear system solution.

Several approaches have been applied to achieve sparse solutions in machine learning methods, such as greedy algorithms in orthogonal matching pursuit (OMP, [17]) for extreme learning machines (OMP-ELM, [1]) and multiresponse sparse regression (MRSR, [19]) for the least squares support vector machines (LSSVMs, [21]). Moreover, bayesian methods as sparse bayesian learning (SBL, [12]) have also been applied to single-hidden layer feedforward neural networks [11] and relevance vector machines (RVM, [22]).

In order to achieve sparsity for MLM, we propose a new method called optimally selected minimal learning machines (OS-MLM), which relies on a ranking method named multiresponse sparse regression (MRSR), used to sort the patterns in terms of relevance, and, after doing so, on the leave-one-out (LOO) criterion, applied to select an appropriate number of reference points. Our proposal is somewhat similar to a method called optimally pruned extreme learning machines (OP-ELM, [16]), which is used to prune neurons in hidden layers of extreme learning machines.

The remaining part of the paper is organized as follows. In Sect. 2 we describe the regular MLM. Section 3 briefly presents some strategies to select RPs. In Sect. 4, we present a multi-response sparse regression method, while our proposal is detailed in Sect. 5. In Section 6, we present the simulations carried out. Finally, the conclusions are outlined in Sect. 7.

2 Minimal Learning Machine

The Minimal Learning Machine is a supervised method whose training step consists of fitting a multiresponse linear regression model between distances computed from the input and output spaces. Output prediction for new inputs is achieved by estimating distances in output spaces using the underlying linear model followed by a optimization procedure in the space of possible outputs.

Let a training dataset $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$, a set $\mathcal{R} = \{(\mathbf{r}_m, \mathbf{t}_m)\}_{m=1}^M \subseteq \mathcal{D}$ of reference points, such that $\mathbf{x}_n, \mathbf{r}_m \in \mathbb{R}^D$ and $\mathbf{y}_n, \mathbf{t}_m \in \mathbb{R}^S$. Furthermore, let $\mathbf{D}, \mathbf{\Delta} \in \mathbb{R}^{N \times M}$ are distance matrices such that their m -th columns are respectively $[\|\mathbf{x}_1 - \mathbf{r}_m\|_2, \dots, \|\mathbf{x}_N - \mathbf{r}_m\|_2]^T$ and $[\|\mathbf{y}_1 - \mathbf{t}_m\|_2, \dots, \|\mathbf{y}_N - \mathbf{t}_m\|_2]^T$. The key idea behind MLM is the assumption of a linear mapping between \mathbf{D} and $\mathbf{\Delta}$, giving rise to the following regression model:

$$\mathbf{\Delta} = \mathbf{D}\mathbf{B} + \mathbf{E} \quad (1)$$

where $\mathbf{B} \in \mathbb{R}^{M \times M}$ is the matrix of regression coefficients and $\mathbf{E} \in \mathbb{R}^{N \times K}$ is a matrix of residuals. Under the normal conditions where the number of selected reference points is smaller than the number of training points (i.e., $M < N$), the matrix \mathbf{B} can be approximated by the usual least squares estimate

$$\hat{\mathbf{B}} = (\mathbf{D}^T \mathbf{D})^{-1} \mathbf{D}^T \mathbf{\Delta}. \quad (2)$$

Given a new input point \mathbf{x} , the vector $\hat{\boldsymbol{\delta}} = [\hat{\delta}_1, \dots, \hat{\delta}_M]$ of the distances between the output \mathbf{y} of point \mathbf{x} and the M output reference points, is given by

$$\hat{\boldsymbol{\delta}} = [\|\mathbf{x} - \mathbf{r}_1\|_2, \dots, \|\mathbf{x} - \mathbf{r}_M\|_2] \hat{\mathbf{B}}. \quad (3)$$

Therefore, an estimate $\hat{\mathbf{y}}$ of \mathbf{y} can be obtained by the following minimization problem:

$$\hat{\mathbf{y}} = \arg \min_{\mathbf{y}} \left\{ \sum_{m=1}^M \left((\mathbf{y} - \mathbf{r}_m)^T (\mathbf{y} - \mathbf{r}_m) - \hat{\delta}_m^2 \right)^2 \right\}, \quad (4)$$

which can be approached via any gradient-based optimization algorithm. In the original paper, the regular MLM applies the Levenberg-Marquardt method [14].

For the classification case, where outputs \mathbf{y}_n are represented using the 1-of- S encoding scheme¹. It was showed in [15] that under the assumption that the classes are balanced, the optimal solution to Eq. (4) is given by $\hat{\mathbf{y}} = \mathbf{t}_{m^*}$, where $m^* = \arg \min_m \hat{\delta}_m$. It means that output predictions for new incoming data can be carried out by simply selecting the output of the nearest reference point in the output space, estimated using the linear model $\hat{\mathbf{B}}$. This method was named Nearest Neighbor MLM (NN-MLM).

3 RPs Selection Methods

In the original MLM proposal, the choice of reference points is random, leaving just the number of points by user's taste [5]. In this paper, the MLM with RP random selection will be called random MLM (RN-MLM). A particular case of this approach is the use of all points in the dataset as RPs. In this case, this approach will be called full MLM (FL-MLM). Another approach to select reference points is the Fuzzy C -means MLM (FCM-MLM) [8], which uses the fuzzy C -means algorithm as main method. In this case, the K parameter represents the maximum number of RPs (i.e. $|\mathcal{R}| \leq K$). This is possible by removing RPs in heterogeneous regions.

4 Multiresponse Sparse Regression

The MRSR is an extension of least angle regression (LARS, [7]) that enables the most accurate prediction averaged over all the target variables rather than only one as LARS does. LARS is a less greedy version of traditional forward selection methods. Forward stagewise linear regression, an iterative technique, builds up the regression function in successive small steps k in direction of the target whose correlation with the current residuals is maximal, so that $0 < k < |c_{\hat{j}}|$; where $c_{\hat{j}}$ is the current correlation between the targets and the regressor \hat{j} with

¹ A S -level qualitative variable is represented by a vector of S binary variables or bits, only one of which is *on* at a time. Thus, the j -th component of an output vector \mathbf{y} is set to 1 if it belongs to class j and 0 otherwise.

higher absolute correlation. A big step size $k = |c_j^t|$ leads to the classic forward selection, which can be overly greedy. In order to improve the process, LARS determines analytically the optimal step size in direction of target such that another regressor has as much correlation as possible with the current predictor. The MRSR technique is shown below.

Let the approximation of the linear system presented in Eq. (1) as $\mathbf{D}\mathbf{B}^t = \mathbf{\Delta}^t$, where, in terms of the MRSR, \mathbf{D} is the regressor matrix, \mathbf{B}^t is the solution for the linear system and, of course, $\mathbf{\Delta}^t$ is the t -th approximation of $\mathbf{\Delta}$. The matrix \mathbf{B}^t is updated by

$$\mathbf{B}^{t+1} = (1 - \gamma^t)\mathbf{B}^t + \gamma^t\bar{\mathbf{B}}^{t+1}, \quad (5)$$

where $\gamma^t = \min\{\gamma : \gamma \geq 0 \text{ and } \gamma \in \Gamma_j \text{ for some } j \notin \mathcal{A}\}$ is the step size at t -th iteration, so that Γ_j is the set

$$\Gamma_j = \left\{ \frac{c_{\max}^t + \mathbf{s}^T(\mathbf{\Delta} - \mathbf{\Delta}^t)^T \mathbf{d}_j}{c_{\max}^t + \mathbf{s}^T(\bar{\mathbf{\Delta}}^{t+1} - \mathbf{\Delta}^t)^T \mathbf{d}_j} \right\}, \quad (6)$$

and $c_{\max}^t = \max_j \{c_j^t = \|(\mathbf{\Delta} - \mathbf{\Delta}^t)^T \mathbf{d}_j\|_1\}$ is the maximum cumulative correlations from the set of regressors that satisfy the maximum $\mathcal{A} = \{j : c_j^t = c_{\max}^t\}$, \mathbf{s} is a vector of ± 1 , and, finally, the matrix $\bar{\mathbf{B}}^{t+1}$ is computed by the usual least squares estimate

$$\bar{\mathbf{B}}^{t+1} = \mathbf{D}_{\mathcal{A}}(\mathbf{D}_{\mathcal{A}}^T \mathbf{D}_{\mathcal{A}})^{-1} \mathbf{D}_{\mathcal{A}}^T \mathbf{\Delta}, \quad (7)$$

where $\mathbf{D}_{\mathcal{A}} = [\dots \mathbf{d}_j \dots]_{j \in \mathcal{A}}$ is an $N \times |\mathcal{A}|$ matrix of collected regressors that belong to \mathcal{A} . Note that \mathbf{d}_j is the i -th column of \mathbf{D} .

The idea behind the MRSR is to have, at beginning, the matrix \mathbf{B}^t with zero values and then to add a new nonzero row at each new step. Therefore, the weight matrix has k nonzero rows at k -th step of the MRSR. As each row is added one after another, the sequence created represents the rank of rows. The row to be added is chosen by computing the cumulative correlation between the regressor (vector $\mathbf{d}_j : j \notin \mathcal{A}$) and the current residuals (the difference $\mathbf{\Delta} - \mathbf{\Delta}^t$). As the linear system output is a vector, the MRSR coincides with the LARS algorithm [7]. In fact, as stated, MRSR is an extension of LARS. More details about MRSR can be found in [19].

5 Our Proposal: OS-MLM

In a nutshell, the optimally selected minimal learning machine (OS-MLM)² relies on three main steps. The first step is to build the matrix of distances \mathbf{D} from the data using all patterns as RPs. The second one is to rank columns of \mathbf{D} (which also means to rank patterns) by MRSR in order to obtain the most relevant columns (i.e., reference points). After that, the last step is the leave-one-out optimization so that the best set of columns is achieved. The main idea is to leave less important columns (patterns) out of the solution by eliminating the columns

² The use of term ‘‘optimally’’ is based on the our inspiration, the optimally pruned extreme learning machines method [16].

with low rank from matrix \mathbf{D} , then to solve the problem by the pseudo-inverse. We highlight the \mathbf{D} 's rows, also associated with a certain reference point, are not removed because its elimination would lead to a loss of labeling information and performance [23]. Despite the MRSR ranks the rows, we can use the row ranking as column ranking since the n -th row equals the n -th column (i.e., $\mathbf{D}^T = \mathbf{D}$). After ranking the patterns, the decision for the best set of reference points in the model is taken by LOO validation method. As such, LOO is computed by the prediction sum of squares (PRESS, [3]), since the LOO in its default version is very time consuming. At last, the matrix \mathbf{B} is calculated in order to obtain the final model. We highlight that our proposal is inspired by recent methodologies, called OP-ELM and OP-LSSVM, proposed to prune hidden layer neurons in extreme learning machines (ELM, [9]) and support vectors in least squares support vector machines (LSSVM, [18]), respectively.

5.1 OS-MLM Algorithm

We present the proposed algorithm for OS-MLM below.

Algorithm 1. OS-MLM

Input: \mathcal{D} : training dataset

Output: Regression model ($\hat{\mathbf{B}}$), set of RPs (\mathcal{R})

- 1: Build input distance matrix \mathbf{D} from the data using all samples as RPs
- 2: Rank the columns of \mathbf{D} (reference points) by MRSR

$$\mathbf{r} \leftarrow \text{RANKING-BY-MRSR}(\mathbf{D})$$

- 3: Using the rank \mathbf{r} , select the best RP set by PRESS LOO

$$\mathcal{R} \leftarrow \text{SELECTING-BY-LOO}(\mathbf{D}, \mathbf{r})$$

- 4: Compute the distance matrices \mathbf{D} and $\mathbf{\Delta}$ for data points and the new \mathcal{R} set.
 - 5: Calculate the regression model $\hat{\mathbf{B}}$ through Eq. (2)
 - 6: **return** $\hat{\mathbf{B}}$, \mathcal{R}
-

6 Simulations and Discussion

For a qualitative analysis, we have also applied OS-MLM, RN-MLM, FL-MLM and FCM-MLM to solve an artificial problem. The problem named simple checkerboard (SCB) consists of 400 points taken from a 2×2 checkerboard (Fig. 1).

Based on Fig. 1, one can see that the number of RPs for OS-MLM is lower than the number of RPs for RN-MLM and very close to the FCM-MLM. Moreover, the decision boundary generated from the FCM-MLM is more smoothed than the other models. Additionally, one can note, in the qualitative analysis, the OS-MLM method avoids RPs on overlapping regions. Thus, the decision boundaries are not overfitted.

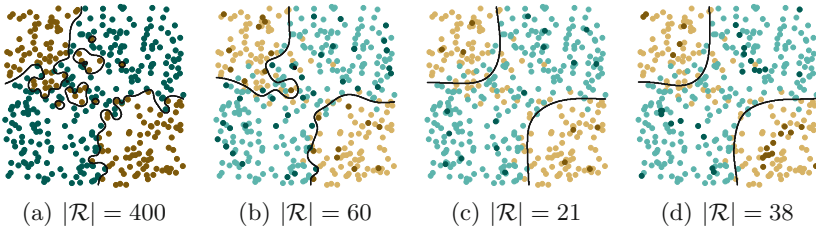


Fig. 1. Decision surface and number of RPs for (a) FL-MLM, (b) RN-MLM, (c) FCM-MLM, and (d) OS-MLM, when applied to SCB problem.

Tests with real-world benchmark datasets were also evaluated in this work. We used some UCI datasets [10]: Haberman’s Survival (HAB) with 3 features and 306 patterns; Vertebral Column Pathologies (VCP) with 4 features and 310 patterns; Liver Disorders (LID) with 6 features and 345 patterns; Ionosphere (ION) with 34 features and 351 patterns; Breast Cancer Winconsin (BCW) with 9 features and 683 patterns; Pima Indians Diabetes (PID) with 8 features and 768 patterns; Car Evaluation (CAR) with 6 features and 1728 patterns; and Human Immunodeficiency Virus protease cleavage (HIV) with 8 features and 3272 patterns. In addition, two well-known artificial data sets were also used in our simulations, Ripley (RIP) with 2 features and 1250 patterns and Two Moon (TMN) with 2 features and 1001 patterns.

In our simulations, 80% of the data was randomly selected for training purposes and the remaining 20% was used for assessing the classifiers’ generalization performance. We carried out 30 executions on each dataset.

The adjustment of the parameter K for the FCM-MLM and the RN-MLM model was performed using grid search with 10-fold cross-validation. The RPs were selected in the range of 5–100% (with a step of 5%) of the available training samples. The accuracy was used to choose the best value of K .

In order to verify the possible equivalence between the classifier accuracies, we perform a two-sample Friedman test [6] with a significance level of 1%. In the hypothesis test, the null and the alternative hypothesis means that the accuracy is equivalent or not, respectively.

In Table 1, we report performance metrics for the aforementioned 30 independent runs. We show the accuracy (ACC), the average of the percentage of reduction (RED) in the RPs, compared to the FL-MLM model, and the results of the Friedman statistic test, in which ✓ is a relation of equivalence between the methods and ✗ indicates a considerable difference among them.

As expected, the performance of the OS-MLM was equivalent to or higher than that achieved by the RN-MLM, FL-MLM, and the FCM-MLM for each evaluated dataset. Particularly, the OS-MLM achieved the best results (in terms of accuracy) in 4 of the datasets. In terms of RPs reduction, our proposal achieved best results in 9 of the datasets. In other words, our proposal achieves accuracies

Table 1. Performance metrics for OS-MLM, RN-MLM, FL-MLM, and FCM-MLM; Accuracy (ACC) and reduction percentage in comparison with the training set (RED); and results of statistical tests.

<i>dataset</i>	<i>metric</i>	OS-MLM	RN-MLM	FL-MLM	FCM-MLM
HAB	ACC	73.44 ± 4.09	71.97 ± 4.27 ✓	68.09 ± 4.98 × ×	71.36 ± 6.03 ✓ ✓ ×
	RED	94.46 ± 3.32	80.20 ± 14.22		59.44 ± 16.50
VCP	ACC	87.20 ± 3.57	87.80 ± 3.92 ✓	87.37 ± 3.98 ✓ ✓	84.78 ± 4.48 ✓ ✓ ✓
	RED	79.87 ± 13.66	49.17 ± 27.95		80.60 ± 5.99
LID	ACC	71.01 ± 5.16	68.84 ± 5.59 ✓	68.16 ± 6.76 × ✓	68.70 ± 5.44 ✓ ✓ ✓
	RED	86.32 ± 6.12	59.82 ± 26.51		63.61 ± 20.79
ION	ACC	92.33 ± 3.04	93.10 ± 2.73 ✓	94.14 ± 2.74 × ×	93.33 ± 2.87 × ✓ ✓
	RED	76.57 ± 18.93	42.18 ± 21.05		57.77 ± 12.40
BCW	ACC	91.33 ± 12.75	96.98 ± 1.40 ×	96.96 ± 1.27 × ✓	97.00 ± 1.31 ✓ ✓ ✓
	RED	74.25 ± 27.37	62.61 ± 22.71		86.91 ± 4.65
PID	ACC	75.15 ± 2.96	74.59 ± 2.58 ✓	73.16 ± 2.38 × ×	73.44 ± 2.86 × × ✓
	RED	95.04 ± 0.87	75.92 ± 16.10		84.61 ± 8.11
CAR	ACC	92.89 ± 1.95	97.00 ± 0.76 ×	98.02 ± 0.69 × ×	96.12 ± 1.29 × × ×
	RED	90.69 ± 3.48	17.64 ± 10.05		27.14 ± 6.96
TMN	ACC	99.52 ± 0.48	99.82 ± 0.28 ×	99.87 ± 0.22 × ✓	99.87 ± 0.22 × × ×
	RED	93.10 ± 6.79	61.92 ± 20.72		63.63 ± 23.71
RIP	ACC	90.49 ± 1.83	89.75 ± 1.77 ✓	88.32 ± 1.61 × ×	89.81 ± 1.88 ✓ ✓ ×
	RED	96.02 ± 0.80	76.64 ± 18.83		87.30 ± 11.01
HIV	ACC	86.62 ± 1.26	86.50 ± 1.30 ✓	85.99 ± 1.14 × ×	86.68 ± 1.30 ✓ × ×
	RED	90.92 ± 2.14	75.32 ± 23.16		90.68 ± 17.02

that are comparable to others variants of MLM, but with a lower number of RPs. It is also important to notice that for most datasets the OS-MLM achieved a low standard deviation.

7 Conclusions

Motivated by the poor generalization capability of the random selection, this paper presents an alternative algorithm to select reference points of the MLM for classification tasks based on multiresponse sparse regression and PRESS statistics. Four strategies of MLM RPs selection are evaluated. The results of the simulations indicate the OS-MLM works very well, providing a competitive classifier while maintaining its simplicity. We also achieved sparseness in our classifier.

References

1. Alcin, O., Sengur, A., Qian, J., Ince, M.: OMP-ELM: orthogonal matching pursuit-based extreme learning machine for regression. *J. Intell. Syst.* **24**(1), 135–143 (2015)
2. Alencar, A.S.C., et al.: MLM-rank: a ranking algorithm based on the minimal learning machine. In: 2015 Brazilian Conference on Intelligent Systems, BRACIS 2015, Natal, Brazil, 4–7 November 2015, pp. 305–309. IEEE (2015)
3. Allen, D.M.: The relationship between variable selection and data agumentation and a method for prediction. *Technometrics* **16**(1), 125–127 (1974)
4. Coelho, D.N., Barreto, G.D.A., Medeiros, C.M.S., Santos, J.D.A.: Performance comparison of classifiers in the detection of short circuit incipient fault in a three-phase induction motor. In: 2014 IEEE Symposium on Computational Intelligence for Engineering Solutions, CIES 2014, Orlando, FL, USA, 9–12 December 2014, pp. 42–48. IEEE (2014)
5. de Sousa, L.S., Dias, M.L.D., Rocha Neto, A.R.: Máquinas de vetores-suporte de mínimos quadrados esparsas via recozimento simulado. In: Simpósio Brasileiro de Automação Inteligente (SBAI). SBA, Rio Grande do Norte, Brasil, October 2015
6. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **7**, 1–30 (2006)
7. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least angle regression. *Ann. Stat.* **32**(2), 407–499 (2004)
8. Florêncio, J.A.V., Dias, M.L.D., da Rocha Neto, A.R., de Souza Júnior, A.H.: A fuzzy C -means-based approach for selecting reference points in minimal learning machines. In: Barreto, G.A., Coelho, R. (eds.) NAFIPS 2018. CCIS, vol. 831, pp. 398–407. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-95312-0_34
9. Huang, G., Zhu, Q., Siew, C.K.: Extreme learning machine: theory and applications. *Neurocomputing* **70**(1–3), 489–501 (2006)
10. Lichman, M.: UCI machine learning repository (2013)
11. Luo, J., Vong, C., Wong, P.: Sparse Bayesian extreme learning machine for multi-classification. *IEEE Trans. Neural Netw. Learn. Syst.* **25**(4), 836–843 (2014)
12. MacKay, D.J.: Bayesian interpolation. *Neural Comput.* **4**(3), 415–447 (1992)
13. Marinho, L.B., Almeida, J.S., Souza, J.W.M., de Albuquerque, V.H.C., Filho, P.P.R.: A novel mobile robot localization approach based on topological maps using classification with reject option in omnidirectional images. *Expert Syst. Appl.* **72**, 1–17 (2017)
14. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. *J. Soc. Ind. Appl. Math.* **11**(2), 431–441 (1963)
15. Mesquita, D.P.P., Gomes, J.P.P., Junior, A.H.S.: Ensemble of efficient minimal learning machines for classification and regression. *Neural Process. Lett.* **46**, 1–16 (2017)
16. Miche, Y., Sorjamaa, A., Bas, P., Simula, O., Jutten, C., Lendasse, A.: OP-ELM: optimally pruned extreme learning machine. *IEEE Trans. Neural Netw.* **21**(1), 158–162 (2010)
17. Pati, Y.C., Rezaifar, R., Krishnaprasad, P.S.: Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In: Proceedings of 27th Asilomar Conference on Signals, Systems and Computers, vol. 1, pp. 40–44, November 1993
18. da Silva Vieira, D.C., da Rocha Neto, A.R., Rodrigues, A.W.D.O.: Sparse least squares support vector regression via multiresponse sparse regression. In: 2016 International Joint Conference on Neural Networks, IJCNN 2016, Vancouver, BC, Canada, 24–29 July 2016, pp. 3218–3225. IEEE (2016)

19. Similä, T., Tikka, J.: Multiresponse sparse regression with application to multidimensional scaling. In: Duch, W., Kacprzyk, J., Oja, E., Zadrozny, S. (eds.) ICANN 2005. LNCS, vol. 3697, pp. 97–102. Springer, Heidelberg (2005). https://doi.org/10.1007/11550907_16
20. de Souza Junior, A.H., Corona, F., Barreto, G.D.A., Miché, Y., Lendasse, A.: Minimal learning machine: a novel supervised distance-based approach for regression and classification. *Neurocomputing* **164**, 34–44 (2015)
21. Suykens, J.A.K., Vandewalle, J.: Least squares support vector machine classifiers. *Neural Process. Lett.* **9**(3), 293–300 (1999)
22. Tipping, M.E.: Sparse Bayesian learning and the relevance vector machine. *J. Mach. Learn. Res.* **1**, 211–244 (2001)
23. Valyon, J., Horvath, G.: A sparse least squares support vector machine classifier. In: Proceedings of IEEE International Joint Conference on Neural Networks, vol. 1, pp. 543–548 (2004)



Neural Collaborative Filtering: Hybrid Recommendation Algorithm with Content Information and Implicit Feedback

Li Ji, Guangyan Lin, and Huobin Tan^(✉)

School of Software, Beihang University, Beijing, China
{keeley, thbin}@buaa.edu.cn

Abstract. Collaborative filtering methods are widely used for recommender systems. However the performance degrades significantly because of highly sparse rating information in practical situations. Recently many hybrid models incorporating content information have been proposed to alleviate data sparsity. This paper first proposes the collaborative variational ranking model (CVRank) that combines variational autoencoder with pairwise ranking based collaborative filtering. The model learns latent factors of users and items from both rating and content information, and makes recommendation by calculating dot products between users and items. CVRank is suitable for other deep learning models, and can be easily extended to other multimedia other than text. Then this paper sums up similar models as neural collaborative filtering and presents a generic optimization criterion for them. Experiments show that CVRank achieves robust performance under different sparsity level, and neural collaborative filtering methods can gain greater recommendation accuracy improvement compared with pure collaborative filtering when the training data is sparser.

Keywords: Collaborative filtering · Variational autoencoder
Recommender system

1 Introduction

With the explosive growth of web information, recommender systems play an increasingly significant role in modern websites. Based on users' preference from their past feedback of items, recommender systems are able to help users find information relevant to their interests, such as important scientific articles related to their research area. There are three methods to produce a recommendation list: content-based filtering, collaborative filtering and hybrid methods. Content-based filtering methods utilize item description and user profile to recommend items that are similar to those that a user liked in the past [1]. Collaborative filtering relies only on past user behavior: explicit feedback or implicit feedback. Explicit feedback includes numerical scores reflecting the extent of users' interest in items, such as star ratings for movies. Implicit feedback indirectly reflects opinion by observing user behavior, such as purchase history, browsing history and clicks [2]. Content-based filtering and collaborative filtering can be combined to provide more accurate recommendations than pure approaches, called hybrid methods.

In practical situations, implicit feedback is easier to obtain, so implicit feedback based recommendation is attracting more and more attention [3–5]. We can only observe positive examples in implicit feedback, so implicit feedback based collaborative filtering is also called one class collaborative filtering (OCCF) [5, 6]. Existing methods for OCCF can be categorized into two classes: pointwise methods and pairwise methods. Pointwise methods aim to minimize the overall difference between binary rating and prediction score of all user-item pairs. Considering unobserved interactions may mean the user dislikes the item, or the user does not see the item, pointwise methods assign observed interactions larger confidence level than unobserved pairs. For example, [1] set confidence to 1 for positive examples and 0.01 for others. Pairwise methods focus on the relative personalized ranking between items for the user. The hypothesis is that user prefers observed items than unobserved items. The foremost of pairwise ranking models is Bayesian Personalized Ranking (BPR) [4]. In addition, many pairwise loss functions can be used for collaborative filtering: log loss, squared loss, hinge loss and exponential loss [7–10].

Collaborative filtering has two challenges: data sparsity and cold start. In practical situations, it is typical for a user to interact with a limited number of items out of thousands of or millions of items [10]. The extremely sparse feedback may leads to overfitting and low prediction accuracy. Second, collaborative filtering cannot determine whether to recommend a new item, as it hasn't interacts with any user.

To solve these problems, many studies try to make use of auxiliary information. Some uses tags type metadata. For example, LightFM uses tags as features, jointly factorizes the user-item, item-feature, and user-feature matrices. And then latent representations of users and items are given by the sum of their tags' latent representations [11]. Some research employs text content information. Collaborative topic regression (CTR) combines the merits of traditional collaborative filtering and probabilistic topic modeling [12]. Collaborative topic ranking (CTRrank) extends pairwise ranking models with topic models by regularizing item factors through item topic proportions [10]. Considering the latent representation learned by CTR may not be very effective when the auxiliary information is very sparse, collaborative deep learning (CDL) and collaborative deep ranking (CDRrank) jointly performs deep representation learning for the content information and collaborative filtering for the feedback matrix [1, 13]. Collaborative variational autoencoder (CVAE) learns deep latent representations from content data through variational autoencoder and implicit relationships between items and users from both content and rating [14].

The contributions of this paper are as follows:

1. We propose a probabilistic generative model with neural networks, named CVRrank, which can simultaneously learn an effective latent representation for content and implicit relationships between items and users for recommendation tasks.
2. We sum up the recently proposed hybrid models like CDL, CDRrank and CVAE as neural collaborative filtering, and present the generic optimization criterion for them.
3. We evaluate the effectiveness of CVRrank on the real-world dataset.

4. We explore the recommendation accuracy improvement which neural collaborative filtering models are able to bring about compared with pure collaborative filtering under different data sparsity.

2 Preliminaries

We first formulate the recommendation problem discussed in this paper, and then review the pointwise and pairwise methods for one class collaborative filtering.

Let N , M , K be the number of users, items and factors respectively. X_c represents the content information about all items, whose j th row $X_{c,j*}$ is the bag-of-words vector of item j . R is a N -by- M rating matrix, $r_{ij} = 1$ if user i interacts with item j , $r_{ij} = 0$ if not. P_i is the set of user i 's pairwise preferences (j, k) that satisfy $r_{ij} > r_{ik}$. u_i and v_j are user i 's latent factor vector and item j 's latent factor vector respectively. $U \in R^{N \times K}$ is the latent factors of all users in matrix form. $V \in R^{M \times K}$ is the latent factors of all items in matrix form. The goal is to learn U and V from interaction matrix R and item content information X_c to produce a personalized recommendation list for each user.

Latent factor model is one of the primary areas of collaborative filtering. Matrix factorization is one of the most popular latent factor models [2]. Users and items are associated with vectors of factors, and prediction scores are calculated by the dot product between corresponding vectors.

$$\hat{r}_{ij} = u_i^T v_j \quad (1)$$

Matrix factorization can be generalized as a probabilistic model [15]. It has the following generative process.

1. For each user, draw user latent vector $u_i \sim N(0, \lambda_u^{-1} I_K)$.
2. For each item, draw item latent vector $v_j \sim N(0, \lambda_v^{-1} I_K)$.
3. For each user-item pair, draw $r_{ij} \sim N(u_i^T v_j, c_{ij}^{-1})$.

The focus of one class collaborative filtering is how to deal with unobserved ratings. In order to apply matrix factorization to OCCF, [3] assumes unobserved ratings as zeros, but assigns observed interactions larger confidence level than unobserved interactions. So the pointwise methods learn user and item latent factors by minimizing the regularized squared error loss. c_{ij} serves as the confidence parameter for r_{ij} . $c_{ij} = a$ if $r_{ij} = 1$ and $c_{ij} = b$ if $r_{ij} = 0$ where $a > b$.

$$L_{point} = \sum_{(i,j)} c_{ij} (r_{ij} - u_i^T v_j)^2 + \lambda_u \|U\|_F^2 + \lambda_v \|V\|_F^2 \quad (2)$$

Pairwise methods focus on the relative personalized ranking between items for the user. The hypothesis is that user prefers observed items than unobserved items. BPR is proposed by Steffen in 2009 [4], which is the foremost pairwise ranking model. The individual probability that a user really prefers item i to item j is defined as (3) where

$\sigma(x) = 1/(1 + \exp(-x))$. Afterwards Steffen used a similar approach log loss in [7]. It maximizes (4), which is equivalent to minimizing the negative log likelihood as (5). D_s is the set of triples (i, j, k) that satisfy $r_{ij} = 1$ and $r_{ik} = 0$. Because of the large number of training triples in D_s , [4, 7] uses uniform sampling and stochastic gradient descent to learn the model.

$$p(j > ik) := \sigma(\hat{r}_{ijk}) = \sigma(\hat{r}_{ij} - \hat{r}_{ik}) \tag{3}$$

$$\prod_{(i,j,k) \in D_s} p(j > ik) \tag{4}$$

$$L_{log} = - \sum_{(i,j,k) \in D_s} \ln \sigma(\hat{r}_{ij} - \hat{r}_{ik}) \tag{5}$$

3 Neural Collaborative Filtering

3.1 Collaborative Variational Ranking

In this section, we first introduce collaborative variational autoencoder ranking model (CVRank) for recommendation with content. Then we sum up the recently proposed hybrid models like CDL, CDRank and CVAE, and propose a generic loss function for this type of neural collaborative filtering.

Autoencoder is a type of neural network used to learn efficient data coding in an unsupervised manner [16]. It learns to encode the input into a short code and then decode that code into output that closely matches the original data. Autoencoder is a good way to extract feature representation from item content information. Denoising autoencoder (DAE) is a variation of autoencoder. It uses corrupted data X_0 as input and tries to recover the original undistorted data. Stacked denoising autoencoder (SDAE) stacks denoising autoencoders to form a deep network. It was introduced in [17], and can be generalized as a probabilistic model [18]. Variational autoencoder (VAE) [19] is one of the most widely used deep generative models. It seeks for a probabilistic latent variable model for the content. It infers the stochastic distribution of the latent variable through an inference network instead of point estimates [14].

CVRank, as shown in Fig. 1, combines variational autoencoder and pairwise ranking based collaborative filtering. There are two generative processes in our model. First the original VAE process extracts feature representation from content information about items and then integrates them into latent factor of items in pairwise ranking model. $f(x_j)$ is the feature representation of x_j compressed by neural network. Second the pairwise ranking model captures special relationship $\delta_{ijk} = r_{ij} - r_{ik}$, which delegates the preference of user i on item j and k . The difference from CVAE is that CVAE directly predicts the value r_{ij} . The generative process of CVRank is as follows:

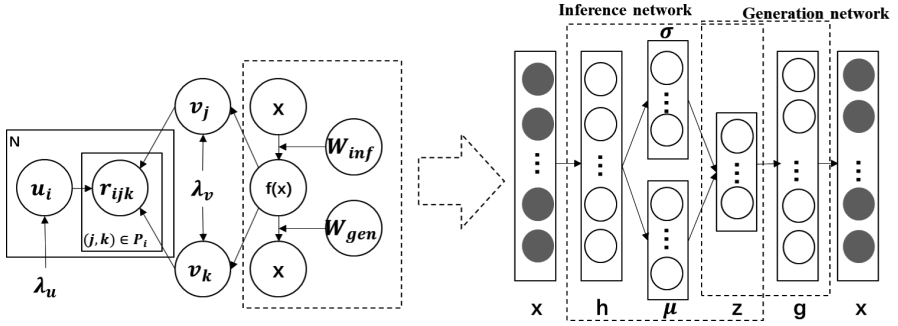


Fig. 1. The graphic model of CVRank. On the right is the inference network and generation network in CVRank.

1. For each layer of the inference network
 - a. For each column n of the weight matrix W_l , draw $W_{l,*n} \sim N(0, \lambda_w^{-1} I_{K_l})$.
 - b. Draw $b_l \sim N(0, \lambda_w^{-1} I_{K_l})$.
 - c. For each row j of h_l , draw $h_{l,j*} \sim N(\sigma(h_{l-1,j*} W_l + b_l), \lambda_s^{-1} I_{K_l})$.
2. For each item j ,
 - a. draw latent mean vector $\mu_j \sim N(\sigma(h_L W_\mu + b_\mu), \lambda_s^{-1} I_K)$, draw latent covariance vector $\log \sigma_j^2 \sim N(\sigma(h_L W_\sigma + b_\sigma), \lambda_s^{-1} I_K)$.
 - b. draw latent content vector $z_j \sim N(\mu_j, \text{diag}(\sigma_j))$.
 - c. draw a latent item offset vector $\epsilon_j \sim N(0, \lambda_v^{-1} I_K)$, set $v_j = \epsilon_j + \mu_j$.
3. For each layer of the generation network
 - a. For each column n of the weight matrix W_l , draw $W_{l,*n} \sim N(0, \lambda_w^{-1} I_{K_l})$.
 - b. Draw $b_l \sim N(0, \lambda_w^{-1} I_{K_l})$.
 - c. For each row j of g_l , draw $g_{l,j*} \sim N(\sigma(g_{l-1,j*} W_l + b_l), \lambda_s^{-1} I_{K_l})$.
4. For each item j , draw $x_j \sim N(g_{L,j*} W_{L+1} + b_{L+1}, \lambda_n^{-1} \mathbf{1})$.
5. For each user
 - a. draw user latent vector $u_i \sim N(0, \lambda_u^{-1} I_K)$.
 - b. for each pairwise preference $(j, k) \in P_i$, draw $r_{ijk} = 1 \sim \text{Bernoulli}(\sigma(u_i^T v_j - u_i^T v_k))$.

Maximum a posterior probability (MAP) can be utilized to learn model parameters. The objective thus becomes to minimize the minus of the joint likelihood of $U, V, \{h_l\}, \{g_l\}, X_c, \{W_l\}, \{b_l\}$ and R as (6). From the perspective of optimization, the first term and the second item are equivalent to an VAE minimizing the reconstruction error and the Kullback-Leibler loss. The fourth term extracts user preference from implicit matrix R to construct pairwise collaborative filtering loss.

In the generative process, we assume that the preference δ_{ijk} follows Bernoulli distribution. Similar loss function can be obtained with different assumption on r_{ijk} , such as Gaussian distribution. The big differences from other hybrid models are that

CTRank [10] uses LDA to extract topic proportions to learn latent factors for ranking, CDRank [13] exploits denoising autoencoder to mine feature representation of items.

$$\begin{aligned}
 L_{CVRank} = & \frac{\lambda_n}{2} \sum_j (x_j - X_{c,j^*})^2 + \frac{1}{2} \sum_j \sum_f \left(\mu_{jf}^2 + \sigma_{jf}^2 - \log \sigma_{jf}^2 - 1 \right)^2 \\
 & + \frac{\lambda_w}{2} \sum_l (W_l^2 + b_l^2) - \sum_{i,j,k} \frac{c_{ijk}}{2} \log \sigma(u_i^T v_j - u_i^T v_k) \\
 & + \frac{\lambda_u}{2} \sum_i u_i^T u_i + \frac{\lambda_v}{2} \sum_j (v_j - \mu_j)^2
 \end{aligned} \tag{6}$$

3.2 Neural Collaborative Filtering

Hybrid models like CDL, CDRank, CVAE and our proposed model CVRank all consist of two key components: a type of neural network and a type of collaborative filtering. Item latent factors and extracted item feature vectors serve as the bridge. We denote these kinds of methods as neural collaborative filtering.

The neural network part extracts effective feature representation from content information. The collaborative filtering part learns user and item latent factors from rating information and the feature representation. These latent factors are able to in turn promote the feature learning. [1] called these methods as tightly coupled methods. We propose a generic loss function for solving this type of neural collaborative filtering as (7).

$$L_{NCF} = L_{content} + L_{cf} + L_{bridge} \tag{7}$$

$$L_{bridge} = \frac{\lambda_v}{2} \sum_j (v_j - f(x_j))^2 \tag{8}$$

L_{NCF} is the total loss we want to minimize. $L_{content}$ is the loss of neural network component. L_{cf} is the loss of collaborative filtering component. L_{bridge} is the bridge loss between two components and is calculated by (8). The minimization of (7) uses coordinate ascent by alternatively optimizing latent factors U, V and weights W, biases b. Given the current W and b, we update U and V by minimizing the sum of $L_{content}$ and L_{bridge} . Given U and V, we can learn W and b using back-propagation learning algorithm. By updating U, V, W and b alternately, we can find a local optimum for L_{NCF} .

Different neural collaborative filtering method can be acquired by changing one of the two components. As for neural network, we can choose denoising autoencoder [1] or variational autoencoder [14]. As for collaborative filtering, we can consider point-wise or pairwise methods. Among pairwise methods, we have many pairwise ranking losses. Take CDRank as example, it uses denoising autoencoder and pairwise collaborative filtering with squared loss. In this situation, $L_{content}$ is the sum of reconstruction loss and regularization as (9), L_{cf} is the log loss of all training samples as (10). We will see that the sum of $L_{content}$, L_{cf} and L_{bridge} is exactly the same as the objective function in [1].

$$L_{content} = \frac{\lambda_n}{2} \sum_j (x_j - X_{c,j*})^2 + \frac{\lambda_w}{2} \sum_l (W_l^2 + b_l^2) \quad (9)$$

$$L_{cf} = \sum_{i,j,k} \frac{c_{ijk}}{2} ((r_{ij} - r_{ik}) - (u_i^T v_j - u_i^T v_k))^2 + \frac{\lambda_u}{2} \sum_i u_i^T u_i \quad (10)$$

4 Experiments

4.1 Dataset

CiteULike dataset is collected from citeulike.org [12]. In CiteULike, users have their own article libraries. If an article is in a user’s library, we take it as a positive interaction between this user and this article. CiteULike dataset includes 5551 users, 16980 items and 204986 interactions. The ratio of observed interactions to all user-item pairs equals 0.02%. The title and abstract of articles are regarded as content information to be incorporated. After text preprocessing, the vocabulary size is 8000, and each article is represented by the words in its title and abstract.

4.2 Evaluation

We use the same evaluation measure as in [14]. For each user, we randomly select P articles from his library, put them in the train set and put remaining articles in the test set. P controls the sparsity of training data. When $P = 1$, there are 199435 interactions in test set, 5551 interactions in train set. Sparsity equals the ratio of the number of positive interactions to the product of user number and item number. The train set sparsity is 0.03% and 0.06% respectively given $p = 5, 10$. Moreover, for each user we randomly select $T = 5$ articles from his library, put them in the test set and put remaining articles in the train set. The train set sparsity is 0.18%.

After optimizing the model on training set, we use it to recommend articles to users. Given user u , all items’ prediction scores are calculated by (1), and the recommendation list is generated as items of top M highest scores. The recall@ M for a user is defined as follows:

$$\text{recall@M} = \frac{\text{number of items the user likes in top } M}{\text{total number of items the user likes}}$$

The performance of model is calculated as average recall from all users.

4.3 Performance Comparison

Experiments Settings. For pure collaborative filtering, $K = 50$. For neural collaborative filtering methods, $K = 50$, $a = 1$, $b = 0.01$. A SDAE architecture ‘8000-200-100-50-100-200-8000’ is used for neural collaborative filtering methods that combine SDAE and collaborative filtering. A VAE architecture ‘8000-200-100-50-100-200-

8000' is used for neural collaborative filtering methods that combine VAE and collaborative filtering. In the pretraining of neural collaborative filtering methods, SDAE and VAE take corrupted input X_0 which is obtained from clean input X_c using a masking noise level of 0.3.

Figure 2 shows the results that compare BPR, CDL, CDRank, CVAE and CVRank under different sparse settings. It can be observed that neural collaborative filtering methods outperform the pure collaborative filtering method. Deep learning approach can make great use of content information to improve recommendation accuracy. Both CDRank and CVRank achieve stable performance under different P. CVRank achieves comparable results to CVAE under $P = 5$, and better results than CVAE under $P = 10$. Table 1 shows the recall growth of neural collaborative filtering, calculated by the ratio of the difference between recall of neural collaborative filtering and recall of pure collaborative filtering to recall of pure collaborative filtering. The recall growth under $P = 5$ and 10 is much higher than under $T = 5$. When data is sparse, incorporating content information is of great importance to recommendation accuracy.

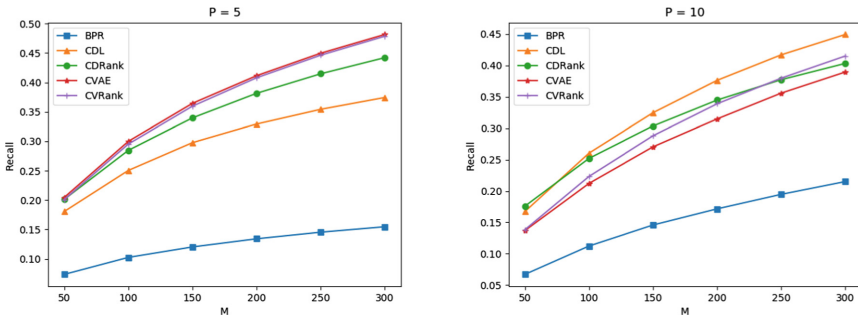


Fig. 2. Performance comparison of BPR, CDL, CDRank, CVAE, CVRank under different P. On the left $P = 5$. On the right $P = 10$.

Table 1. The recall growth of neural collaborative filtering under different data sparsity.

	Sparsity	CDL	CDRank	CVAE	CVRank
$P = 5$	0.03%	1.4770	1.8275	2.0328	1.9927
$P = 10$	0.06%	1.2256	1.0812	0.8532	0.9740
$T = 5$	0.18%	0.2178	0.1511	0.2261	0.1652

Impact of λ_r . We add a factor of λ_r to the first line of (6). The value of λ_r controls the penalty of variational autoencoder loss. In order to investigate the difference between focusing more on content information and focusing more on rating information, we change the value of λ_r . When λ_r is large, the content information dominates the learning of item latent vector. When λ_r is small, the performance mainly depends on the pairwise ranking model. The left part of Fig. 3 shows the performance of CVRank

for different values of λ_r under $P = 5$ while fixing other hyper parameters. It can be seen that the recall is lower whether λ_r is too large or too small. As a result, we need to choose an appropriate λ_r to balance the impact of neural network and the impact of pairwise ranking.

Impact of K. We explore the dimensionality of latent space’s effects on recommendation performance. The right part of Fig. 3 shows the performance of CVRank for different values of the number of latent factors K. It can be seen that $K = 10$ and $K = 20$ have much lower recall than $K = 50$. The main reason is that when K is too small, the low dimensional latent space does not have enough representation ability for content. A proper value of K is able to help make the most of neural network and content information of items.

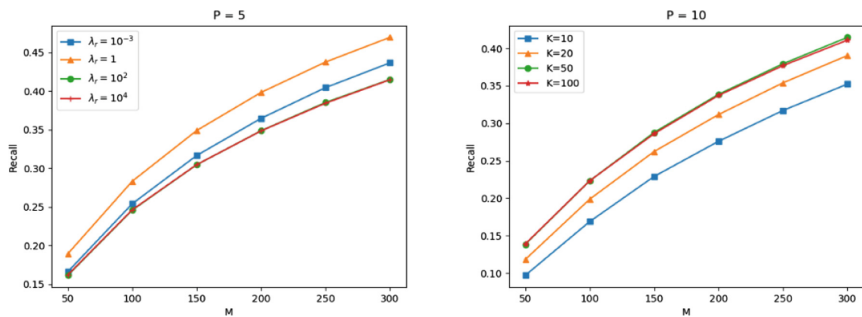


Fig. 3. Performance comparison of CVRank for different values of λ_r and K.

5 Conclusion

In this paper, we propose a collaborative variational ranking model that combines variational autoencoder with pairwise ranking based collaborative filtering, named CVRank. The introduction of content information of items leads to a great improvement on recommendation accuracy, especially in the case of high data sparsity. Then we sum up the recently proposed hybrid models like CDL, CDRank and CVAE as neural collaborative filtering, and present a generic optimization criterion for them. Experimental results show that the recall of CVRank is lower whether λ_r is too large or too small, and the dimensionality of latent space should be large enough so that the model has enough representation ability for content. Recall growth of neural collaborative filtering is larger when data is sparser. Other deep learning models and the incorporation of other multimedia content information can be explored in the future.

References

1. Wang, H., Wang, N., Yeung, D.Y.: Collaborative deep learning for recommender systems, pp. 1235–1244 (2014)
2. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. *Computer* **42**(8), 30–37 (2009)
3. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: Eighth IEEE International Conference on Data Mining, pp. 263–272. IEEE (2009)
4. Rendle, S., Freudenthaler, C., Gantner, Z., et al.: BPR: bayesian personalized ranking from implicit feedback. In: Conference on Uncertainty in Artificial Intelligence, pp. 452–461. AUAI Press (2009)
5. Pan, R., Zhou, Y., Cao, B., et al.: One-class collaborative filtering. In: Eighth IEEE International Conference on Data Mining, pp. 502–511. IEEE Computer Society (2008)
6. Li, Y., Hu, J., Zhai, C.X., et al.: Improving one-class collaborative filtering by incorporating rich user information, vol. 38, no. 2, pp. 959–968 (2010)
7. Rendle, S., Freudenthaler, C.: Improving pairwise learning for item recommendation from implicit feedback. In: ACM International Conference on Web Search and Data Mining, pp. 273–282. ACM (2014)
8. Joachims, T.: Optimizing search engines using click through data. In: ACM Conference on Knowledge Discovery and Data Mining, pp. 133–142. ACM (2002)
9. Freund, Y., Iyer, R., Schapire, R.E., et al.: An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.* **4**(6), 170–178 (1999)
10. Yao, W., He, J., Wang, H., et al.: Collaborative topic ranking: leveraging item meta-data for sparsity reduction. In: Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 374–380. AAAI Press (2015)
11. Kula, M.: Metadata embeddings for user and item cold-start recommendations. In: Proceedings of the 2nd Workshop on New Trends in Content-based Recommender Systems, co-located with the 8th ACM Conference on Recommender Systems, pp. 14–21. CEUR Workshop Proceedings (2015)
12. Wang, C., Blei, D.M.: Collaborative topic modeling for recommending scientific articles. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 448–456. ACM (2011)
13. Ying, H., Chen, L., Xiong, Y., Wu, J.: Collaborative deep ranking: a hybrid pair-wise recommendation algorithm with implicit feedback. In: Bailey, J., Khan, L., Washio, T., Dobbie, G., Huang, J.Z., Wang, R. (eds.) PAKDD 2016. LNCS (LNAI), vol. 9652, pp. 555–567. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-31750-2_44
14. Li, X., She, J.: Collaborative variational autoencoder for recommender systems. In: Proceedings of the ACM SIGKDD International Conference, pp. 305–314. ACM (2017)
15. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: International Conference on Neural Information Processing Systems, pp. 1257–1264. Curran Associates Inc. (2007)
16. Liou, C.Y., Cheng, W.C., Liou, J.W., et al.: Autoencoder for words. *Neurocomputing* **139**, 84–96 (2014)
17. Vincent, P., Larochelle, H., Bengio, Y., et al.: Extracting and composing robust features with denoising autoencoders. In: International Conference on Machine Learning, pp. 1096–1103. ACM (2008)
18. Wang, H., Shi, X., Yeung, D.Y.: Relational stacked denoising autoencoder for tag recommendation. In: Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 3052–3058. AAAI Press (2015)
19. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: Proceedings of the 2nd International Conference on Learning Representations. Preprint at [arXiv:1312.6114](https://arxiv.org/abs/1312.6114) (2013)



Overlap-Based Undersampling for Improving Imbalanced Data Classification

Pattaramon Vuttipittayamongkol¹(✉), Eyad Elyan¹, Andrei Petrovski¹,
and Chrisina Jayne²

¹ Robert Gordon University, Aberdeen, UK
{p.vuttipittayamongkol,e.elyan,a.petrovski}@rgu.ac.uk

² Oxford Brookes University, Oxford, UK
cjayne@brookes.ac.uk

Abstract. Classification of imbalanced data remains an important field in machine learning. Several methods have been proposed to address the class imbalance problem including data resampling, adaptive learning and cost adjusting algorithms. Data resampling methods are widely used due to their simplicity and flexibility. Most existing resampling techniques aim at rebalancing class distribution. However, class imbalance is not the only factor that impacts the performance of the learning algorithm. Class overlap has proved to have a higher impact on the classification of imbalanced datasets than the dominance of the negative class. In this paper, we propose a new undersampling method that eliminates negative instances from the overlapping region and hence improves the visibility of the minority instances. Testing and evaluating the proposed method using 36 public imbalanced datasets showed statistically significant improvements in classification performance.

Keywords: Undersampling · Overlap · Imbalanced data
Classification · Fuzzy C-means · Resampling

1 Introduction

In classification, sufficient data with balanced class distribution often results in more accurate models. However, in many real-world scenarios, datasets contain relatively few samples that belong to the class of interest, *e.g.* in fraud detection, where there are considerably more instances representing legitimate transactions. Such data form is so called imbalanced datasets. In a binary imbalanced dataset, the class with more instances is referred to as *the majority* or *negative class* whereas the rare class is regarded as *the minority* or *positive class*.

Generally, available learning algorithms are not designed to handle classification of datasets with skewed distributions. Without appropriate adjustments, the minority class tend to be overlooked, and hence are likely to be misclassified. In addition, imbalanced datasets also often suffer from class overlap [6], which proved to have a higher impact on classification than class imbalance [3, 7, 9, 15].

Methods for handling imbalanced datasets can be grouped into two main categories: data-level and algorithm-level [11, 16]. Algorithm-level methods mostly involve modifying existing learning algorithms to account for class imbalance [9, 13, 14]. Data-level methods typically reconstruct the original dataset into a more class-balanced version. This is often achieved by means of resampling, which includes undersampling and oversampling. Undersampling reduces instances in the majority class; in contrast, oversampling increases instances in the minority class. The advantages of data-level methods over algorithm-level ones are that no deep understandings of the learning algorithm are required [2] and it is flexible to any learning algorithm.

Among undersampling techniques, *k-means* has been widely utilised in recent literature [2, 10–12]. By applying *k-means*, the majority class is divided into clusters before undersampling is performed resulting in a more balanced and diversified class distribution of the data. However, these approaches as well as most existing undersampling methods only aim at data rebalancing and neglect the overlap issue, which may need a closer attention.

It has been suggested that class imbalance on its own may not affect classifier’s performance. Japkowicz *et al.* [8] and Denil *et al.* [3] demonstrated that when sufficient training data was available, any extent of imbalance did not hinder classification. On the other hand, class overlap has been reported to cause more deteriorations than class imbalance [3, 15].

In this paper, we propose a new undersampling framework that reduces the dominance of the majority class instances and more importantly removes them from the overlapping region. For convenience, we refer to our *Overlap-Based Undersampling method* as *OBU*. The method incorporates a soft clustering algorithm to determine overlapped instances. We hypothesise that an instance with uncertain membership degrees assigned by the soft clustering algorithm is likely to be in the overlapping region. Then, using the proposed *OBU*, overlapped negative instances can be potentially removed. Subsequently, the visibility of the minority class to the learner will be improved leading to better classification without the need of data rebalancing. Extensive experiments on 36 public datasets showed significant improvements in classification over the baseline while in most cases, higher results against the state-of-the-art’s were achieved. *OBU* is demonstrated with a well known soft clustering algorithm, Fuzzy C-means (FCM); however, it is worth noting that any existing soft clustering algorithm can be applied. Therefore, the overlap-based undersampling method is a general framework for handling class overlap in imbalanced dataset classification.

2 Methods

2.1 Fuzzy C-Means Algorithm

Fuzzy c-means [1] is one of the most commonly-used soft clustering algorithms. Unlike hard clustering, soft clustering algorithms allow each data instance to be a member of many clusters with membership degrees between 0 and 1. FCM follows similar clustering procedure to *k-means*, a well-known hard clustering algorithm

except that FCM's objective function involves two additional parameters, which are the membership degree and the fuzziness degree.

In this paper, the main FCM parameter which is the C value (number of clusters) was set to equal 2 as it serves the propose of differentiating between the characteristics of the two classes in binary datasets.

2.2 The Proposed Overlap-Based Undersampling Algorithm

Unlike other clustering-based undersampling methods, our proposed framework uses membership degrees obtained from the clustering process to facilitate the elimination of negative instances from the overlapping region. Given an imbalanced training set D that is made of positive class (D_{pos}) and negative class (D_{neg}), a soft clustering is then applied to give each instance a negative membership degree m_d . In this paper, we use FCM as the soft clustering algorithm; however, OBU is a framework that can be adapted to use other soft clustering techniques. All indecisive negative instances whose membership degrees are vague are considered as part of the overlapping region. These instances are then removed from the training set D . The resulting undersampled training set will include the remaining negative instances d 's in $D_{neg_{OBU}}$ along with the positive instances in D_{pos} .

Since binary datasets are used in this paper, when applying soft clustering, the number of clusters is set to 2 to differentiate between the positive and negative classes. Thus, each instance is assigned with 2 membership degrees. These are the degrees of being in *cluster 1* and *cluster 2*, which sum up to 1. By default, the higher membership degree determines the predicted cluster of the instance. Thus, it can be said that a negative instance has been clustered to the correct class if the resulting negative membership degree is 0.5 or higher. On the other hand, it is considered misclustered when the negative membership degree is less than 0.5. In OBU, all misclustered negative instances are removed from the training set. In addition, to allow flexibility and avoid excessive eliminations, an elimination threshold α -cut is introduced. The α -cut is set such that any negative instance whose m_d is below the α -cut is removed from the training set. Finally, a fuzzy set $D_{neg_{OBU}}$ is expressed as

$$D_{neg_{OBU}} = \{d \in D_{neg} \mid m_d \geq \alpha\text{-cut}\} \quad (1)$$

In this paper, the α -cut values between 0.3 to 0.5 were empirically experimented to achieve the global α -cut that optimised the overall results. This will be discussed in the next section.

2.3 Selection Process

In our framework, when two clusters are created, they may not be readily matched with the two prior class labels. For linearly separable problems, this can be resolved by simply finding the dominant class of the cluster. However, in

a complex dataset where both imbalance and overlap exist, an alternative and principled approach to perform this matching process is needed.

Figure 1 shows a complex scenario where the data is both imbalanced (minority:majority = 3:10) and highly overlapped as an example. The negative and positive classes are presented with blue circle and red triangle, respectively. Performing FCM clustering on the data resulted in two clusters showed in the left diagram. The between-class border was roughly sketched with the grey line. There are 80 and 100 negative instances in the left and the right clusters, respectively. With OBU, the 100 negative instances in right cluster are supposed to be eliminated. Note that these are the majority of the negative class. Thus, a criterion to eliminate a smaller number of negative instances cannot be applied as a selection process of OBU. It is also worth pointing out that judging from the positive class is not valid for all cases either.

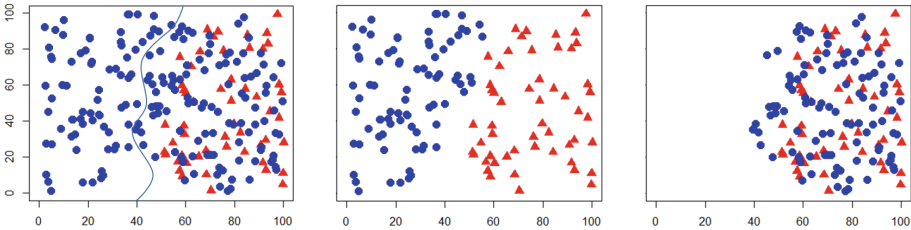


Fig. 1. Original data with the cluster boundary from FCM clustering (left), correctly undersampled data (middle), incorrectly undersampled data (right) (Color figure online)

In imbalanced and overlapping domains, besides this example, there are various problematic cases that prevent the clustering labels to be matched correctly with the actual labels. Therefore, our proposed framework has been adapted to handle such ambiguous scenarios (Fig. 2). To achieve this, negative instances in both clusters (batch 1 and 2) are considered for elimination, one at a time. One batch remains in the training set while the other is eliminated. As a result, two classification models are obtained. Since the positive class should be more visible in the overlapping region after applying OBU, the model obtained from the correctly undersampled case is expected to yield higher positive class accuracy. The selection is performed at this stage and the other model is discarded.

3 Experiment

3.1 Setup

Three different experiments were carried out to evaluate our proposed method. First, the datasets were classified after applying OBU. Second, we compared our results with the baseline which was simply classifying the datasets using Random

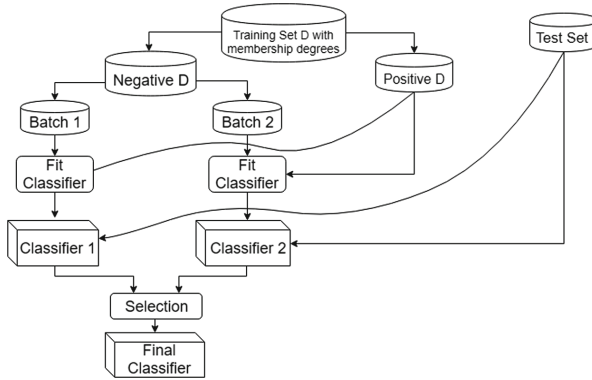


Fig. 2. Overlap-based undersampling framework

Forests with no undersampling. Finally, we reproduced one of the state-of-the-art methods [10], and compared it with our proposed technique.

Random Forest (RF) was chosen as the baseline as it proved to be amongst the top performing traditional machine learning algorithms [4, 5]. For all three experiments, the default parameter settings for RF in *caret* package in *R* were used (500 trees, and \sqrt{n} features at each split, where n is the total number of features in the dataset). In the first experiment, α -cut was set to 0.45 based on empirical results over a range of α -cut values (0.3, 0.36, 0.42, 0.45, and 0.5). The full code for reproducing is available on *GitHub*¹.

The partitioning ratio of the training and testing sets is 80:20, and the 10-fold cross-validation technique was adopted across the three experiments.

3.2 Datasets

In all experiments, 36 frequently used datasets in class-imbalance classification were selected. These datasets are obtained from UCI and KEEL repositories. Shown in Table 1, these datasets vary in terms of size (129 to 5472 instances), imbalance ratio (1.87 to 129.44), and number of features (3 to 19). These variations allowed the proposed technique to be tested for its robustness under different situations.

3.3 Evaluation Metrics

The evaluation metrics used in the experiments were sensitivity and balance accuracy. Sensitivity is the true positive rate (TPR). It receives the most attention in imbalanced data classification since the positive class is of higher concern. Higher sensitivity is desired; however, high sensitivity by itself is not sufficient to assess a classifier. An overall classification performance is also needed.

¹ https://github.com/fonkafon/Overlap_based_Undersampling.

Balance accuracy is a measure of the overall performance. Shown in Eq. (2), balance accuracy is the average of the TPR and the true negative rate (TNR). It is preferable to the traditional accuracy, which neglects the fact that the cardinality of the positive class is relatively very small in imbalanced domains.

$$Balance\ accuracy = \frac{TPR + TNR}{2} \tag{2}$$

4 Results and Discussion

4.1 Results

As can be seen in Table 1, the proposed OBU produced the most favourable results among the different experiments and outperformed the recently proposed undersampling technique, *k-means clustering-based undersampling*, which proved to give comparable results with state-of-the-art methods [10]. Wilcoxon signed rank tests indicate that by employing our approach, the classification improvements over the baseline were statistically significant as detailed below.

Table 1. Comparative results

Dataset	Instances	ImbRatio	Features	nMajLeft	OBU_Sens	OBU_BA	kmeans_Sens	kmeans_BA	Org_Sens	Org_BA
Abalone09-18	731	16.4	8	287	0.75	0.7181	0.5	0.6186	0.375	0.6839
Ecol1	336	3.36	7	170	1	0.9412	0.8	0.8902	0.8	0.8706
Ecol2	336	5.46	7	146	0.9	0.9232	0.8	0.9	0.8	0.9
Glass016vs2	192	10.29	9	5	1	0.5571	0.3333	0.3952	0	0.5
Glass4	214	15.47	9	7	1	0.825	0.5	0.6875	0.5	0.7375
Haberman	306	2.78	3	97	0.75	0.6306	0.625	0.6125	0.3125	0.534
Ecol0137vs26	281	39.14	7	89	1	0.9907	1	0.6111	1	0.9815
Ecol4	336	15.8	7	169	1	1	1	0.9841	0.5	0.75
New-thyroid1	215	5.14	5	78	1	0.9722	1	0.9583	0.8571	0.9286
Vowel0	988	9.98	13	312	1	0.986	1	0.9078	0.9444	0.9722
Yeast5	1484	32.73	8	776	1	0.9688	1	0.941	0.5	0.7483
Iris0	150	2	4	77	1	1	1	1	1	1
Page-blocks13vs2	472	15.86	10	351	1	1	1	0.9773	1	1
Shuttle2vs4	129	20.5	9	60	1	1	1	1	1	1
Glass0	214	2.06	9	6	1	0.6429	0.7143	0.8393	0.5714	0.7857
Glass0123vs456	214	3.2	9	5	1	0.5156	0.9	0.9188	0.8	0.8688
Glass1	214	1.82	9	5	1	0.5185	0.6667	0.7407	0.6667	0.8148
Glass6	214	6.38	9	4	1	0.6351	0.8	0.8865	0.6	0.8
Pima	768	1.87	8	86	0.9057	0.5028	0.7736	0.7568	0.6415	0.7308
Vehicle1	846	2.9	18	187	0.8372	0.5306	0.8372	0.8106	0.5814	0.7307
Vehicle2	846	2.88	18	233	1	0.772	0.9767	0.9644	0.9535	0.9767
Yeast1	1484	2.46	8	477	0.8824	0.7042	0.8588	0.7422	0.5647	0.7255
Ecol3	336	8.6	7	185	0.8571	0.7952	0.8571	0.8286	0.2857	0.6345
Glass016vs5	184	19.44	9	5	1	0.5143	1	0.9	0	0.5
Glass5	214	22.78	9	7	1	0.8293	1	0.8902	0	0.5
Segment0	2308	6.02	19	577	1	0.9899	1	0.9937	0.9846	0.9923
Yeast05679vs4	528	9.35	8	209	0.8	0.8526	1	0.7526	0.5	0.7447
Yeast1289vs7	693	22.1	8	461	0.3333	0.6639	1	0.5027	0.1667	0.5806
Yeast1458vs7	459	14.3	8	337	0.1667	0.5455	0.5	0.428	0	0.5
Yeast4	1484	28.1	8	768	0.8	0.8493	1	0.507	0.3	0.65
Yeast6	1484	41.4	8	775	0.7143	0.8122	1	0.5173	0.4286	0.7126
Abalone19	4174	129.44	8	1661	0.5	0.5707	0.8333	0.6848	0	0.5
Glass2	214	11.59	9	39	0.6667	0.5	1	0.7051	0	0.5
Vehicle3	846	2.99	18	341	0.7857	0.7381	0.8571	0.8095	0.3571	0.6349
Yeast2vs4	514	9.08	8	164	0.8	0.8946	1	0.9402	0.5	0.75
Yeast3	1484	8.1	8	675	0.7813	0.8414	1	0.9015	0.625	0.8068

The overlap-based undersampling method produced the best results across 26 and 23 datasets in terms of sensitivity and balance accuracy, respectively. It is

also shown that on 14 of these datasets, OBU won in both metrics, which far outnumbered the k-means based technique. These results are highlighted in Table 1 as **bold** indicating that our method is winning and in *italic* indicating that our method is winning over one method and having similar (tie) performance to the other. However, it should be noted that most of these ties occurred with the sensitivity value of 100%. In other words, these datasets are imbalanced yet already linearly separable and do not need to be resampled. The results suggest that our method improved the classification on most of the datasets. At the same time, it was unlikely to hurt the classification performance on a linearly separable dataset, where the classification accuracy was already at the maximum. This could be attributed to the fact that OBU only undersamples negative instances in the overlapping region.

To further assess the significance of the improvements using our proposed framework, one-tailed Wilcoxon signed rank tests were carried out. The resulting p-values for OBU paired with the baseline and k-means undersampling on the sensitivity were 1.16×10^{-6} and 0.473, and of balance accuracy were 0.108 and 0.271, respectively. These results suggest that at the significance level of 0.05, our method led to statistically significant improvements over the baseline on sensitivity while the other pairs had insufficient evidence to conclude.

4.2 Discussion

By applying the proposed undersampling framework, it was possible to remove negative instances from the overlapping region, where misclassification often occurs. This made positive instances more visible to the learner, and as a result, the sensitivity values of most datasets were improved.

Table 1 presents 4 groups of the results based on the classification improvements obtained with OBU. The first group is the datasets that OBU produced winning results in both sensitivity and balance accuracy. The second group has winning sensitivity values but not balance accuracy. This must have occurred due to the tradeoff between the accuracy of the positive and the negative classes in the overlapping region, and thus can be slightly adjusted for a more compromised result. In the third group, OBU produced the best results in balance accuracy, but not the sensitivity. This implies that more undersampling can be applied to further eliminate the overlapped negative instances. For the last group, our approach outperformed the baseline but not the k-means based undersampling method. Our assumption is that the variation in the results is due to the inherent data characteristics. Also, it should be noted that these results are based on a global empirical setting of the α -cut value. In other words, fine-tuning this value for individual datasets could potentially improve the results further.

Unlike common undersampling methods, our framework minimises information loss by undersampling from the overlapping region only, which also results in maximising the visibility of the positive instances. This is evident by higher sensitivity and balance accuracy obtained in most datasets as the tradeoff between lower negative accuracy and higher positive accuracy has been compromised.

5 Conclusions and Future Work

In this paper, a new overlap-based undersampling framework was proposed. By removing negative instances from the overlapping region, an exceptional improvement in the minority class accuracy with a relatively small trade-off of the TNR was achieved, resulting in a significant improvement in sensitivity. This technique has proved to enhance the classification of well-known imbalanced datasets and outperformed the state-of-the-art method in most of the demonstrated cases. These results can be attributed to several advantages of our method over other common undersampling techniques. These include: first, the amount of undersampling done by OBU is proportional to the overlap degree; second, OBU is unlikely to eliminate instances outside the overlapping region, which minimises information loss. For a future direction, we are experimenting to further improve this undersampling framework, especially in the selection process and an adaptive α -cut approach.

References

1. Bezdek, J.C., Ehrlich, R., Full, W.: FCM: the fuzzy c -means clustering algorithm. *Comput. Geosci.* **10**(2–3), 191–203 (1984)
2. Branco, P., Torgo, L., Ribeiro, R.P.: A survey of predictive modeling on imbalanced domains. *ACM Comput. Surv. (CSUR)* **49**(2), 31 (2016)
3. Denil, M., Trappenberg, T.: Overlap versus imbalance. In: Farzindar, A., Kešelj, V. (eds.) *AI 2010. LNCS (LNAI)*, vol. 6085, pp. 220–231. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13059-5_22
4. Elyan, E., Gaber, M.M.: A fine-grained random forests using class decomposition: an application to medical diagnosis. *Neural Comput. Appl.* **27**(8), 2279–2288 (2016)
5. Elyan, E., Gaber, M.M.: A genetic algorithm approach to optimising random forests applied to class engineered data. *Inf. Sci.* **384**, 220–234 (2017)
6. Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., Herrera, F.: A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **42**(4), 463–484 (2012)
7. García, V., Mollineda, R.A., Sánchez, J.S.: On the k -NN performance in a challenging scenario of imbalance and overlapping. *Pattern Anal. Appl.* **11**(3–4), 269–280 (2008)
8. Japkowicz, N., Stephen, S.: The class imbalance problem: a systematic study. *Intell. Data Anal.* **6**(5), 429–449 (2002)
9. Lee, H.K., Kim, S.B.: An overlap-sensitive margin classifier for imbalanced and overlapping data. *Expert Syst. Appl.* **98**, 72–83 (2018)
10. Lin, W.C., Tsai, C.F., Hu, Y.H., Jhang, J.S.: Clustering-based undersampling in class-imbalanced data. *Inf. Sci.* **409**, 17–26 (2017)
11. Ng, W.W., Hu, J., Yeung, D.S., Yin, S., Roli, F.: Diversified sensitivity-based undersampling for imbalance classification problems. *IEEE Trans. Cybern.* **45**(11), 2402–2412 (2015)
12. Ofek, N., Rokach, L., Stern, R., Shabtai, A.: Fast-CBUS: a fast clustering-based undersampling method for addressing the class imbalance problem. *Neurocomputing* **243**, 88–102 (2017)

13. Oh, S.H.: Error back-propagation algorithm for classification of imbalanced data. *Neurocomputing* **74**(6), 1058–1061 (2011)
14. Song, J., Lu, X., Wu, X.: An improved adaboost algorithm for unbalanced classification data. In: Sixth International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2009, vol. 1, pp. 109–113. IEEE (2009)
15. Visa, S., Ralescu, A.: Learning imbalanced and overlapping classes using fuzzy sets. In: Proceedings of the ICML, vol. 3 (2003)
16. Yen, S.J., Lee, Y.S.: Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Syst. Appl.* **36**(3), 5718–5727 (2009)



Predicting Online Review Scores Across Reviewer Categories

Michela Fazzolari¹(✉) , Marinella Petrocchi¹ , and Angelo Spognardi² 

¹ Istituto di Informatica e Telematica, CNR, Pisa, Italy
{m.fazzolari,m.petrocchi}@iit.cnr.it

² Dipartimento di Informatica, Sapienza Università di Roma, Rome, Italy
spognardi@di.uniroma1.it

Abstract. In this paper, we propose and test an approach based on regression models, to predict the review score of an item, across different reviewer categories. The analysis is based on a public dataset with more than 2.5 million hotel reviews, belonging to five specific reviewers' categories. We first compute the relation between the average scores associated with the different categories and generate the corresponding regression model. Then, the extracted model is used for prediction: given the average score of a hotel according to a reviewer category, it predicts the average score associated with another category.

Keywords: Online reviews · Subjectivity · Predictive analysis
Reviews score prediction · Reviewers category · Score re-balancing

1 Introduction

Review platforms allow customers to submit a textual review, plus a numerical score, as a way for users to express their opinions about products and services. An overall score is usually computed by averaging the single scores: products can be sorted *wrt* their overall score, allowing next users to select the *best* product. The overall score could not be sufficient to convey the point of view of specific users. For example, a backpack traveler can appreciate the low rates of a hotel, while businessmen can rather point out the lack of fine restaurants nearby the same hotel. The same product can be reviewed in different ways, depending on the characteristics of the customer. For the sake of simplification, let the reader consider a hotel with only two reviews, one by the backpack traveler and the other by the businessman. When the next potential customer consults the overall score for that hotel, she simply gets the average of the two reviews. Such value does not consider the customer characteristics: as a backpack traveler, she will probably be more interested in reviews given by similar users.

In this work, we investigate the possibility of automatically compute the bias between two different categories of users, by relying on available reviews for similar products and services. Most of the existing platforms allow the users to obtain the score projected only on one reviewer category. Here, we show how to predict the score for a missing category of users, by solely relying on the score

given for the other categories. The benefits are that (1) in some cases, the reviews for a certain user category can be unavailable, therefore that category of users has access to a biased score; with the proposed approach it is possible to derive the missing score for a certain target category starting from the score of another category; (2) the computed model can be used to re-balance the scores of the other user categories to obtain a refined (unbiased) score for a certain category, which can be used in conjunction with other user preferences to build a refined recommender system; (3) we can infer and reconstruct the missing information in reviews that is generally needed to all the mechanisms based on data analysis (like machine-learning or other data driven approaches, e.g., [1, 3, 14, 15]).

The analysis is grounded on a Booking dataset¹, with more than 2,500,000 hotel reviews, belonging to five specific reviewers categories. The results of the experiments show that the models constructed by applying a simple linear regression algorithm are accurate enough to estimate the score of one category given the score of another category. We obtain the best results by relying on homogeneous samples to construct the prediction models.

We further show an application of the prediction models, by employing them to build a compensation system that refines the Booking reputation model. The compensation system acts as a function that can be used to compute a better overall score, corrected to account the lack of a certain user category. The results of this application indicate that our models are capable to predict the score for a certain category of users (*target category*) with a higher accuracy than the more general overall score (computed without considering the target category).

Next section describes related work in the area. Section 3 introduces the dataset. In Sect. 4, we describe the methodology for score prediction and the application of the compensation procedure. Section 5 gives experiments and results. We conclude with final remarks.

2 Related Work

In recent years, a considerable attention has been directed to automatically mine useful knowledge from online reviews. Studies have been carried out for developing systems to support travellers' decisions. Work in [20] introduces a travel planning tool, which is based on user-generated contents, including reviews from TripAdvisor. In [17], a specific decision support system for restaurants is developed, grounded on data from TripAdvisor too. The authors of [7] also focus on improving the recommendation accuracy in a restaurant review scenario, by relying on soft clustering techniques.

The identification of helpful reviews has been also widely studied, even taking into account users' characteristics [2, 18, 19]. For example, in [8] the authors propose two ranking mechanisms: the first is consumer-oriented and it aims at ordering reviews according to their expected helpfulness, while the second is manufacturer-oriented and it ranks the reviews according to their expected effect

¹ <http://www.booking.com> - All URLs have been lastly accessed on July 22, 2018.

on sales. The contribution in [13] presents several text regression models, used to predict the helpfulness of online reviews, by considering characteristics of the reviewers, such as their reputation, engagement, and frequency of activities.

So called summarization systems aim at identifying representative expressions and opinions from customers' reviews [11, 12]. A further method to help customers in identifying the most important reviews is to predict the rating associated to a certain product, according to customers' characteristics. An example of this contribution is in [3], which, as done in the current proposal, considers the problem of hotels rating prediction. Findings are that travellers from different countries have different rating behaviours.

In [16], a unified probabilistic model is proposed, which integrates the advantages of collaborative filtering and opinion mining. The model first learns users' personalized preferences on different aspects from their past reviews, and then predicts the users' ratings for new products. The authors of [5] propose a probabilistic model that extracts aspects from movie reviews and determines the polarity of the sentiment and the rating of each aspect category. An original approach has been presented in [15], where the authors propose a method for predicting the review rating based on neural networks. Unlike previous neural network methods for sentiment prediction, the neural network takes into account not only semantic features, but also user characteristics.

This brief overview highlights a research direction aimed at extracting and/or predicting useful knowledge from reviews and associated metadata. Our work follows the same direction, specifically considering numerical scores and the capability to predict them across different categories of users. A preliminary investigation was carried out in [4], where the authors manually showed the feasibility of applying a simple multiplicative factor to the scores of different categories, even if considering a negligible number of hotel samples. Here, we propose a straightforward but effective linear regression model to predict reviewers' ratings and we demonstrate that this technique well behaves in reaching the goal.

3 Dataset Definition

We rely on a dataset of hotel reviews collected from the Booking website, from June to August 2016. We consider 10 cities and - for each city - all the reviews associated to hotels with at least 1,000 reviews, in all languages. In fact, the proposed method exploits only the review numerical score and it does not rely on the textual part of the reviews. The 10 cities are Cape Town, Dubai, London, Los Angeles, New York, Paris, Pisa, Rio De Janeiro, Rome, Sydney. Overall, the dataset includes 2,598,111 reviews, referring to 1,202 hotels. Table 1 recaps some statistics extracted from the dataset described so far.

Booking Scoring System. The Booking website only accepts verified reviews, after an actual stay in the facility. This is achieved since the user that makes the reservation through the Booking website, only after the trip, she receives an email, asking her to evaluate her recent stay. To keep the rating score and the review content relevant and updated for the upcoming users, Booking archives

Table 1. Statistics of the dataset

City name	#Total reviews	#Hotels
Cape Town	10,586	8
Dubai	637,677	272
London	726,327	317
Los Angeles	99,937	56
New York	416,597	167
Paris	139,216	95
Pisa	47,632	31
Rio De Janeiro	118,330	63
Rome	204,634	107
Sydney	197,175	86

reviews older than 24 months. There are six features to be rated, namely *staff*, *facilities*, *cleanliness*, *comfort*, *value for money* and *location* and for each of them, the reviewer can express her satisfaction by assigning an emoticon and each emoticon corresponds to a possible value, namely 2.5, 5, 7.5 and 10. The overall review score is computed as the average of the single features scores, thus the range of possible values for the review score is from 2.5 to 10. Therefore, it is 2.5 (resp. 10) if the user assigns the lowest (resp. highest) score to all the features. Non rated features do not impact the overall score.

User Categories. During the review process, the user is asked to give some answers related to her staying, which are then used to generate the *tags* field of the review. Some of these tags describe the type of traveler. The traveler categories identified by Booking are five: *solo*, *couples*, *families*, *group of friends* and *business travelers*. Therefore, for each review there is a tag that specifies the *user category* and user categories can be applied to filter reviews, so that a user can visualize only reviews written by a certain traveler type.

4 Methodology

In this section, we present the methodology applied to construct models that can be used to predict the review score across different user categories. We rely on a regression analysis [6], in particular, we investigate if the average score of a hotel given by a user category can be effectively predicted by knowing the average score given by another user category. The obtained models can be used to predict the score of one category, when this is missing, or to obtain the “compensated overall score” for one category, by compensating all the reviews available from other categories. In the following, the proposed approach is described in details.

4.1 Useful Notions

A review and its associated numerical score is given by a user over an object o . Hence, we consider a set of users \mathcal{U} , a set of objects \mathcal{O} , a set of atomic scores \mathcal{S} ,

and a set of users categories \mathcal{C} . We define $\mathcal{R}_c = \mathcal{U} \times \mathcal{C} \times \mathcal{O} \times \mathcal{S}$ as the set of the reviews posted by the users of category c .

The scores associated to each review are assumed belonging to a domain that provides usual arithmetic operations, and it is therefore possible to calculate the average of a set of scores. In the investigated scenario, the atomic scores correspond to the averages of the feature scores.

Given an object $o \in \mathcal{O}$, we introduce the function $\gamma : \mathcal{O} \times \mathcal{C} \rightarrow \mathcal{S}$, which gives the score of an object o for a certain category of users, and it is defined as the average of all the scores assigned to o by all the users of that category:

$$\gamma(o, c) = \mathbf{avg}(s | (u, c, o, s) \in \mathcal{R}_c) \quad \forall o \in \mathcal{O} \quad \forall c \in \mathcal{C} \quad (1)$$

where \mathbf{avg} is the function returning the average of a set of review scores.

4.2 Regression Analysis and Score Prediction

Following the intuition that reviewers of different categories may appreciate different features of the same object, we investigate if the relationship between the scores of different categories can be predicted. We consider a fixed set \mathcal{X} of objects and two fixed categories c_1 and c_2 . By applying a simple linear regression model [6] to the set of sample points $\gamma(x, c_1)$ and $\gamma(x, c_2)$, for each object $x \in \mathcal{X}$, it is possible to define the tuning function $\Delta(c_1, c_2, \mathcal{X})$, which allows to pass from the score of users $\in c_1$ to the one of users $\in c_2$. Then, the predicted score of the object o for the category c_1 is computed by the prediction function $\pi : \mathcal{O} \times \mathcal{C} \times \mathcal{C} \rightarrow \mathcal{S}$, which is defined as a function f of the score assigned to the object o for the category c_2 and the tuning function Δ between c_2 and c_1 .

$$\pi(o, c_1, c_2) = f(\gamma(o, c_2), \Delta(c_2, c_1, \mathcal{X})) \quad \forall o \in \mathcal{O} \setminus \mathcal{X} \quad \forall c_1, c_2 \in \mathcal{C} \quad (2)$$

It is worth noting that the tuning function Δ is defined on a set of objects \mathcal{X} that does not include the objects on which we aim at predicting the score.

4.3 Overall Score Compensation

As usual on review sites, the overall score of an object is computed as the average of all the available review scores, coming from users of different categories. However, when a category is completely missing, the overall score can be considered biased. Then, we can use the prediction function π to *compensate* the lack of the scores of a category and correct the overall score obtained from the scores of the other categories. Therefore, the *compensated overall score* of the object o with respect to category c can be computed as:

$$\bar{S}(o, c) = \mathbf{avg}(\{\pi(o, c, c_j) | c_j \in \mathcal{C} \setminus \{c\}\} \cup \{\gamma(o, c)\}) \quad (3)$$

In general, the compensated overall score is a desirable value for users of all the categories, since it better reflects the real score of a given object: by

compensating the missing scores by means of the prediction function, it reduces the uncertainty of the available overall score.

From a different point of view, the compensated overall score allows the user of category c to obtain the review score of an object o as if all the review scores were submitted by reviewers from the same category. Clearly, this works even if $\gamma(o, c) = 0$, i.e., there are no reviews from users of category c .

5 Experimental Results

We describe the experiments to validate the methodology proposed in Sect. 4. As seen in Sect. 3, the dataset includes hotels of different cities and different quality levels, namely number of stars, from 1 to 5 (or NA, if not defined). We consider three scenarios to prove the approach effectiveness:

- Scenario 1: a single city, a single hotel quality level.
- Scenario 2: a single city, all the hotel quality levels.
- Scenario 3: all cities, all the hotel quality levels.

To reduce variability and limit overfitting [10], we performed a k -fold cross-validation, with $k = 5$, and averaged the results over the rounds. The performance of each prediction model is measured in terms of: (i) the Mean Absolute Error (MAE) value, which is the average magnitude of the errors in a set of predictions, considering only their absolute values, and (ii) the Root Mean Squared Error (RMSE), which is the square root of the average of the squared differences between predicted and actual values. They are computed as:

$$MAE = \frac{1}{N} \sum_{i=1}^N |\hat{\theta}_i - \theta_i| \quad RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (\hat{\theta}_i - \theta_i)^2} \quad (4)$$

where $\hat{\theta}_i$ is the predicted value, θ_i is the actual value, $|\hat{\theta}_i - \theta_i|$ is the absolute error and N is the number of predictions considered.

Both MAE and RMSE express the average prediction error of the model in terms of the variable of interest. In this case, they express the error in the range $[0, 10]$: the lower their values, the lower the errors. The RMSE gives relatively high weights to large errors, since the errors are first squared and then averaged. Thus, if the MAE is steady and the RMSE increases, this means that the variance associated with the frequency distribution of error magnitudes also increases.

In order to apply the proposed method to the dataset introduced in Sect. 3, we pre-processed the data: we first grouped the reviews according to the user categories, specified in the *tag field*. Then, we computed the function $\gamma(o, c)$, for each hotel o and for each category c . Thus, for each hotel situated in a city, the pre-processed dataset includes a row with hotel name, city and quality level (number of stars), plus the value of $\gamma(o, c)$ for each user category.

After this pre-processing stage, the dataset includes 1,202 patterns, corresponding to the number of hotels available in the original dataset. An excerpt of the pre-processed dataset is reported in Table 2.

Table 2. Excerpt of the pre-processed dataset

Hotel	City	Stars	Couples	Families	Friends	Solo	Business
Holiday Inn Sydney Airport	Sydney	4	8.58	8.57	8.71	8.50	8.28
Big Hostel	Sydney	NA	7.67	8.05	7.96	7.99	7.66
Ibis Budget - Enfield	Sydney	2	6.72	6.97	6.84	6.95	7.04
Marco Polo Motor Inn	Sydney	3	6.92	6.79	7.31	7.12	6.92

Table 3. Interpolation function: each category pair, 4* hotels, London.

	Couples		Families		Friends		Solo		Business	
	α	β	α	β	α	β	α	β	α	β
Couples	—		0.99	0.19	1	-0.10	0.94	0.71	0.92	0.89
Families	0.96	0.27	—		1	-0.11	0.92	0.77	0.91	0.93
Friends	0.91	0.80	0.93	0.70	—		0.87	1.36	0.86	1.44
Solo	0.97	0.07	0.98	0.06	0.98	-0.13	—		0.96	0.38
Business	1	-0.26	1.01	-0.31	1.03	-0.58	1.01	-0.14	—	

5.1 Scenario 1

For the first scenario, we consider the hotels located in London and with a quality level of 4 stars (136 samples). We firstly try to observe the correlation between the average scores for each pair of reviewer categories. The scatter-plots obtained by considering the $\gamma(o, c)$ function, for each pair of categories, on the x and y axes, respectively, indicate a strong positive linear correlation between the average scores given by pairs of categories, namely reviewers of different categories tend to have similar opinion for the same hotel. The scatter plots have not been reported here for the sake of brevity.

We, then, apply the linear regression procedure available in [9] to determine, for each pair of categories, the function that best fits the distributions. The obtained results are shown in Table 3, which reports, for each pair of categories, the function that best interpolates the given distribution. Such a function is a linear function with variable γ and with parameters α and β for each category c , namely $f = \alpha \cdot \gamma(o, c) + \beta$.

Then, to evaluate the soundness of the approach, we evaluated and report in Tables 4, 5 and 6 the MAE and the RMSE for all the performed experiments.

Observing the values of Table 4 for the London's hotels, we can see that both MAE and RMSE are limited by an upper bound of 0.17 and 0.23, respectively. The pairs that present lower MAE and RMSE are $(solo, business)$ (0.07 and 0.10) and $(families, couples)$ (0.10 and 0.13). Their models are more accurate than the others. Instead, the pair $(solo, friends)$ presents the highest average errors. Tables 5 and 6 show the results of the same experiment, considering the 4 stars hotels in Dubai and New York, resp. MAE and RMSE reported in Table 5

Table 4. MAE and RMSE, each pair of categories, 4 stars hotels, London.

	Couples		Families		Friends		Solo		Business	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Couples	—	—	0.10	0.14	0.13	0.17	0.14	0.18	0.13	0.16
Families	0.10	0.13	—	—	0.12	0.15	0.14	0.18	0.13	0.16
Friends	0.13	0.16	0.12	0.15	—	—	0.16	0.21	0.14	0.18
Solo	0.14	0.18	0.15	0.19	0.17	0.23	—	—	0.07	0.10
Business	0.13	0.17	0.13	0.17	0.16	0.20	0.07	0.11	—	—
Mean	0.125	0.16	0.125	0.16	0.145	0.19	0.13	0.17	0.12	0.15

Table 5. MAE and RMSE, each pair of categories, 4 stars hotels, Dubai.

	Couples		Families		Friends		Solo		Business	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Couples	—	—	0.16	0.20	0.14	0.18	0.13	0.17	0.11	0.14
Families	0.18	0.21	—	—	0.20	0.24	0.23	0.26	0.18	0.22
Friends	0.14	0.18	0.17	0.22	—	—	0.13	0.17	0.13	0.16
Solo	0.12	0.16	0.19	0.23	0.12	0.16	—	—	0.08	0.10
Business	0.11	0.14	0.16	0.20	0.13	0.15	0.08	0.10	—	—
Mean	0.137	0.172	0.17	0.212	0.147	0.182	0.142	0.175	0.125	0.155

Table 6. MAE and RMSE, each pair of categories, 4 stars hotels, New York.

	Couples		Families		Friends		Solo		Business	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Couples	—	—	0.12	0.16	0.11	0.15	0.10	0.13	0.09	0.12
Families	0.13	0.17	—	—	0.13	0.16	0.16	0.20	0.15	0.19
Friends	0.12	0.16	0.12	0.16	—	—	0.16	0.19	0.14	0.18
Solo	0.10	0.13	0.16	0.19	0.15	0.19	—	—	0.07	0.09
Business	0.10	0.13	0.17	0.20	0.15	0.20	0.08	0.10	—	—
Mean	0.11	0.147	0.142	0.177	0.135	0.175	0.125	0.155	0.112	0.145

are limited by an upper bound of 0.23 and 0.26, respectively. That upper bound is related to the pair (*families, solo*), while the lowest average error is achieved for the pair (*solo, business*). For the 4 stars New York hotels, Table 6, the worst average error is related to the pair (*business, families*), while the lowest one is achieved with the pair (*business, solo*).

Table 7. MAE and RMSE, each pair of categories, ALL hotels, London.

	Couples		Families		Friends		Solo		Business	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Couples	—	—	0.11	0.15	0.13	0.17	0.17	0.21	0.16	0.20
Families	0.11	0.15	—	—	0.12	0.15	0.17	0.22	0.16	0.21
Friends	0.13	0.16	0.11	0.14	—	—	0.17	0.22	0.16	0.20
Solo	0.15	0.19	0.16	0.21	0.17	0.22	—	—	0.08	0.10
Business	0.15	0.19	0.16	0.20	0.16	0.21	0.08	0.11	—	—
Mean	0.135	0.17	0.135	0.175	0.145	0.19	0.15	0.19	0.14	0.18

Table 8. MAE and RMSE, each pair of categories, ALL hotels, Dubai.

	Couples		Families		Friends		Solo		Business	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Couples	—	—	0.15	0.20	0.14	0.19	0.14	0.18	0.13	0.17
Families	0.16	0.21	—	—	0.17	0.22	0.20	0.25	0.17	0.22
Friends	0.14	0.19	0.16	0.21	—	—	0.14	0.19	0.14	0.19
Solo	0.13	0.18	0.18	0.23	0.14	0.19	—	—	0.09	0.13
Business	0.13	0.17	0.16	0.21	0.14	0.19	0.10	0.13	—	—
Mean	0.14	0.187	0.162	0.212	0.147	0.197	0.145	0.187	0.132	0.177

Table 9. MAE and RMSE, each pair of categories, ALL hotels, New York.

	Couples		Families		Friends		Solo		Business	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Couples	—	—	0.13	0.18	0.12	0.17	0.12	0.16	0.13	0.18
Families	0.13	0.17	—	—	0.12	0.16	0.15	0.19	0.15	0.19
Friends	0.12	0.17	0.13	0.17	—	—	0.16	0.20	0.16	0.22
Solo	0.12	0.16	0.15	0.19	0.15	0.19	—	—	0.09	0.12
Business	0.15	0.19	0.16	0.19	0.17	0.22	0.10	0.13	—	—
Mean	0.13	0.17	0.14	0.18	0.14	0.185	0.13	0.17	0.13	0.18

5.2 Scenario 2

For the second scenario, we consider all the London hotels, regardless of the number of their stars. By enlarging the scenario, we aim at assessing if the results previously obtained can be generalized to all the hotels, without distinction among their quality level. For this experiment, 317 samples are available. We apply again the linear regression procedure to obtain a model for each pair of categories. For the sake of brevity, we only report the MAE and RMSE, in Table 7. We notice that the most accurate model is obtained again with the

pair (*solo, business*) (as for the only 4 stars London hotels, analysed in Table 4, followed by (*friends, families*) and (*families, couples*).

By comparing these errors with the ones in the previous experiment, their values increase, even if by a smaller quantity. Thus, even in this scenario it is possible to predict, with a high accuracy, the score of a category given the score of another category, by considering hotels of any quality level. This is also confirmed by the error values shown in Tables 8 and 9, which consider all the hotels of Dubai and New York available in our dataset. For example, comparing the last lines of the two tables referred to Dubai (Tables 5 and 8), the mean errors are either the same or they are a bit lower in the 4 stars hotels case.

5.3 Scenario 3

We consider all the 1,202 patterns available in our dataset. This includes hotels of 10 different cities and with different quality levels. We apply the linear regression algorithm to each pair of categories and we determine a generalized regression model for each pair. The MAE and RMSE are in Table 10: the outcome from the previous experiments are still valid. Indeed, even if the mean errors increase, *wrt* considering only hotels in a single city, the highest errors never exceed 0.17 (MAE) and 0.23 (RMSE). The most accurate model is again obtained for the (*solo, business*) pair, followed by the models obtained by combining the *families, friends* and *couples* categories.

Table 10. MAE and RMSE, each pair of categories, ALL hotels.

	Couples		Families		Friends		Solo		Business	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Couples	—	—	0.14	0.19	0.14	0.19	0.15	0.20	0.15	0.19
Families	0.14	0.19	—	—	0.14	0.19	0.18	0.23	0.17	0.23
Friends	0.14	0.18	0.14	0.18	—	—	0.16	0.21	0.16	0.22
Solo	0.15	0.19	0.17	0.22	0.16	0.21	—	—	0.09	0.12
Business	0.15	0.19	0.17	0.22	0.17	0.22	0.10	0.13	—	—
Mean	0.145	0.19	0.155	0.20	0.15	0.20	0.15	0.19	0.14	0.19

6 Application Example

To demonstrate the effectiveness of the methodology, here we report an application, which consists in compensating the scores given by four categories, to obtain a compensated overall score for a target category – different from each of those four. Let the reader suppose to be planning a travel to London with a group of friends. Aiming at obtaining the list of hotels ranked according to their overall score, the best option would be to rank hotels according to the average score given by the *target category*, which is *friends* in our example. However, this

information could not be available for some of the hotels reviewed on the review platform. Thus, we can use the methodology described in Sect. 4.3 to obtain the overall compensated score for these hotels. We apply the following steps:

- From the pre-processed dataset (as explained in the introductory part of Sect. 5), we select the patterns related to all the hotels in London and we split them randomly into training and test set. The patterns of the training set are used to build the prediction models, as described in Sect. 4.2, to predict scores of the target category (*friends*, in our example) with respect to the other categories (Eq. 2).
- For each pattern in the test set, we compute the *compensated overall score* for friends by applying Eq. 3 of Sect. 4.3, thus exploiting the models previously generated. We also compute what we call the *actual overall score*, by combining the actual scores of all the four categories, but the target one. This score is computed only for comparison purposes, i.e. to evaluate the goodness of our approach.
- At this point, for each hotel o , we can consider three possible scores to generate a rank: the actual friends score, computed as $\gamma(o, \textit{friends})$, the actual overall score $S(o)$ and the compensated overall score for friends $\hat{S}(o, \textit{friends})$. The best choice would be to consider the actual friends score. To assess the accuracy of the compensation process, we compute the distance between the compensated overall score for friends and the actual friends score, obtaining the error margin $\bar{\Delta}(o, \textit{friends}) = \hat{S}(o, \textit{friends}) - \gamma(o, \textit{friends})$ and the distance between the actual overall score and the actual friends score, namely $\Delta(o) = S(o) - \gamma(o, \textit{friends})$.
- Finally, we use these errors to compute the MAE and MRSE, to evaluate if the compensated score obtained by applying the prediction model is more accurate than the actual overall score.

We show the results, by varying the target category. We select 60% of patterns for the training set and the remaining 40% of patterns for the test set. The graphs in Fig. 1 show the MAE and the RMSE of the actual and the compensated overall score, for each target category. On average, the compensation obtains better results: the error is lower in four cases out of five. Only when using the *families* category as target, the MAE and RMSE of the compensated overall score are slightly higher than the values of the actual overall score.

We argue that the effectiveness of our methodology could lead to practical applications that would further incentive the adoption of e-advice platforms and improve their usefulness. For example, we can imagine a new feature, offered by the websites, which can make a projection of the scores for any category: the user could have a switch to transform the global ranking - or the evaluation of the very single hotel - to another category, even if there were only few - or no - reviews for that category.

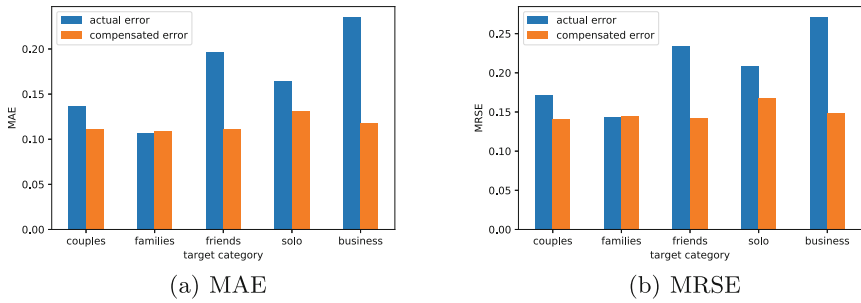


Fig. 1. MAE of actual overall score and compensated overall score for each category, considering hotels in London city (40% of patterns in the test set).

7 Conclusions

This paper considered more than 2.5 million reviews, referring to 1,202 hotels from [Booking.com](#). Relying on regression models, we effectively predicted the numerical average score assigned by a category of users to a hotel, given the average score assigned to the same hotel by another category. The result is useful when there are few - or no - reviews from one category of users, since the prediction can provide the score that would have been given by users of the missing category, by compensating all the scores given by the other categories. The models, which have been proved to be accurate enough for prediction, can be employed to build a compensation system and refine the Booking reputation model, to obtain an overall score adapted to a certain user category.

It would be interesting to apply the same methodology to the scores of single facilities (e.g. cleanliness, comfort, location, etc.) but unfortunately the details of these scores for each user are not available on Booking.

As future work, we will test the effectiveness of the approach on a more general scenario, by extending (i) the categories to diverse reviewers' features and (ii) the target of the reviews, which in our case are limited to hotels, to various products and services, on different e-review platforms.

Acknowledgements. Partially funded by Fondazione Cassa Risparmio Lucca, under the ReviewLand project; and MIUR, under grant “Dipartimenti di eccellenza 2018–2022”, Computer Science Dept., Sapienza University, Rome.

References

1. Allahbakhsh, M.: Robust evaluation of products and reviewers in social rating systems. *World Wide Web* **18**(1), 73–109 (2015)
2. Chen, J., Zhang, C., Niu, Z.: Identifying helpful online reviews with word embedding features. In: Lehner, F., Fteimi, N. (eds.) *KSEM 2016*. LNCS, vol. 9983, pp. 123–133. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47650-6_10

3. Chu, W.T., Huang, W.H.: Cultural difference and visual information on hotel rating prediction. *World Wide Web* **20**(4), 595–619 (2017)
4. Costantino, G., Morisset, C., Petrocchi, M.: Subjective review-based reputation. In: *Symposium on Applied Computing*, pp. 2029–2034. ACM (2012)
5. Diao, Q., et al.: Jointly modeling aspects, ratings and sentiments for movie recommendation. In: *20th ACM KDD*, pp. 193–202 (2014)
6. Freedman, D.: *Statistical Models: Theory and Practice*. Cambridge University Press, Cambridge (2009)
7. Ganu, G., Kakodkar, Y., Marian, A.: Improving the quality of predictions using textual information in online user reviews. *Inf. Syst.* **38**(1), 1–15 (2013)
8. Ghose, A., Ipeirotis, P.G.: Designing novel review ranking systems: predicting the usefulness and impact of reviews. In: *Electronic Commerce*, pp. 303–310 (2007)
9. Hall, M., et al.: The WEKA data mining software: an update. *ACM SIGKDD Explor. Newsl.* **11**(1), 10–18 (2009)
10. Hawkins, D.: The problem of overfitting. *J. Chem. Inf. Comput. Sci.* **44**(1), 1–12 (2004)
11. Hu, Y.H., Chen, K., Lee, P.J.: The effect of user-controllable filters on the prediction of online hotel reviews. *Inf. Manag.* **54**(6), 728–744 (2017)
12. Hu, Y.H., Chen, Y.L., Chou, H.L.: Opinion mining from online hotel reviews: a text summarization approach. *Inf. Process. Manag.* **53**(2), 436–449 (2017)
13. Ngo-Ye, T.L.: Influence of reviewer engagement characteristics on online review helpfulness: a text regression model. *Decis. Support Syst.* **61**, 47–58 (2014)
14. Şensoy, M., Yolum, P.: Automating user reviews using ontologies: an agent-based approach. *World Wide Web* **15**(3), 285–323 (2012)
15. Tang, D., et al.: User modeling with neural network for review rating prediction. In: *24th Artificial Intelligence*, pp. 1340–1346. AAAI Press (2015)
16. Wu, Y., Ester, M.: FLAME: a probabilistic model combining aspect based opinion mining and collaborative filtering. In: *8th ACM Web Search and Data Mining*, pp. 199–208 (2015)
17. Zhang, H.Y.: A novel decision support model for satisfactory restaurants utilizing social information. *Tour. Manag.* **59**, 281–297 (2017)
18. Zhang, R., Tran, T.: An information gain-based approach for recommending useful product reviews. *Knowl. Inf. Syst.* **26**(3), 419–434 (2011)
19. Zhang, R., Tran, T., Mao, Y.: Opinion helpfulness prediction in the presence of “words of few mouths”. *World Wide Web* **15**(2), 117–138 (2012)
20. Zhou, X., Wang, M., Li, D.: From stay to play - a travel planning tool based on crowdsourcing user-generated contents. *Appl. Geogr.* **78**, 1–11 (2017)



Improving SeNA-CNN by Automating Task Recognition

Abel Zacarias^(✉)  and Luís A. Alexandre^(✉) 

Instituto de Telecomunicações, Universidade da Beira Interior,
Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal
{abel.zacarias,luis.alexandre}@ubi.pt

Abstract. Catastrophic forgetting arises when a neural network is not capable of preserving the past learned task when learning a new task. There are already some methods proposed to mitigate this problem in artificial neural networks. In this paper we propose to improve upon our previous state-of-the-art method, SeNA-CNN, such as to enable the automatic recognition in test time of the task to be solved and we experimentally show that it has excellent results. The experiments show the learning of up to 4 different tasks with a single network, without forgetting how to solve previous learned tasks.

Keywords: Supervised learning · Lifelong learning
Catastrophic forgetting · Convolutional neural networks

1 Introduction

Deep learning has emerged as one of the most important tools to deal with the large amount of data available today. One of more widely used methods to process image data are Convolutional Neural Networks (CNNs), that have demonstrated human level ability in many computer vision tasks.

Many practical applications, for instance, in robotics, require the agent to learn new capabilities without forgetting the previous learned ones. To be able to do this using neural networks, one is confronted with the problem of catastrophic forgetting, where the network forgets previous tasks as it learns new ones. So to build lifelong learning systems, there are several proposals (a short review is provided in Sect. 2). In this paper we improve a proposal we made recently [15] where we add layers to an existing CNN such that it can cope with new tasks, without requiring the network to train again on old tasks. This is achieved by selectively adding new layers to an existing model trained on isolated learning.

The improvement we present focus on the automatic selection of the branch of the network that should deal with a new test pattern. This is accomplished

This work was supported by National Founding from the FCT- Fundação para a Ciência e a Tecnologia, through the UID/EEA/50008/2013 Project. The GTX Titan X used in this research was donated by the NVIDIA Corporation.

using a gate neural network. This an interesting solution given that there is biological support for an identical approach: in [8] they showed that this gate mechanism is possible in a pre-frontal cortex of a primate where there are neural representations to select the relevant inputs.

The gate network learns to distinguish between the different learned tasks and therefore, makes the SeNA-CNN a completely automatic solution to the problem of lifelong learning in CNNs. The results presented in the experiments section show that our method is able to deal with the catastrophic forgetting problem and presents better results than a state-of-the-art method [6].

2 Related Work

The problem of catastrophic forgetting is a big issue in machine learning and artificial intelligence if the goal is to build a system that learns through time, and is able to deal with more than a single problem. According to [7], without this capability we will not be able to build truly intelligent systems, we can only create models that solve isolated problems in a specific domain. There are some recent works that tried to overcome this problem, e.g., the Learning without Forgetting (LwF) algorithm proposed in [6] adds nodes to an existing network for a new task only in the fully connected layers and this approach demonstrated to preserve the performance on old tasks without re-training the old tasks after a new one is learned. We compare SeNA-CNN with LwF algorithm. The first main difference is that, instead of adding nodes in fully connected layers, we add convolutional and fully connected layers for the new tasks to an existing model, creating branches for each task; the second one, is that we use a gate to enable automatic recognition in test time. The improved method has a better capability of learning new problems than LwF because we train a series of convolutional and fully connected layers while LwF only trains the added nodes in the fully connected layer and hence, depends on the original task's learned feature extractors to represent the data from all problems to be learned. Apart from that, LwF is not able to automatically recognize the problem to be solved, as the SeNA-CNN improvement we propose in this paper.

A mixture of experts for large-scale image categorization was proposed in [3] which consists of a tree-structured network architecture. The generalist is a CNN optimized on jointly train over all classes and is used to select which expert to process the input data. We also use a CNN as our gate, also with joint train, to help choosing which branch to predict a test image. Differently from our approach, this approach was used for image categorizations only (not applied to lifelong learning), showing that using a tree-structured network architecture can yield a substantial improvement in accuracy over the base CNN trained for example on CIFAR100 and ImageNet.

Our approach, when training the gate, is to make it able to distinguish between the data that is to be processed by each different branch of the SeNA-CNN. To that end, similar to what is done in [3], we cast the definition of the gate as the problem of learning a label mapping $\ell : I \mapsto Z$, where I is the input

image, $Z = \{0, 1, 2, \dots, n - 1\}$ and n is the number of branches, which is the same as the number of tasks. The gate then indicates which branch is to be used for making the classification of a given pattern.

Another difference between the work in [3] and our proposal is that the tree-structured network consists of single trunk feeding all branches, one branch for each expert and the trunk is initialized with convolutional layers of the generalist. We do not follow this approach, we simply use the output of the gate to choose which model to predict at a given time.

3 Proposed Method to Overcome Catastrophic Forgetting

Our proposal is a method that is able to preserve the performance on old tasks while learning new tasks, using only training data from the old tasks for adjusting the gate. Note that other approaches, as [6], are not able to detect automatically to which of the learned tasks the input image belongs to.

A model that is capable of learning two or more tasks has several advantages against that which only learns one task. First advantage is that the previous learned task can help better and faster learning the new task. Second, the model that learns multiple tasks may result in more universal knowledge and it can be used as a key to learn new task domains [13].

The gate network is trained to separate the images from the different tasks automatically. After training the gate, a CNN is created with random initialization of its weights. The network is then trained until convergence for the first task. Figure 1(a) presents the model for the first task trained on isolated learning and Fig. 1(b) is our proposed model with the gate model used to choose automatically a task to deploy at test time. In Fig. 1(b) the blue colour represents the old task network and the orange corresponds to the new added nodes for the new tasks.

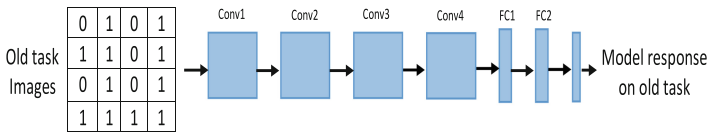
When a new task is going to be learned instead of adding nodes in fully connected layers as is done in [6], we add convolutional, pooling, dropout and fully connected layers for the new task. Typically the added layers contain a structure similar to the network that we trained on isolated learning. We consider the option of not adding the first two layers, because the neurons in those layers find several simple structures, such as oriented edges as demonstrated in [11]. The remaining layers seem to be devoted to more complex objects, and hence, are more specific to each problem, and that is why we choose to create these new layers instead of re-using the original ones. It also resembles the idea of mini-columns in the brain [9]. We add those layers and train them initialized with weights of an old task, keeping the other task layers frozen.

When switching to a third task and fourth task, we freeze the previous learned tasks and only train the new added layers. This process can be generalized to any number of tasks that we wish to learn.

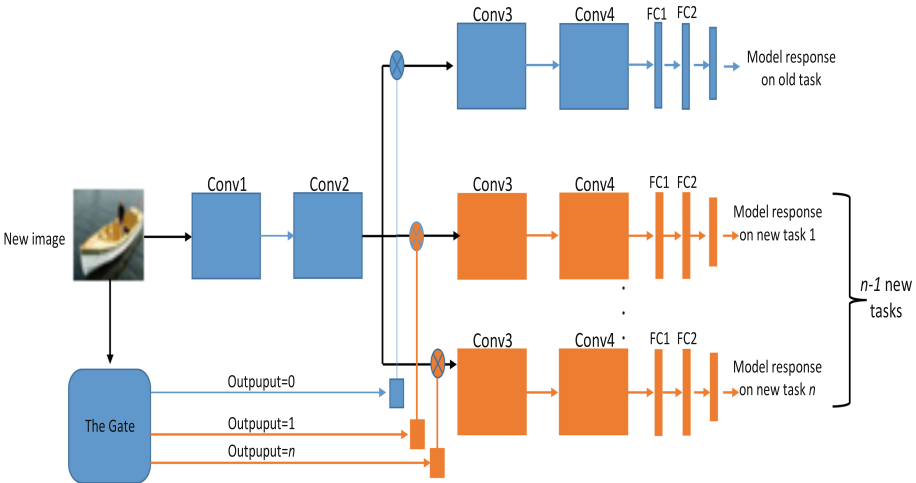
3.1 Using a Gate to Select the Correct Branch

The goal at this stage is to learn a gate neural network with a superclass of the branches that process each task. Our main goal is that the gate represent all tasks to be used at test time to choose automatically which branch to deploy. The gate is trained by joint optimization of images from all tasks. In our approach the gate network has the same architecture as each branch. The only difference is the size of the output that changed each time new task was added to our method. The gate network is initialized randomly and each task is labelled $\{0, 1, \dots, n - 1\}$, where n represents the total number of tasks to be deployed. This way, there is no need for all branches to process the input and produce an output and hence, only one branch is chosen to make the final decision.

Figure 1 shows this process when a new image is received by the proposed model.



(a) Original model for old task trained on isolated learning.



(b) Proposed model: the gate and the added new layers for the $n - 1$ new tasks.

Fig. 1. Original and our proposal used in the experiment process to avoid the catastrophic forgetting by selective network augmentation. The blue coloured boxes correspond to the old task and the orange coloured correspond to the added layers. Adapted from [15]. (Color figure online)

3.2 Training Methodology

The network is going to learn how to solve several tasks. First the gate is trained by using images from each task, with labels that represent those tasks (and not the original task classes). Second, we train a first network branch, starting with randomly initialized weights. Then, for each new task, a new branch is added to the network, where the first two convolutional layers are reused from the first branch.

Figure 1(a) shows the model trained on the first task. After all parameters have been optimized we use the first two convolutional layers of this network to help learn the new tasks as they come. Figure 1(b) illustrates the overall architecture of the proposed method.

During training, we followed the same practice as [6]. The main difference is that when adding a new task we freeze all layers of the original model and only train the added nodes, freezing the first two layers that are common to all branches. During train we use back-propagation with SGD algorithm with dropout enabled. All branches had the same architecture, and the learning rate was set to 0.01, weight decay of $1e - 6$ and momentum 0.9. Table 2 shows the performance of each task trained isolated after 12 training epochs. We run each experiment ten times and present results corresponding to the mean and standard deviation of these 10 repetitions. We run our experiments using a GeForce GTX TITAN X with 12 GiB.

4 Experiments

Our experiments evaluate if the proposed method can effectively avoid the catastrophic forgetting problem and if the gate can be used to automatically choose which task to deploy at test time. The experimental results presented in the following tables, contain the accuracy values of our proposed method. Is important to refer that we begin by training the gate to learn all features that are necessary to distinguish which branch to use for each input image. In all tables we present the accuracy of the gate, which is very important because the final accuracy of our method is calculate taking in count the errors that the gate made during the train and the error that each branch make when the branch is choose to predict the new task. This process was done because during the experiment the gate did not classified correctly all images and if we discards those images, the comparison would not be fair.

We conducted our experiments using four known datasets namely CIFAR10 [2], Caltech101 [5], SVHN2 [10] and Caltech-UCSD Birds-200-2011 [14]. Table 1 shows information on each dataset, and the number of images on training and test sets. CIFAR10 has 10 classes, SVHN2 corresponds to street house numbers and has 11 classes, Caltech101 has 102 classes while Birds has 200 classes. The last one has a substantially lower accuracy than the others, as shown in Table 2. The first two datasets were obtained from `kerosene` that provides six datasets namely binarized-mnist, CIFAR10, CIFAR100, IRIS, MNIST and SVHN2 for machine learning projects in `hdf5` formats. `Kerosene` datasets depend on the

`fuel` library [12] which is a framework based on Python to train neural networks on large datasets.

Table 1. Number of images for train and test sets. In all tables we represent the different datasets using the following scheme: A = CIFAR10, B = Birds, C = Caltech101 and D = SVHN2.

Data set	A	D	C	B
Train	50000	73257	8229	10609
Test	10000	26032	915	1179

4.1 Network Architecture

We used a standard network architecture with an input layer followed by a convolution layer, a ReLU activation function, a convolution layer also followed by a ReLU, maxpooling and a dropout layer. As Fig. 1 shows each new task added to the model has two convolution layers, two ReLU activation function layers and the last layer is the activation that corresponds to the softmax function with categorical cross-entropy loss, two dropout layers, a flatten layer and two dense layers one with 512 units and the other one corresponding to the number of classes for each task.

Input images have 32×32 pixels in all cases. The first convolution layer has filters with 32×32 while the other two convolution layers have filters with 64×64 . We used the keras API [4] running on tensorflow [1].

Table 2. Network performance on isolated learning.

Train	Test	Baseline [%]	Exec.time[s]
A	A	74.80 ± 0.70	144
D	D	93.13 ± 0.69	204
C	C	64.04 ± 1.00	36
B	B	14.54 ± 1.40	48

4.2 Adding New Tasks to the Model

Table 3 presents the performance of the proposed method when adding new tasks and compares it with the baseline [6].

In the presented results, our method clearly outperforms LwF on all tasks showing that adding layers is a good approach to maintain for learning new tasks

Table 3. Gate and two tasks accuracy (and standard deviation) on new tasks for SeNA-CNN and LwF. Execution time for train and test.

Old	New	LwF	Gate	SeNA-CNN	Time[s]
A	D	84.69(0.84)	99.56(0.06)	88.52 (1.50)	798
A	C	63.84(0.34)	99.83(0.12)	67.28 (1.19)	228
A	B	7.92(1.22)	99.99(0.04)	10.48 (1.36)	408
C	A	66.57(2.31)	99.73(0.11)	69.29 (1.52)	306
C	D	84.78(2.31)	99.81(0.12)	90.67 (0.78)	480
C	B	7.73(0.66)	99.56(0.10)	8.54 (1.57)	120
B	A	56.28(1.24)	98.61(2.04)	57.18 (3.94)	396
B	C	57.04 (1.15)	97.94(0.55)	56.81(1.39)	132
B	D	78.61(0.54)	95.72(0.08)	82.76 (1.68)	552
D	A	64.12(0.97)	99.98(0.01)	70.44 (0.78)	804
D	C	53.46(1.97)	99.85(0.02)	57.30 (0.55)	588
D	B	7.94(0.54)	99.91(0.04)	11.36 (1.15)	580

Table 4. Gate and two tasks test accuracy (and standard deviation) on old tasks for SeNA-CNN and LwF. Execution time for train and test.

New	Old	LwF	Gate	SeNA-CNN	Time[s]
A	D	85.41(0.12)	99.56(0.06)	90.08 (1.63)	798
A	C	59.84(1.16)	99.83(0.12)	61.12 (0.34)	228
A	B	11.48(0.63)	99.99(0.04)	13.91 (0.97)	408
C	A	69.45(1.67)	99.73(0.11)	70.75 (0.11)	306
C	D	89.62(0.93)	99.81(0.12)	90.27 (0.54)	480
C	B	12.00(0.64)	99.56(0.10)	13.14 (1.81)	120
B	A	68.22(0.51)	99.61(2.04)	70.70 (1.60)	396
B	C	57.71(0.72)	97.94(0.55)	59.45 (1.03)	132
B	D	91.51 (1.04)	98.93(0.08)	89.12(0.38)	552
D	A	69.36(1.16)	99.98(0.01)	70.50 (0.62)	804
D	C	60.80 (0.92)	99.85(0.02)	60.15(0.39)	588
D	B	11.91(0.41)	99.91(0.04)	13.38 (1.03)	580

while preserving the performance on previous learned tasks. It is interesting to verify that when we use a model trained on CIFAR10 to train a model on Caltech101 the performance for Caltech101 increases compared to isolated learning with a difference of 3.24% and once again it suggest that using one model in a task can help learn even better the other one.

Results show that by applying our method it is possible to overcome the problem of catastrophic forgetting when new tasks are added and the choice of the branch to use when predicting a specific task is done by the gate model.

After learning the new task is important to test if there was no catastrophic forgetting for the task previously learned. This is the main goal of our method: guarantee that the network doesn't forget what was previously learned. To demonstrate that our gate model did not forget what was previously learned we tested again on old tasks. Results are shown on Table 4

4.3 Three Tasks Scenario

To demonstrate that our method is able to deal with different problems, we experiment by learning three different tasks. In this case we used the three datasets previously presented and we combine them two by two for train and one for test. Table 5 presents the results from this scenario and clearly our method is able to maintain the performance and once again it shows that adding new task nodes to an existing neural network is a good option to overcome the catastrophic forgetting problem.

Analysing results in Table 5 its clear that our model is effective in learning new tasks by adding the correspondent branches. In this experiment only in two cases LwF outperformed our proposed model with a slight difference of 1.32% in Birds→SVHN2 and SVHN2→Caltech101 with 0.50% of difference. In the remaining cases our method outperformed LwF.

Overall this situation also occurs in the other experiments when we compare the performance for two and three tasks scenarios.

Table 5. Gate and three tasks test accuracy (and standard deviation) on new tasks for SeNA-CNN and LwF. Execution time for train and test.

Old	New	LwF	Gate	SeNA-CNN	Time[s]
A, D	C	62.90(0.38)	99.14(0.12)	65.88 (0.78)	696
A, C	B	6.84 (0.68)	98.07(1.20)	5.53(0.63)	672
A, C	D	84.48(1.16)	99.69(0.19)	90.76 (0.34)	663
C, D	A	67.72(0.30)	99.52(0.01)	67.24 (0.65)	660
C, A	D	82.26(0.97)	99.67(0.11)	89.55 (0.70)	668
C, D	B	5.16(0.63)	98.99(0.19)	6.45 (1.25)	556
B, C	D	75.86(1.00)	99.37(0.07)	86.21 (1.10)	648
B, A	C	56.24(0.37)	99.62(0.05)	60.50 (0.79)	612
B, D	C	57.12(1.17)	98.38(0.01)	57.43 (0.96)	698
D, A	C	52.86(0.24)	99.71(0.07)	62.34 (0.40)	876
D, C	B	7.47(0.60)	99.63(0.05)	8.80 (1.28)	828
D, B	A	63.83(0.63)	99.48(0.07)	71.55 (0.48)	873

Table 6. Gate and three tasks test accuracy (and standard deviation) on old tasks for SeNA-CNN and LwF. Execution time for train and test.

New	Old	LwF	Gate	SeNA-CNN	Time[s]
A	C, D	52.28(0.79), 87.68(0.47)	99.14(0.12)	56.48 (0.27), 88.30 (0.86)	696
A	B, C	9.93(1.55), 56.29(0.14)	98.07(1.20)	10.72 (0.94), 58.88 (1.01)	672
A	D, B	87.31(1.14), 5.73(0.84)	99.69(0.19)	91.38 (0.87), 9.50 (0.64)	663
C	A, D	70.81(0.19), 86.38(0.67)	99.52(0.01)	70.61 (1.41), 90.33 (0.72)	660
C	B, A	11.73(1.11), 67.89(0.23)	99.67(0.11)	12.36 (0.84), 68.85 (1.46)	668
C	D, A	86.60(0.93), 68.96(0.57)	98.99(0.19)	88.00 (0.58), 70.14 (0.83)	556
B	A, C	66.71(1.32), 55.12(1.06)	99.37(0.07)	71.98 (1.24), 58.78 (1.03)	648
B	C, D	54.72(1.01), 87.54(1.02)	99.62(0.05)	59.47 (0.96), 91.66 (0.59)	612
B	D, C	85.96(0.89), 56.07(0.69)	98.38(0.01)	88.47 (0.67), 59.24 (1.02)	698
D	A, C	64.13(0.76), 56.63(0.95)	99.71(0.07)	70.38 (1.59), 59.73 (0.68)	876
D	C, A	58.03(0.80), 64.99(1.67)	99.63(0.05)	61.60 (1.46), 70.41 (0.98)	828
D	B, C	7.06(0.70), 56.19(0.87)	99.48(0.07)	9.13 (0.48), 61.24 (0.83)	873

Table 6 shows performance after evaluating our method for old tasks when adding two new tasks. Results show that the model did not forget what it learned before.

4.4 Four Tasks Scenario

In this subsection we present results when we add the fourth task. In this experiment we use one task as old and then add sequentially the other three tasks and when learning the new tasks we consider the previous learned tasks as old. Results are shown in the Table 7.

Table 7. Gate and four tasks test accuracy (and standard deviation) on new tasks for SeNA-CNN and LwF. Execution time for train and test.

Old	New	LwF	Gate	SeNA-CNN	Time[s]
A, C, B,	D	83.76(1.07)	99.53(0.13)	90.32 (1.03)	954
C, B, D	A	67.39(1.06)	99.69(0.10)	70.27 (1.02)	936
B, D, A	C	56.48(1.09)	99.70(0.03)	58.98 (1.01)	948
D, A, C	B	7.96(0.98)	99.61(0.05)	10.64 (0.90)	967

Table 8 presents the accuracy of LwF on old. Results in this table are compared with results on Table 9 where we present the performance accuracy on old task of our model. Similarly to the others experiments, here results also show that our model outperformed LwF in combination of all tasks.

Table 8. Four tasks test accuracy (and standard deviation) on old tasks for LwF.

New	Old	LwF
A	C, B, D	51.67(0.98), 7.75(1.07), 82.98(1.03)
C	B, D, A	8.49(1.03), 83.76(1.07), 64.76(1.01)
B	D, A, C	86.44(1.02), 64.95(1.07), 51.18(0.83)
D	A, C, B	68.96(0.98), 54.81(0.97), 7.87(1.02)

Table 9. Gate and four tasks test accuracy (and standard deviation) on old tasks for SeNA-CNN. Execution time for train and test.

New	Old	Gate	SeNA-CNN	Time[s]
A	C, B, D	99.53(0.13)	58.58 (1.03), 10.21 (1.04), 88.00 (1.03)	954
C	B, D, A	99.69(0.10)	9.37 (1.02), 90.01 (1.04), 70.53 (1.03)	936
B	D, A, C	99.70(0.03)	91.08 (0.97), 69.23 (0.83), 60.72 (0.98)	948
D	A, C, B	99.61(0.05)	72.27 (0.93), 59.25 (1.07), 10.27 (1.07)	967

5 Conclusion

In this paper we presented an improvement to our previously proposed method, SeNA-CNN, to avoid the problem of catastrophic forgetting by selective network augmentation, now with automatic task recognition. The proposed method demonstrated to preserve the previous learned tasks without accessing the old task's data, except for updating the gate. It also show that it is feasible to recognize in an automatic manner, which branch should process a given input image, using a gate network. We demonstrated the effectiveness of our method to avoid catastrophic forgetting on image classification tasks by running it on four different datasets and comparing it with the baseline LwF algorithm.

As future work we consider a real application of our method in the field of robotics, for object, location and face recognition problems.

References

1. Abadi, M., et al.: TensorFlow: large-scale machine learning on heterogeneous systems (2015). <https://www.tensorflow.org/>, Software available from tensorflow.org
2. Acemoglu, D., Cao, D., Acemoglu, D., Cao, D.: MIT and CIFAR (2010)
3. Ahmed, K., Baig, M.H., Torresani, L.: Network of experts for large-scale image categorization. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9911, pp. 516–532. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46478-7_32
4. Chollet, F., et al.: Keras (2015)
5. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In: 2004 Conference on Computer Vision and Pattern Recognition Workshop, p. 178, June 2004. <https://doi.org/10.1109/CVPR.2004.109>

6. Li, Z., Hoiem, D.: Learning without forgetting. CoRR abs/1606.09282 (2016). <http://arxiv.org/abs/1606.09282>
7. Liu, B.: Lifelong machine learning: a paradigm for continuous learning. Front. Comput. Sci. 1–3 (2016). <https://doi.org/10.1007/s11704-016-6903-6>
8. Mante, V., Sussillo, D., Shenoy, K.V., Newsome, W.T.: Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature* **503**(7474), 78–84 (2013). <https://doi.org/10.1038/nature12742>. <http://europepmc.org/articles/PMC4121670>
9. Mountcastle, V.B.: Modality and topographic properties of single neurons of cat's somatic sensory cortex. *J. Neurophysiol.* **20**(4), 408–434 (1957). <https://doi.org/10.1152/jn.1957.20.4.408>, pMID: 13439410
10. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning (2011)
11. Rafegas, I., Vanrell, M., Alexandre, L.A.: Understanding trained CNNs by indexing neuron selectivity. CoRR abs/1702.00382 (2017). <http://arxiv.org/abs/1702.00382>
12. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. CoRR abs/1506.01497 (2015). <http://arxiv.org/abs/1506.01497>
13. Shin, H., Lee, J.K., Kim, J., Kim, J.: Continual learning with deep generative replay. CoRR abs/1705.08690 (2017). <http://arxiv.org/abs/1705.08690>
14. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The Caltech-UCSD Birds-200-2011 Dataset. Technical report
15. Zacarias, A., Alexandre, L.A.: SeNA-CNN: overcoming catastrophic forgetting in convolutional neural networks by selective network augmentation. In: Pancioni, L., Schwenker, F., Trentin, E. (eds.) ANNPR 2018. LNCS (LNAI), vol. 11081, pp. 102–112. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99978-4_8



Communication Skills Personal Trainer Based on Viola-Jones Object Detection Algorithm

Álvaro Pardo Pertierra¹, Ana B. Gil González¹(✉),
Javier Teira Lafuente², and Ana de Luis Reboredo¹(✉)

¹ Department of Computing and Automation,
University of Salamanca– Sciences Faculty,
Plaza de la Merced s/n, 37008 Salamanca, Spain
pardopertierraalvaro@gmail.com, {abg, adeluis}@usal.es
² Philosophy Faculty, University of Salamanca, Salamanca, Spain
teira@usal.es

Abstract. The paper presents a personal training system that allows the user to use non-verbal communication techniques that help him improve his way of speaking in public. The tool is a software solution that implements algorithms to identify elements of non-verbal communication (NVC), such as the positions of the head, hands and trunk of the users making use of cascading classifiers. It presents the developed technology and the different characteristics of the application that is an important part of the treatment in artificial vision techniques in the detection of NVC and its training.

Keywords: Oratorical coach · Artificial vision · Viola-Jones algorithm
AdaBoost · Non-verbal communication (NVC) · Learning environment

1 Introduction

Communication is one of the keys to success in the academic, institutional, professional and business fields. Oratory, nonverbal communication techniques and other communication techniques allow you to become a good speaker and learn to express ideas before an audience.

The fourth industrial revolution, far from replacing communication processes, will revalue personal communication and multiply its importance exponentially. The different technological means, each time available to more people (Video conference, VR, augmented reality, internet of things...), facilitate and multiply the occasions of the vis-à-vis communication, particularly necessary when it comes to motivating the decision making, no matter what order: personal, technological, political, economic, etc. Hence, training in oratory and personal communication is now even more relevant and necessary if it fits than it has never been before. In fact, executives and leaders from 48 countries interviewed by The IBM Institute for Business Value in collaboration with Oxford Economics¹ place communication competencies in third place on the list of most demanded, behind computer skills and STEM skills.

¹ <https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=GBE03788USEN>

The paper presents a personal training system that allows the user to practice non-verbal communication techniques that will help them improve their way of speaking in public. The tool is a software solution that implements algorithms capable of recognizing some of the elements of nonverbal communication, such as the head, hands and trunk positions of the users, concluding possible errors of the same. This error detection can be carried out either from a webcam or from a video file stored on disk for which an important treatment in the techniques of artificial vision is made.

The rest of the paper is organized as follows: Sect. 2 includes a short survey of works about automatic recognition of objects in digital images and their main drawbacks, with special focus on Viola-Jones object detection framework. The proposed tool is described in Sect. 3. Finally, the conclusions are given in Sect. 4.

2 Automatic Recognition of Objects in Digital Images

For the detection of the head and trunk of the users of the exercises of the tool, Cascade Classifiers based on Haar type characteristics have been used, which employ the Viola-Jones object detection algorithm. The detection of change or subtraction of background is an important step of preprocessing within the application, being critical for the detection of the hands and its movement as well as the detection of its relation with the different elements in the scene, taken in time real to be treated with artificial vision techniques or by computer.

2.1 The Viola-Jones Object Detection Framework

This is the first object detection framework that provides detection rates of competitive objects in real time. It was proposed in 2001 by Paul Viola and Michael Jones [7]. Although it can be trained to detect a wide variety of objects it was motivated in the first instance by the problem of face detection. The three elements that make Viola-Jones a good detection algorithm are (1) *Robust*: very high detection rate (true positive rate) and positive detection rate always very low (2) *In real time*: for practical applications at least two frames per second must be processed and (3) *detection even-not object recognition* (detection is the first step in the recognition process).

The algorithm is based on a series of weak classifiers called Haar-like features that can be calculated efficiently from an integral image. These classifiers, which by themselves have a probability of matching only slightly higher than chance, are grouped in a cascade using a learning algorithm based on *AdaBoost* to achieve high detection performance as well as a high discriminative capacity in the first. The characteristics sought by the detection algorithm universally involve the sums of the pixels of the image within rectangular areas. The algorithm has four phases:

Feature selection Haar. Haar-like features are the basic elements with which detection is carried out [4]. These features are very simple features or characteristics that are sought in the images and that consist of the difference of light intensities between adjacent rectangular regions. The features are therefore defined by some rectangles and their position relative to the search window and acquire a numerical value resulting from the comparison they evaluate. A Haar feature considers adjacent rectangular

regions at a specific location in a detection window, summarizes the intensities of the pixels in each region and calculates the difference between these sums. This difference is used to classify the subsections of an image. In the detection phase of the Viola-Jones object detection framework, a window of the target size is moved to the input image, and for each sub-section of the image the Haar function is calculated. This difference is then compared to a learned threshold that separates non-objects from objects. Because a Haar characteristic is only a classifier (its detection quality is a bit better than random guessing) a large number of Haar characteristics are necessary to describe an object with sufficient precision. In the Viola-Jones object detection framework, Haar features are organized in a cascading classifier to generate a better classifier. The main advantage of a Haar characteristic is its calculation speed.

The values of the functions then indicate certain characteristics of a particular area of the image. Each type of object can indicate the existence (or absence) of certain characteristics in the image, such as edges or changes in texture. For example, a 2-rectangle feature can indicate where the boundary between a dark region and a luminous region are (Fig. 1).



Fig. 1. Characteristic 2- rectangle (L) and 4- rectangle (R)

Creation of An Integral Image. A representation of the image called integral image evaluates the rectangular characteristics in constant time, this gives it a considerable advantage in terms of speed over more sophisticated alternative features. The integral image at the point (x, y) contains the sum of all the pixels that are above and to the left of that point in the original image. Since the rectangular area of each feature is always adjacent to at least one other rectangle, then any two-rectangle feature can be computed in six references of the matrix, any characteristic of three rectangles in eight, and any characteristic of four rectangles in nine.

Learning Process. Inside a standard 24×24 pixel sub-window, there is a total of $M = 162,336$ possible characteristics, and it would be prohibitively expensive to evaluate them all by testing an image. The object detection algorithm uses a variant of the AdaBoost learning algorithm, a machine-learning adaptive meta-algorithm whose name is an abbreviation of adaptive boosting [2]. AdaBoost is used both to select the best features and to train classifiers that use them. It is necessary to carry out a supervised training process to create the cascade.

Boosting involves taking a series of weak classifiers and combining them to build a strong classifier with the desired precision. AdaBoost was introduced by Freund and Schapire in 1995, solving many of the practical difficulties associated with the boosting

process [14]. This algorithm, Eq. (1) constructs a “strong” classifier as a linear combination of simple “weak” classifiers.

$$h(x) = \text{sgn}\left(\sum_{j=1}^M \alpha_j h_j(x)\right) \quad (1)$$

Each “weak” classifier, as shown in Eq. (2), is a threshold function based on the characteristic \mathbf{f}_j .

$$h_j(x) = \begin{cases} -s_j & \text{if } f_j < \theta_j \\ s_j & \text{otherwise} \end{cases} \quad (2)$$

The threshold value Θ_j and the polarity s_j in the range of ± 1 are determined in training, as well as the coefficients α_j .

Cascade Classification Architecture. Instead of building a single classifier using the process described in the previous section, smaller and more efficient classifiers can be constructed that reject many negative windows (that is, those that do not include any instance of the searched object) maintaining almost all the positive ones (it is say, those that contain an instance of the searched object). These simpler classifiers are used to re-search most of the search windows and only in those where there are more probabilities of finding faces is it called more complex classifiers that decrease the number of false positives [8]. Cascaded classifiers require that training and test data are strongly (and identically) biased toward a small number of easy to detect classes targeted towards classification. Specialization depends on skew (or bias) in the temporal distribution of classes presented to the classifier [6].

2.2 Background Model Subtraction

Background subtraction is mainly used to track and detect moving objects. This is done by finding the difference between the current frame and the previous one and it is also called subtraction of the background model. This model consists of two main steps, the Initialization of the background and the Update of the background. In the first step, an initial model of the background is calculated, while in the second step that model is updated in order to adapt to possible changes in the scene.

OpenCV [1, 5] has implemented three very easy-to-use algorithms of this type. *Back-groundSubtractorMOG*, *BackgroundSubtractorMOG2*, *BackgroundSubtractorGMG*, which are what we will use in the process.

2.3 Color Detection

The detection and segmentation of objects is the most important and challenging task of artificial vision or computer vision. It is a critical part in many applications such as image search, scene understanding, etc. However, it remains an open problem due to the variety and complexity of the object and fund classes. In general, OpenCV [1, 5] captures images and videos in 8 bits, unsigned integer, BGR format. But the HSV color space (HUE, SATURATION and VALUE) is the most appropriate color space for

color-based image segmentation. Then, in most applications, we will have to convert the color space of the original video image from BGR to HSV. In OpenCV, the range of values for HSV is respectively 0–179, 0–255 and 0–255.

3 Algorithm for Non-verbal Communication Error Detection

This algorithm is dedicated to the detection, correction and indication of the extent to which possible non-verbal communication errors are committed in the exercises provided by the application. Each frame of the image flow is subject to processing for positioning and detection of the person and their movements and is subsequently monitored and managed through the interface, as detailed below.

3.1 Positioning and Detection of Speaker Elements

As detailed in the previous sections, a series of cascade classifiers are applied based on Haar-like characteristics on the grey scale image in order to obtain the rectangles of positioning of the faces, right and left profiles, as well as upper body parts of users: head, shoulders, trunk and hands. For this, a background subtraction is carried out initially, due to the low precision and adaptability to the light conditions of the color detection offered by OpenCV to detect the user. This technique requires a photo of the place where the user will give his speech and uses a differentiation of frames between it (background) and the frames of the flow of images that are processed (close-up). Finally, morphological operations like eroding and dilating are applied to improve the results (Fig. 2 (L)). The next step is in a color detection and contour search applied to the result of the previous step. The color detection performs a separation of the HUE, SATURATION and VALUE (HSV) channels color, saturation and brightness of the image to then apply the filters that will obtain the color of the skin to each one of them and that is can be detected, hands and face. Finally, these filters are combined to obtain the output image separating the desired color. Applied to the previous image, the contour search detects the head and the hands (Fig. 2 (R)).

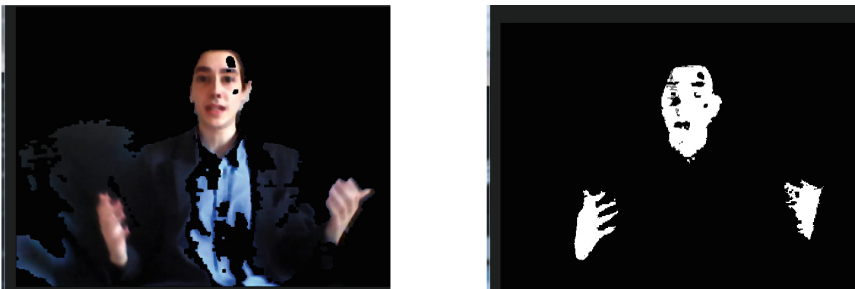


Fig. 2. (L) background subtraction application - (R) detection of head and hands in movement

Instead of using a single strong classifier, a cascade of three gradually more complex classifiers has been used to achieve better detection rates. The cascade has,

therefore, the shape of a degenerate tree. In the case of faces, the first classifier in the cascade - called attentional operator - uses only two characteristics to achieve a false negative rate of approximately 0%. The effect of this unique classifier is to reduce to approximately half the number of times that the entire cascade is evaluated.

OpenCV includes functions for both training a waterfall and detecting objects. Only the functions relating to detection are described here. The algorithm of detection that is implemented in OpenCV is a version of the algorithm of Viola-Jones developed by Lienhart that allows to use features inclined to 45° [3], although the cascade of features that we have used only the original features of Viola-Jones. Another particularity of the method used in OpenCV is that the features are defined on a basic search window of 20×20 pixels instead of 24×24 . Specifically, in this work we have used the cascade “haarcascade_frontalface_alt”.

The result of the application of this technique (Fig. 3) is plotted in rectangles of different colours delimiting the position of its head, face, profiles and hands as well as the upper trunk in case it was needed to detect NVC errors and by monitoring and management.

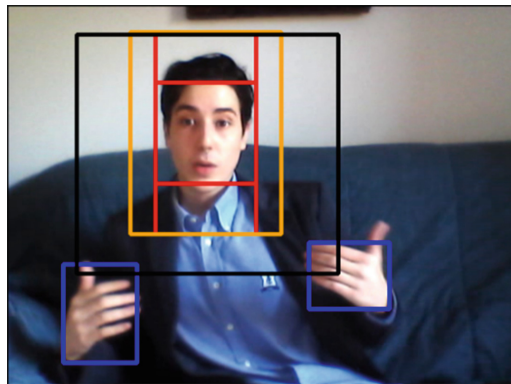


Fig. 3. Detection of elements for the detection of NVC

3.2 Functionalities of the Tool

From the identification of each of the elements through the positioning rectangles, the tool interface for CNV training is constructed. The application corrects, by default and by default, four nonverbal communication errors (1) high hands with respect to posture (2) hands together (3) crooked head and (4) unused hands. Other errors are detected and can be included through the configuration of the session, such as touching the face excessively, scratching the head/face, twisting the body in the wrong percentages, etc. The tool allows you to operate (1) from loading a video or (2) activate the application to work in real time with the help of a camera and the included teleprompter tool.

As an example for detection errors, for twisting body errors the tool check that the position with respect to the *PictureBox* of the body positioning rectangle is sufficiently

displaced to the right or left inside correct parameters. A fragment of the code is shown at Fig. 4. At this way all the CNV errors are detected.

```

if (Properties.Settings.Default.peBodyTwisted == true)
{
    if (bodys.Length != 0)
    {
        imageFrame.Draw(bodys[0], bodysColor, 3);
        if (bodys[0].X < 120 || bodys[0].X > 220)
        {
            System.Windows.Application.Current.Dispatcher.BeginInvoke(
                DispatcherPriority.Background,
                new Action(() => errorBox.Text = "!MANTÉN EL CUERPO RECTO Y CENTRADO!");

            System.Windows.Application.Current.Dispatcher.BeginInvoke(
                DispatcherPriority.Background,
                new Action(() => BodyTwisted.Value = BodyTwisted.Value - Properties.Settings.Default.incrDif));
        }
    }
}
    
```

Fig. 4. Fragment of code for twisting body errors detection

These detected errors are displayed on the right side of the interface by means of progress bars errors filled in three colours that degrade from green to red through yellow to represent the severity of the extent to which we have made each mistake, and errors to train (Fig. 5 (L)). Result of the configurable exercises and/or the training of a speech are scored with qualification and a summary of the mistakes reported (Fig. 5 (R)). This summary allow you to configure the tool to train a series of elements as a priority to improve your learning.

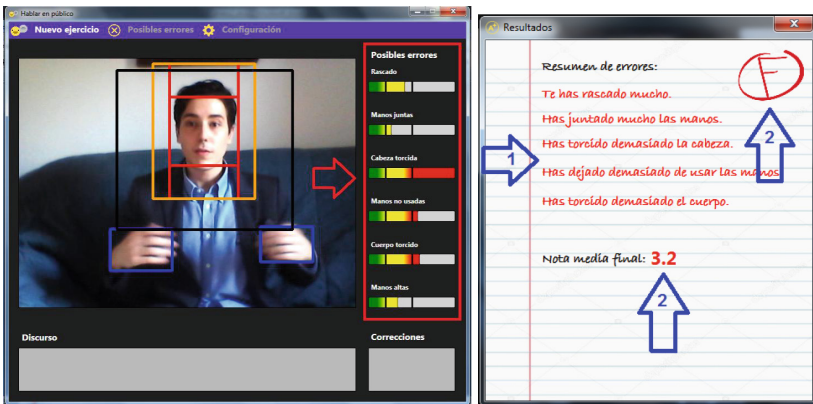


Fig. 5. (L) example of use in the interface during the speech. (R) summary of results with score achieved

4 Conclusions

The main objective of the project is to develop a computer tool (which can be used from a webcam or from a video file stored on disk) that allows focusing the functionality on the use of the OpenCV artificial vision library, to acquire the techniques of non-verbal communication (NVC) while guiding the user in his work of becoming a better speaker. It is a personal training system that allows the user to practice the techniques of NVC that will help him to improve these abilities for speaking in public. This project consists mainly of a software solution that implements making use of an algorithm based on cascading classifiers able to recognize some of the elements of NVC, such as the positions of head, hands and trunk of users concluding possible errors with Image Recognition solutions. The interface facilitates the learning environment through progress bars, an available teleprompter tool, as well as specific activities for each type of element in CNV added. Completed these workouts, also shows a results screen, with a summary of the errors committed.

As future lines of work, we'll add a dimension of verbal communication skills and improve more characteristics of NVC. Several are the current work lines to increase the potential of the tool.

Acknowledgments. This work was carried out under the frame of the “SURF: Arquitectura autoorganizativa de sensores y biometría para el control dinámico de vehículos en ciudades inteligentes Ref. TIN2015-65515-C4-3-R” project. The project was supported and funded by the Spanish Ministerio de Economía, Industria y Competitividad. Retos de investigación, Proyectos I+D+i.

References

1. Bradski, G.: The OpenCV library. Dr. Dobb's J. Softw. Tools (2000). <http://opencv.org/>
2. Freund, Y., Schapire, R.: A short introduction to boosting. *J. Japan. Soc. Artif. Intell.* **14**(5), 771–780 (1999)
3. Lienhart, R., Kuranov, A., Pisarevsky, V.: Empirical analysis of detection cascades of boosted classifiers for rapid object detection. *Pattern Recogn. Lect. Notes Comput. Sci.* **2781**, 297–304 (2003)
4. Mallat, S.G.: A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Trans. Pattern Anal. Mach. Intell.* **11**(7), 674–693 (1989)
5. OpenCV documentation. <http://docs.opencv.org>
6. Shen, H., Han, S., Philipose, M., Krishnamurthy, A.: Fast video classification via adaptive cascading of deep models. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2197–2205. IEEE, July 2017
7. Viola, P., Jones, M.: Robust real-time face detection. In: Proceedings Eighth IEEE International Conference on Computer Vision, ICCV (2001)
8. Xu, Z.E., Kusner, M.J., Weinberger, K.Q., Chen, M., Chapelle, O.: Classifier cascades and trees for minimizing feature evaluation cost. *J. Mach. Learn. Res.* **15**, 2113–2144 (2014)



Optimizing Meta-heuristics for the Time-Dependent TSP Applied to Air Travels

Diogo Duque¹, José Aleixo Cruz¹ , Henrique Lopes Cardoso^{1,2} ,
and Eugénio Oliveira^{1,2} 

¹ Faculdade de Engenharia, Universidade do Porto,
Rua Dr. Roberto Frias, 4200-465 Porto, Portugal
{up201406274,up201403526,hlc,eco}@fe.up.pt

² Laboratório de Inteligência Artificial e Ciências dos Computadores (LIACC),
Porto, Portugal

Abstract. A travel agency has recently proposed the Traveling Salesman Challenge (TSC), a problem consisting of finding the best flights to visit a set of cities with the least cost. Our approach to this challenge consists on using a meta-optimized Ant Colony Optimization (ACO) strategy which, at the end of each iteration, generates a new “ant” by running Simulated Annealing or applying a mutation operator to the best “ant” of the iteration. Results are compared to variations of this algorithm, as well as to other meta-heuristic methods. They show that the developed approach is a better alternative than regular ACO for the time-dependent TSP class of problems, and that applying a K-Opt optimization will usually improve the results.

Keywords: Traveling Salesman Problem · Air travel
Ant Colony Optimization · Meta-optimization

1 Introduction

We have reached an age where air traveling is easy and accessible. With the expansion of air travel comes a substantial increase in the number of available flights. Travel agencies help people find the best and cheapest flights for a journey. Their search engines chew through massive quantities of data to determine the suitable itinerary for a client. If the traveler only needs to go from a city to another, finding the fittest solution is not that hard. But if the journey consists of more than one destination, obtaining the optimal combination of flights becomes more difficult. Kiwi is a Czech online travel agency that has a feature on their website which allows a user to search for a multi-city itinerary. They have launched a Traveling Salesman Challenge (TSC) [13] in 2017 with the purpose of perfecting this feature. That challenge was the motivation for this work, which proposes a combined algorithm with meta-optimization of parameters.

The TSC consists of solving a seemingly simple problem: given a set of cities to visit and a list of flights with the respective origin, destination, price and day, find the combination of flights with the lowest price that visits each city once and returns to the first city. Participants have 30s to find a solution. For higher simplicity, Kiwi defined a couple of restrictions: (a) on each day you have to board exactly 1 flight, and (b) flights are immediate, *i.e.*, they take no time.

By such a description, we can immediately associate this problem with the classic Traveling Salesman Problem (TSP). In TSP, we try to determine the path with the least cost that visits a certain amount of cities only once and returns to the origin city. In graph theory, that is the equivalent of finding the Hamiltonian Cycle with the lowest cost. Nonetheless, the challenge's conversion to a TSP is not as simple as it seems, as the price of a plane ticket does not depend only on the origin and destination, but also on the date of the flight. Also, flights may not even exist in some days. The TSC is, in fact, a Time Dependent TSP.

This problem can be addressed more formally as follows. Let there be n cities to visit. The number of days d to visit all cities is equal to n , because of the first restriction. For each day, there is a different set of available flights and respective prices between the cities. The cost of a trip from city i to j in day t is $c_{i,j}^t$, where $1 \leq t \leq d$. A valid path is one that starts at city A , visits all cities, returns to city A , and only consists of existing flights. Let $x_{i,j}^t$ be a binary variable that has a value of 1 if in day t the flight from i to j was taken, or 0 otherwise. The objective function to minimize is defined as $\sum_{i=1}^n \sum_{j=1}^n \sum_{t=1}^n c_{i,j}^t \cdot x_{i,j}^t$. Just as the regular TSP, the proposed challenge is NP-hard. Thus, solving it for a large number of cities using exact algorithms is not feasible. Most approaches use meta-heuristics to find a “good enough” solution in a timely fashion.

The central objective of this work is to implement a meta-heuristic to find the best solution to Kiwi's Traveling Salesman Challenge [13] in 30s. The rest of the paper is organized as follows. Section 2 reviews related work and frames our work within existing literature. Section 3 describes our approach to the TSC, including optimizations. Section 4 focuses on experimental evaluation. Finally, Sect. 5 concludes the paper and points directions for future work.

2 Literature Review

In the 19th century, the problem of finding a Hamiltonian cycle on a graph was mathematically formulated by Hamilton [1]. The Hamiltonian cycle is a well-known reduction of the TSP. Some of the first techniques to be used to solve the TSP include Branch and Bound [14], the Christofides algorithm [5] and the 2-Opt [16].

Gambardella and Dorigo, who wrote the original paper on the Ant Colony Optimization (ACO) meta-heuristic, combined ACO and Q-Learning to calculate the optimal solution for a TSP instance [9]. Results indicated that this procedure was better than some other popular approaches, such as Simulated Annealing. ACO has been compared favorably to other heuristic approaches [18, 19]. More recently, Mohsen [17] used ACO with a particular operator that combines Genetic Algorithms (GA) and Simulated Annealing (SA) to solve the TSP.

The operator was used to control ant diversity, a technique that has shown improved performance when compared to a standard ACO implementation.

As meta-heuristics generally involve setting a few parameters that affect their performance, it is necessary to tune them for a particular problem. However, there is no rule for the tuning. *Meta-optimization* consists of using an optimization method to tune another optimization method. It has been used to improve, for example, the performance of genetic algorithms [8].

Inspired by other works, instead of creating a new meta-heuristic, we adopted Mohsen's mixed approach [17] to solve the TSC. We also perform meta-optimizations using genetic algorithms [8] and local search by applying K-Opt moves [11]. With this work, we intend to improve Mohsen's meta-heuristic performance and develop an even faster algorithm for the TSP and for Kiwi's TSC.

3 Methodological Approach

A TSP is commonly represented using graph theory. In an air travel context, each node represents a city and each edge a flight. The final solution is a set of ordered flights, expressing the trips to do. In the case of the TSC, however, given the time-dependent nature of trip prices, we need to add a restriction stating that, on a given day, only available flights can be used to travel between cities. We do this by representing a city with different nodes for each day.

We can also formalize the TSP as a scheduling problem [2], by using the $\alpha|\beta|\gamma$ notation [10]: $1|s_{ij}|C_{max}$. There is a single machine, the traveler. A job is a city to visit: job j is characterized by its processing time p_j and by a setup time s_{ij} that represents the amount of time needed to setup job j after job i . The setup time of a job depends on the job that precedes it. The processing time is the cost of remaining in the city. Since it is not relevant to this particular problem, every processing time is equal to a constant that we disregard. The setup time is the cost of going from city i to city j . The objective function is to reduce the overall make-span C_{max} , that is, the total cost of executing every job (which in our case corresponds to the sum of flight prices).

Although this formalization is correct for a regular TSP, it does not fit the TSC because of time-dependencies: going from city i to city j on day x may have a different price than doing so on day y . Thus, the setup time s_{ij} is dependent on the slot t in which the machine executes the job. This t variable corresponds to the day of flight in the original problem proposed by the Kiwi travel agency. A more appropriate formulation is: $1|s_{ij}^t|C_{max}$. Instead of a single cost matrix, there are t matrices, one for each day.

3.1 Algorithms

Meta-heuristics can be applied to solving all kinds of optimization problems. We have adopted Simulated Annealing and Ant Colony Optimization as our chosen meta-heuristics. For comparison, we also implemented Greedy search (Basic Hill Climbing) and Backtracking as a deterministic algorithm.

Simulated Annealing. Many algorithms are greedy and tend to get stuck in local optima (such as the well-known hill-climbing algorithm). Simulated Annealing (SA) is a well-known probabilistic technique for finding the global optimum. SA comprises a local-search optimization procedure and includes a temperature parameter: the higher it is, the higher the chance of accepting a worse neighboring state for the next iteration of the algorithm. After each iteration, temperature decreases. SA will start to look for solutions in a vast space, but with time it will search in an increasingly small space until the temperature gets close to 0.

Ant Colony Optimization. Ant Colony Optimization (ACO) is a rather common approach to solving the TSP [6, 15], based on the concept of collective intelligence of an ant colony. Every ant tries to find the best path to the food, and the better the path, the more pheromones will be laid on it. These make certain trails more “attractive”, causing a convergence to better paths. While ACO can be seen as a probabilistic approach, it explores the search space using a set of individuals. It has some similarities with genetic algorithms, but while the latter combines individuals through crossover and mutation operators, ACO explores synergies between individuals through environment-based communication.

Backtracking. Backtracking [1] is a general deterministic algorithm whose strength is that it does not need to explore the entire search space. Whenever Backtracking finds itself in a branch that does not lead to an improvement on the current best solution, it moves on to the different branch.

3.2 Optimizations

Discovering new valuable meta-heuristics is very difficult. For this work, we focus on optimizing a meta-heuristic’s behavior in order to improve its performance.

Parallelized Ant Colony. Parallelization is often a good solution to speed execution time, if the program has parts that can be parallelized, that is, executed independently. In ACO, the ant’s path generation (line 8 of Algorithm 1) can be parallelized, as each ant’s path is generated on its own, needing only the graph and the pheromone trails. Pheromone updates occur on a single thread. The computer we ran the simulation on had a dual-core CPU with hyper-threading; as such, we defined a thread pool of 4, so as to match the maximum number of threads available on the CPU. A larger thread pool would mean that one or more *CPU threads* would need to handle the extra *software thread*, thus limiting the total speed of the algorithm.

Ant Colony with Simulated Annealing. Many so-called hybrid metaheuristics [4] have been proposed in the past, and the optimization described here can be seen as one such attempt. This approach [17] is a variation of the original ACO, with a twist at the end of each iteration: if the population *diversity* is

Algorithm 1: Ant Colony Optimization with Simulated Annealing

```

1 function antColonySA (g, n, ph_w, vis_w, ev_f, q, ini_ph_lvl, temp_dec,
  iter_per_temp);
  Input : graph g, number of ants  $n = 30$ , pheromone weight  $ph_w = 0.5$ ,
    visibility weight  $vis_w = 0.5$ , evaporation factor  $ev_f = 0.8$ ,  $q = 1000$ ,
    initial pheromone level  $ini\_ph\_lvl = 20$ , temperature decrease  $temp\_dec = 0.05$ ,
    initial temperature  $init\_temp = 1$  and number of iterations
    per temperature  $iter\_per\_temp = 1$ 
  Output: the path with the least cost obtained
2 startTimer();
3 best_cost  $\leftarrow +\infty$ ;
4 best_path  $\leftarrow null$ ;
5 initializePheromones(ini_ph_lvl);
6 while not timerFinished() do
7   createAnts(b);
8   moveAnts(g, ph_w, vis_w);
9   new_paths  $\leftarrow$  getAntsPaths();
10  updatePathPheromones(new_paths, ev_f, q);
11  diversity = (averageCost - bestCost) / (worstCost - bestCost);
12  if diversity > 0.7 then
13    | simulatedAnnealing(getBestAnt(), temp_dec, iter_per_temp, init_temp);
14  end
15  else
16    | mutate(getBestAnt());
17  end
18  updatePathPheromones(getBestAnt().getPath(), ev_f, q);
19  if getBestAnt().getCost() < best_cost then
20    | best_cost  $\leftarrow$  getBestAnt().getCost();
21    | best_path  $\leftarrow$  getBestAnt().getPath();
22  end
23 end
24 return best_path;

```

high, simulated annealing is applied to the best-performing ant and then its pheromones are applied to the paths as usual; if the population diversity is low, a mutation is performed on that ant (swapping the order of two random adjacent cities) and the pheromones are also applied. By intensifying the pheromone trail of a globally good solution through the application of SA to the best ant, attention is concentrated on that search area, reducing dispersion. The mutation operation brings a reinforcement of random paths, increasing the level of exploration by opening up the search space. Algorithm 1 shows our implementation.

Meta-optimizing Using Genetic Algorithms. The amount of adjustable parameters resulting from the combination of ACO and SA is too high to manually test all possible configurations. Optimizing these values is, in itself, a new problem that we tackle using a genetic algorithm. Chromosomes have one gene

Algorithm 2: Meta-optimization using Genetic Algorithms

```

1 function metaOptimization ( $g, n$ );
   Input : the graph  $g$  and the number of generations  $n$ 
   Output: the meta-optimized set of parameters
2 randomly initialize population  $p$ ;
3 for  $i = 1$  to  $n$  do
4   while processing time < maximum processing time do
5     for each individual  $i$  in population  $p$  do
6       |  $fitness(i) = antColonySA(i.getParameters());$ 
7     end
8   end
9   generate new population based on fitness values;
10 end
11 return best individual's parameters;

```

for each parameter. The fitness function indicates the performance of the ACO-SA algorithm when using the values of the chromosome. We use the average result of ten runs for the fitness function. The lower the average cost obtained, the greater the value of fitness. The procedure we use for this genetic algorithm is listed in Algorithm 2. Running meta-optimization is costly, but it only needs to be performed once. Since Kiwi's challenge is mostly about getting the best result in a short period of time, by already having good parameters we are increasing our chances of finding the optimal solution.

Solution Optimization with K-Opt. K-Opt [11] is a local search algorithm involving swap moves over a previously-obtained path. K is the number of edges to delete, creating $K + 1$ sub-tours. Apart from the first and last sub-tour, all others are reordered and/or reversed to try to find a better global tour. The algorithm attempts all possible combinations of removed edges (that still preserve $K+1$ sub-tours) to find the best combination of reordering and/or reversing. We used $K \in \{2, 3\}$ since, even though the algorithm runs in polynomial time $T(n^K)$, the running time for higher K in heavier datasets is too high. 3-Opt usually gets within 3–4% of an optimal tour [11], so it ensures an already rather good proximity to the optimal solution.

4 Experimental Evaluation

In order to assess the merits of each optimization approach (described in Sect. 3.2) in enhancing the performance of the heuristics, we compare the results obtained with and without such optimizations. The dataset used was the one provided by Kiwi for the challenge [12], which is a comma-separated values (CSV) file. The first line contains the string corresponding to the origin city. The following lines contain the available flights, according to the syntax *origin.city, destination.city, day, price*.

We tested the heuristics for several dataset sizes of up to 100 cities. Results shown in Table 1 are all relative to one single documented run. However, in most of the undocumented runs we executed, the scenario was the same. According to Kiwi’s challenge, the output route of an execution is the best route found in 30 s. After this time, the execution of the algorithm terminates.

Table 1. Results of the algorithms in one run with 15, 60 or 100 cities.

Algorithm	15 cities		60 cities		100 cities	
	Lowest cost	Iterations	Lowest cost	Iterations	Lowest cost	Iterations
Backtrack	5268	–	29387	–	53202	–
Greedy	4801	15	12073	60	17893	100
SA	4344	5596770	15450	319760	20116	66810
ACO	4464	19218	10396	652	17002	150
ACO (p)	4500	23623	10340	1213	16614	237
ACO-SA	4385	22189	10213	696	16156	165
ACO-SA (p)	4385	20740	10551	630	16452	145
ACO-SA (m)	4385	15538	10007	665	16206	150
ACO-SA (p) (m)	4385	16660	10374	670	16037	162

We start by comparing the performance of a regular version of Ant Colony Optimization with other heuristics, such as Simulated Annealing and Hill Climbing (Greedy), but also exact algorithms like Backtrack. These algorithms gave us a control group to compare the performance of our approach. Figure 1 demonstrates how the ACO variations behave for different dataset sizes.

For datasets with a number of cities smaller than 15, the results achieved by brute-force and heuristic algorithms were identical. Table 1 documents the differences that arise as the search space grows larger. As expected, with the increase in the number of cities, Backtrack and Hill Climbing struggled to find satisfactory solutions in the 30 s time window (in fact, the running time needed for Backtracking to run with 15 cities is of 7 m 16 s, compared to 155 ms for 10 cities and 1 ms for 5 cities). Ant Colony Optimization proved better than Simulated Annealing in all sizes. The parallelization of the Ant Colony Optimization proved valuable in increasing the number of iterations the algorithm could perform before running out of time, although it did not bring a much better result, leading us to conclude that convergence was already achieved most of the times.

The Simulated Annealing tweak makes an iteration take longer to execute. Because of the 30 s time limit, ACO-SA makes less iterations than the regular ACO. However, ACO-SA consistently leads to a better solution than ACO, as seen in Fig. 1. The control of the ants diversity proved to have a great effect in the speed of convergence. ACO-SA found better intermediate solutions than ACO faster, so its performance is effectively better than ACO.

Parameter optimization through genetic algorithms was done by targeting a dataset with 60 cities. For datasets with this approximate size, meta-optimization did improve the results of the algorithm. In datasets of

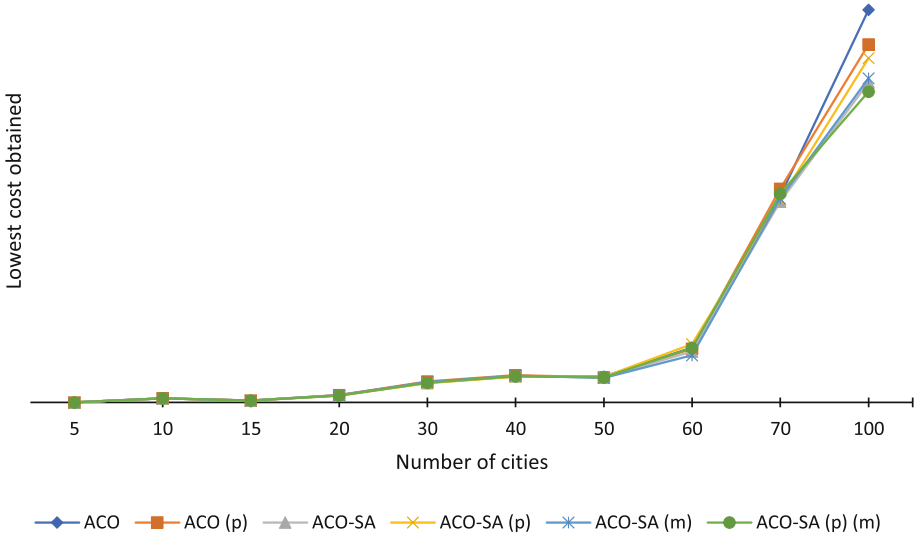


Fig. 1. Lowest cost obtained for different dataset sizes. (p) means that the algorithm has been parallelized. (m) means that meta-optimization was applied. The values have been raised to the power of 4 to better demonstrate how algorithms behave as the search space gets larger.

significantly different sizes, however, it did not behave as well as we thought, being worse than the manually defined parameters. We believe that by redefining the meta-optimization strategy to take into account datasets of different sizes the results would have been better.

Applying K-Opt on the final solution allows us to find even better solutions. The actual value of using the K-Opt technique is better for a number of cities between 10 and 80, as can be observed by crossing information in Fig. 2. In smaller datasets, the difference K-Opt makes is not so noticeable because there is not much room for improvement. In larger datasets, and given the fact that the time complexity of a K-Opt execution for n cities is n^K , 3-Opt reveals itself as inappropriate due to the time it takes to run. Thus, we can conclude that the usage of the K-Opt technique should be targeted mainly to medium-sized datasets, and only 2-Opt should be used in larger datasets so as to avoid a significant increase in execution time.

K-Opt’s running time does not count to the 30s total of the challenge. As their runtime is constant (for a constant dataset size), we assume that anyone planning on using it could reduce the main algorithm’s running time by a calculated margin that would allow for the optimization to be executed. As such, when comparing our main algorithms in all datasets, we chose to keep the 30s total so as to not affect the comparison between every result.

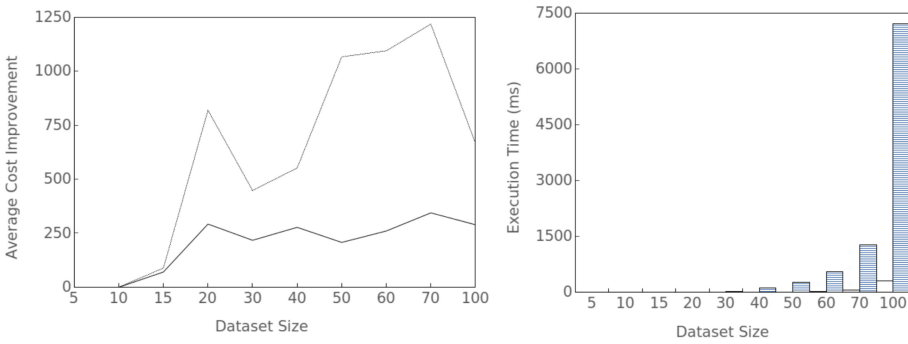


Fig. 2. Average cost improvement by applying K-Opt (left) and execution time of 2-Opt and 3-Opt on the final solution of ACO-SA (right).

5 Conclusions

Combining Ant Colony Optimization with Simulated Annealing proved to have better performance than the regular ACO version to solve Kiwi’s multi-city air travel challenge. Applying the K-Opt technique to an already good solution is useful mostly for medium sized datasets, because of the trade-offs between solution improvement and computation time.

Our experiments in applying meta-optimization to the ACO-SA did not improve the results much, but we believe that by slightly altering our approach we could achieve better results. Since the optimization was made using a fixed dataset size, it was not so effective on the other sizes, which means that if we had used datasets with different sizes, the result might have been better. We also found that Backtracking could be useful in a real-life context (which uses few cities), but clearly is not enough in the TSC.

In the future, an improvement that could be made would be to use racing conditions in the meta-optimization [3]. Instead of spending resources evaluating every single generated individual, we would drop the least promising candidates during the evaluation process, focusing on the best individuals. Another improvement could be the parallelization of 3-Opt so that it becomes faster to execute and thus less cumbersome to use with larger datasets. The code used for the project is hosted in a GitHub repository [7].

References

1. Biggs, N., Lloyd, E.K., Wilson, R.J.: Graph Theory, 1736–1936. Clarendon Press, New York (1986)
2. Bigras, L.-P., Gamache, M., Savard, G.: The time-dependent traveling salesman problem and single machine scheduling problems with sequence dependent setup times. *Discrete Optim.* **5**(4), 685–699 (2008)
3. Birattari, M.: The problem of tuning metaheuristics: as seen from the machine learning perspective. Master’s thesis, Université libre de Bruxelles (2005)

4. Blum, C., Aguilera, M.J.B., Roli, A., Sampels, M.: Hybrid Metaheuristics: An Emerging Approach to Optimization, 1st edn. Springer, Heidelberg (2008). <https://doi.org/10.1007/978-3-540-78295-7>
5. Christofides, N.: Worst-case analysis of a new heuristic for the traveling salesman problem, p. 10, February 1976
6. Dorigo, M., Di Caro, G.: Ant colony optimization: a new meta-heuristic. In: Proceedings of the Congress on Evolutionary Computation, pp. 1470–1477. IEEE Press (1999)
7. Duque, D., Cruz, J.A.: (2018). <https://github.com/jazzchipc/MPE-FEUP>
8. Fernandez-Prieto, J.A., Canada-Bago, J., Gadeo-Martos, M.A., Velasco, J.R.: Optimisation of control parameters for genetic algorithms to test computer networks under realistic traffic loads. *Appl. Soft Comput.* **12**(7), 1875–1883 (2012)
9. Gambardella, L.M., Dorigo, M.: Ant-Q: a reinforcement learning approach to the traveling salesman problem. In: Proceedings of the 12th International Conference on Machine Learning, ICML1995, pp. 252–260. Morgan Kaufmann Publishers Inc. (1995)
10. Graham, R.L., Lawler, E.L., Lenstra, J.K., Kan, A.H.G.R.: Optimization and approximation in deterministic sequencing and scheduling: a survey. In: Hammer, P.L., Johnson, E.L., Korte, B.H. (eds.) *Discrete Optimization II*, volume 5 of *Annals of Discrete Mathematics*, pp. 287–326. Elsevier (1979)
11. Johnson, D., McGeoch, L.A.: The traveling salesman problem: a case study in local optimization. In: Aarts, E.H.L., Lenstra, J.K. (eds.) *Local Search in Combinatorial Optimization*. Wiley (1997)
12. Kiwi.com: Rules for kiwi.com travelling salesman competition (2018). <https://github.com/kiwicom/travelling-salesman>. Accessed 29 June 2018
13. Kiwi.com: Travelling salesman challenge (2018). <https://travellingsalesman.cz>. Accessed 29 June 2018
14. Kolesar, P.J.: A branch and bound algorithm for the knapsack problem. *Manage. Sci.* **13**(9), 723–735 (1967)
15. Li, B., Wang, L., Song, W.: Ant colony optimization for the traveling salesman problem based on ants with memory. In: 2008 Fourth International Conference on Natural Computation (ICNC), pp. 496–501 (2008)
16. Lin, S.: Computer solutions of the traveling salesman problem. *Bell Syst. Tech. J.* **44**(10), 2245–2269 (1965)
17. Mohsen, A.M.: Annealing ant colony optimization with mutation operator for solving TSP. *Comput. Intell. Neurosci.* **2016**, 13 (2016)
18. Mukhairez, H., Maghari, A.: Performance comparison of simulated annealing, GA and ACO applied to TSP. *Int. J. Intell. Comput. Res.* **6**(4), 647–654 (2015)
19. Selvi, V., Umarani, R.: Comparative analysis of ant colony and particle swarm optimization techniques. *Int. J. Comput. Appl.* **5**(4), 1–6 (2010)



Compositional Stochastic Average Gradient for Machine Learning and Related Applications

Tsung-Yu Hsieh^{1,2}(✉), Yasser EL-Manzalawy^{1,3}, Yiwei Sun^{1,2},
and Vasant Honavar^{1,2,3}

¹ Artificial Intelligence Research Laboratory, The Pennsylvania State University,
University Park, PA 16802, USA

{tuh45, yme2, yus162, vuh14}@psu.edu

² Department of Computer Science and Engineering,
The Pennsylvania State University, University Park, PA 16802, USA

³ College of Information Science and Technology,
The Pennsylvania State University, University Park, PA 16802, USA

Abstract. Many machine learning, and statistical inference problems require minimization of a composition of expected value functions (CEVF). Of particular interest is the finite-sum versions of such compositional optimization problems (FS-CEVF). Compositional stochastic variance reduced gradient (C-SVRG) methods that combine stochastic compositional gradient descent (SCGD) and stochastic variance reduced gradient descent (SVRG) methods are the state-of-the-art methods for FS-CEVF problems. We introduce compositional stochastic average gradient descent (C-SAG) a novel extension of the stochastic average gradient method (SAG) to minimize composition of finite-sum functions. C-SAG, like SAG, estimates gradient by incorporating memory of previous gradient information. We present theoretical analyses of C-SAG which show that C-SAG, like C-SVRG, achieves a linear convergence rate for strongly convex objective function; However, C-CAG achieves lower oracle query complexity per iteration than C-SVRG. Finally, we present results of experiments showing that C-SAG converges substantially faster than full gradient (FG), as well as C-SVRG.

Keywords: Machine learning · Stochastic gradient descent
Compositional finite-sum optimization · Convex optimization

1 Introduction

Many problems in machine learning and statistical inference (e.g., [8]), risk management (e.g., [10]), multi-stage stochastic programming (e.g., [32]) and adaptive simulation (e.g., [14]) require minimization of linear or non-linear compositions of expected value functions (CEVF) [37]. Of particular interest are finite-sum versions of the CEVF optimization problems (FS-CEVF for short) which find

applications in estimating sparse additive models (SpAM) [28], maximizing softmax likelihood functions [12] (with a wide range of applications in supervised learning [2, 13]), portfolio optimization [23], and policy evaluation in reinforcement learning [33].

Stochastic gradient descent (SGD) methods [19, 29], fast first-order methods for optimization of differentiable convex functions, and their variants, have long found applications in machine learning and related areas [1, 3–5, 7, 9, 20–22, 34, 41]. SGD methods continue to be an area of active research focused on methods for their parallelization [16, 41–43], and theoretical guarantees [17, 26, 27, 31].

The linearity of the objective function in the sampling probabilities is crucial for establishing the attractive properties of SGD since SGD update the parameter iteratively by using *unbiased samples* (queries) of the gradient. However, since the CEVF objective functions violate the required linearity property, classical SGD methods no longer suffice [37]. Against this background, Wang et al. [37], introduced a class of stochastic compositional gradient descent (SCGD) algorithms for solving CEVF optimization problems. SCGD, a compositional version of stochastic quasi-gradient methods [11], has been proven to achieve convergence rate that is sub-linear in K , the number of stochastic samples (i.e. iterations) [36–38]. Lian et al. [23] proposed compositional variance-reduced gradient methods (C-SVRG) for FS-CEVF optimization problems. C-SVRG methods combine SCGD methods for CEVF optimization [37] with stochastic variance-reduced gradient (SVRG) method [18] to achieve a convergence rate that is constant linear in K , the number of stochastic samples, when the FS-CEVF to be optimized is strongly convex [23].

Recently, several extensions and variants of the C-SVRG method have been proposed. Liu et al. [25] investigated compositional variants of SVRG methods that can handle non-convex compositions of inner and outer finite-sum functions. Yu et al. [39] incorporated the alternating direction method of multipliers (ADMM) method [6] into C-SVRG to obtain com-SVR-ADMM to accommodate compositional objective functions with linear constraints. They also showed that com-SVR-ADMM has a linear convergence rate for strongly convex and Lipschitz smooth objectives. Others [15, 24] incorporated proximal gradient descent techniques into C-SVRG to handle objective functions with a non-smooth convex regularization penalty. Despite the different motivations behind the various extensions of C-SVRG and their mathematical formulations, the query complexity per iteration of gradient-based update remains unchanged from that of C-SVRG. Specifically, C-SVRG methods make two oracle queries for every targeted component function to estimate gradient per update iteration.

In light of the preceding observations, it is tempting to consider compositional extensions of stochastic average gradient (SAG) [30] which offers an attractive alternative to SVRG. SAG achieves $O(1/K)$ convergence rate when the objective function is convex and a linear convergence rate when the objective function is strongly-convex [30]. While in the general setting, the memory requirement of SAG is $O(np)$ where n is the number of data samples and p is the dimensionality of the space, in many cases, it can be reduced to $O(n)$ by exploiting

problem structure [30]. We introduce the compositional stochastic average gradient (C-SAG) method to minimize composition of finite-sum functions. Like SAG, C-SAG estimates gradient by incorporating memory of previous gradient information. We present theoretical analyses of C-SAG which show that C-SAG achieves a linear convergence rate when the objective function is strongly convex. However, C-SAG achieves lower oracle query complexity per iteration than C-SVRG. We present results of experiments showing that C-SAG converges substantially faster than full gradient (FG), as well as C-SVRG.

The rest of the paper is organized as follows. Section 2 introduces the FS-CEVF problem with illustrative examples. Section 3 describes the proposed C-SAG algorithm. Section 4 establishes the convergence rate of C-SAG. Section 5 presents results of experiments. Section 6 concludes with a summary and discussion.

2 Finite-Sum Composition of Expected Values Optimization

In this section, we introduce some of the key definitions and illustrate how some machine learning problems naturally lead to FS-CEVF optimization problems.

2.1 Problem Formulation

A **finite-sum composition of expected value function** (FS-CEVF) optimization problem [23] takes the form:

$$\min_x f(x) := F \circ G(x) = F(G(x)), \quad G(x) = \frac{1}{m} \sum_{j=1}^m G_j(x), \quad F(y) = \frac{1}{n} \sum_{i=1}^n F_i(y), \quad (1)$$

where the inner function, $G : \mathbb{R}^p \rightarrow \mathbb{R}^q$, is the empirical mean of component functions, G_j , and the outer function, $F : \mathbb{R}^q \rightarrow \mathbb{R}$, is the empirical mean of component functions, F_i . Throughout the following content, we denote by $z_{(i)}$ the i -th component of vector z .

2.2 Example: Statistical Learning

Estimation of sparse additive models (SpAM) [28] is an important statistical learning problem. Suppose we are given d -dimensional input vectors $x_i = (x_{i1}, x_{i2}, \dots, x_{id})^T \in \mathbb{R}^d$, and associated responses y_i . Suppose that $y_i = \sum_{j=1}^d h_j(x_{ij}) + \epsilon_i$ for each pair (x_i, y_i) . Here, $h_j : \mathbb{R} \rightarrow \mathbb{R}$ is a feature extractor, which can in general be non-linear functions, and ϵ_i is zero-mean noise. SpAM estimates the feature extractor functions by solving the following minimization problem:

$$\min_{h_j \in \mathbb{H}_j, j=1, \dots, d} \frac{1}{n} \sum_{i=1}^n \left(y_i - \sum_{j=1}^d h_j(x_{ij}) \right)^2,$$

where \mathbb{H}_j is a pre-defined set of functions chosen to ensure that the model is identifiable. In many practical applications of machine learning, h_j 's are assumed to be linear, and hence the model is simplified to $y_i = w^T x_i + \epsilon_i$, where w is a coefficient vector. Penalized variants of such models (e.g. [35], [40]) are often preferred to enhance stability and to induce sparse solutions. Of particular interest is the l_1 -penalized regression, or known by the LASSO estimator [35], which can be written as follows:

$$w^* = \arg \min_w \frac{1}{n} \sum_{i=1}^n (y_i - w^T x_i)^2 + \lambda \|w\|_1.$$

This objective function can be formulated as an FS-CEVF problem in Eq. (1):

$$\begin{aligned} G_j(w) &= (\lambda |w_{(j)}|, w_{(j)} \cdot x_{ij})^T \in \mathbb{R}^2 \\ z &= \frac{1}{d} \sum_{j=1}^d G_j(w) = \left(\frac{\lambda}{d} \|w\|_1, \frac{1}{d} w^T x_i \right)^T \in \mathbb{R}^2 \\ F_i(z) &= (y_i - d \cdot z_{(2)})^2 + d \cdot z_{(1)} \\ \Rightarrow f &:= \frac{1}{n} \sum_{i=1}^n F_i \left(\frac{1}{d} \sum_{j=1}^d G_j(w) \right) = \frac{1}{n} \sum_{i=1}^n (y_i - w^T x_i)^2 + \lambda \|w\|_1. \end{aligned}$$

2.3 Example: Reinforcement Learning

Policy evaluation in reinforcement learning [33] presents an instance of FS-CEVF optimization. Let S be a set of states, A a set of actions, $\pi: S \rightarrow A$ a policy that maps states into actions, and $r: S \times A \rightarrow \mathbb{R}$ a reward function, Bellman equation for the value of executing policy π starting in some state $s \in S$ is given by $V^\pi(s) = \mathbb{E}_\pi \{r_{s,s'} + \gamma V^\pi(s') | s\}$, $\forall s, s' \in S$, where $r_{s,s'}$ denotes the reward received upon transitioning from s to s' , and $0 \leq \gamma < 1$ is a discounting factor. The goal of reinforcement learning is to find an optimal policy π^* that maximizes $V^\pi(s) \forall s \in S$.

The Bellman equation becomes intractable for moderately large S . Hence, in practice it is often useful to approximate $V^\pi(s)$ by a suitable parameterized function. For example, $V^\pi(s) \approx \phi_s^T w$ for some $w \in \mathbb{R}^d$, where $\phi_s \in \mathbb{R}^d$ is a d -dimensional compact representation of s . In this case, reinforcement learning reduces to finding an optimal w^* :

$$w^* = \arg \min_w \frac{1}{|S|} \sum_s (\phi_s^T w - q_{\pi, s'}(w))^2,$$

where $q_{\pi,s'}(w) = \sum_{s'} P_{ss'}^\pi (r_{s,s'} + \gamma\phi_{s'}^T w)$, $P_{ss'}$ is the state transition probability from s to s' . The objective can be formulated as an FS-CEVF problem in Eq. (1):

$$\begin{aligned}
 G_j(w) &= (w, P_{s_j}^\pi (r_{s,j} + \gamma\phi_j^T w))^T \in \mathbb{R}^{d+1} \\
 z &= \frac{1}{|S|} \sum_{j=1}^{|S|} G_j(w) = \left(w, \frac{1}{|S|} \sum_{j=1}^{|S|} P_{s_j}^\pi (r_{s,j} + \gamma\phi_j^T w) \right)^T \in \mathbb{R}^{d+1} \\
 F_i(z) &= (\phi_i^T z_{(1:d)} - |S| \cdot z_{(d+1)})^2 \\
 \Rightarrow f &:= \frac{1}{|S|} \sum_{i=1}^{|S|} F_i \left(\frac{1}{|S|} \sum_{j=1}^{|S|} G_j(w) \right) = \frac{1}{|S|} \sum_{i=1}^{|S|} (\phi_i^T w - q_{\pi,s'}(w))^2.
 \end{aligned}$$

2.4 Example: Mean-Variance Portfolio Optimization

Portfolio optimization characterizes human investment behavior. For each investor, the goal is to maximize the overall return on investment while minimizing the investment risk. Given N assets, the objective function for the mean-variance portfolio optimization problem can be specified as:

$$\max_w \frac{1}{n} \sum_{i=1}^n \langle r_i, w \rangle - \frac{1}{n} \sum_{i=1}^n \left(\langle r_i, w \rangle - \frac{1}{n} \sum_{j=1}^n \langle r_j, w \rangle \right)^2,$$

where $r_i \in \mathbb{R}^N$ is the reward vector observed at time point i . The investment vector w has dimensionality \mathbb{R}^N and is the amount invested in each asset. Profit is described as the mean of the inner products while risk is measured by the variance.

The objective can be formulated as an instance of FS-CEVF optimization problem. First, we change the sign of the maximization problem and turn it into a minimization problem. Then it can be formulated in the form of Eq. (1):

$$\begin{aligned}
 G_j(w) &= (w, \langle r_j, w \rangle)^T \in \mathbb{R}^{N+1} \\
 y &= \frac{1}{n} \sum_{j=1}^n G_j(w) \in \mathbb{R}^{N+1} \\
 F_i(y) &= -y_{(N+1)} + (\langle r_i, y_{(1:N)} \rangle - y_{(N+1)})^2 \in \mathbb{R} \\
 \Rightarrow f &:= \frac{1}{n} \sum_{i=1}^n F_i \left(\frac{1}{n} \sum_{j=1}^n G_j(w) \right) = \frac{1}{n} \sum_{i=1}^n \left(\langle r_i, w \rangle - \frac{1}{n} \sum_{j=1}^n \langle r_j, w \rangle \right)^2 - \frac{1}{n} \sum_{i=1}^n \langle r_i, w \rangle.
 \end{aligned}$$

3 Compositional Stochastic Average Gradient Method

3.1 Stochastic Average Gradient Algorithm

We begin by briefly reviewing SAG [30]. SAG is designed to optimize the sum of a finite number of smooth functions:

$$\min_x f(x) := \frac{1}{n} \sum_{i=1}^n F_i(x) \tag{2}$$

The SAG iterations take the form

$$x^k = x^{k-1} - \frac{\alpha_k}{n} \sum_{i=1}^n y_i^k, \text{ and } y_i^k = \begin{cases} F'_i(x^{k-1}) & \text{if } i = i_k \\ y_i^{k-1} & \text{otherwise,} \end{cases} \quad (3)$$

where α_k is the learning rate and a random index i_k is selected at each iteration.

Essentially, SAG maintains in memory, the gradient of each function in the summation. At each iteration, the gradient value of only one such function is computed and updated. Then SAG evaluate the estimated gradient at each iteration by averaging the gradient values stored in memory. Like stochastic gradient (SG) methods [19, 29], the cost of each iteration is independent of the number of functions in the sum. However, SAG has improved convergence rate compared to conventional SG methods, from $O(1/\sqrt{K})$ to $O(1/K)$ in general, and from the sub-linear $O(1/K)$ to a linear convergence rate of the form $O(\rho^K)$ for $\rho < 1$ when the objective function in Eq. (2) is strongly-convex.

3.2 From SAG to C-SAG

C-SAG extends SAG to accommodate a finite-sum compositional objective function. Specifically, C-SAG maintains in memory, the items needed for updating the parameter. At each iteration, the relevant indices are randomly selected and the corresponding gradients are computed and updated in the memory.

Notations. We denote the value of the variable x at the k -th iteration by x^k . Given a smooth function $H(x) : \mathbb{R}^N \rightarrow \mathbb{R}^M$, ∂H denotes the Jacobian of H defined as:

$$\partial H := \frac{\partial H}{\partial x} = \begin{pmatrix} \frac{\partial H_{(1)}}{\partial x_{(1)}} & \cdots & \frac{\partial H_{(1)}}{\partial x_{(N)}} \\ \vdots & \ddots & \vdots \\ \frac{\partial H_{(M)}}{\partial x_{(1)}} & \cdots & \frac{\partial H_{(M)}}{\partial x_{(N)}} \end{pmatrix}.$$

For function $h(x) : \mathbb{R}^N \rightarrow \mathbb{R}$, the gradient of h is defined by $\nabla h = \left(\frac{\partial h(x)}{\partial x}\right)^T = \left(\frac{\partial h}{\partial x_{(1)}}, \dots, \frac{\partial h}{\partial x_{(N)}}\right)^T$. The gradient of a composition function $f = F(G(x))$ is given by chain rule:

$$\nabla f(x) = (\partial G(x))^T \nabla F(G(x)), \quad (4)$$

where $\nabla F(G(x))$ stands for the gradient of F evaluated at $G(x)$. We use \hat{f} to denote an estimate of the function f , and analogously, $\partial \hat{H}$ and $\nabla \hat{h}$ to denote estimates of Jacobian matrices ∂H and gradient vectors ∇h respectively. To minimize notational clutter, we use G_j^k to denote $G_j(x^k)$, ∂G_j^k to denote $\partial G_j(x^k)$, and ∇F_i^k to denote $\nabla F_i(x^k)$. Given a set A , the cardinality of A is denoted by $|A|$. We use \mathbb{E} to denote expectation.

Algorithm 1. C-SAG

```

1: Require:  $K$  (update period),  $\alpha$  (step size),  $S$  (maximum number of training iterations),
    $a$  (mini batch size)
2: while true do
3:    $J_j = \partial G_j(\tilde{x})$  for  $j = 1, \dots, m$  ▷  $m$  queries
4:    $V_j = G_j(\tilde{x})$  for  $j = 1, \dots, m$  ▷  $m$  queries
5:    $Q_i = \nabla F_i(G(\tilde{x}))$  for  $i = 1, \dots, n$  ▷  $n$  queries
6:    $x_0 = \tilde{x} - \alpha \nabla f(\tilde{x})$ 
7:   for  $k = 1$  to  $K$  do
8:     Uniformly select  $j_k$  in  $[1, 2, \dots, m]$ ,  $A_k$  in  $[1, 2, \dots, m]$  and  $i_k$  in  $[1, 2, \dots, n]$ 
9:     Update memory  $J$  by Eq.(5) and estimate  $\partial \hat{G}(x^{k-1}) = \frac{1}{m} \sum_{j=1}^m J_j^k$  ▷ 1 query
10:    Update memory  $V$  by Eq.(6) and estimate  $\hat{G}(x^{k-1}) = \frac{1}{m} \sum_{j=1}^m V_j^k$  ▷  $|A_k|$  queries
11:    Update memory  $Q$  by Eq.(7) and estimate  $\nabla \hat{F}(\hat{G}(x^{k-1})) = \frac{1}{n} \sum_{i=1}^n Q_i^k$  ▷ 1 query
12:     $x^k = x^{k-1} - \alpha \nabla \hat{f}^k$ 
13:     $\tilde{x} = x^K$ 
14:    if converged or  $S$  reached then
15:      return  $x^K$ 

```

C-SAG Algorithm. We define memories J_j for storing the Jacobian of the inner function $\partial G_j(\cdot)$, V_j for storing the value of the inner function $G_j(\cdot)$, and Q_i for storing the gradient of the outer function $\nabla F_i(\cdot)$. At iteration k , we randomly select indices j_k , i_k , and a mini-batch A_k . We update the memories as follow:

$$J_j^k = \begin{cases} \partial G_j(x^{k-1}) & \text{if } j = j_k \\ J_j^{k-1} & \text{otherwise,} \end{cases} \quad (5)$$

$$V_j^k = \begin{cases} G_j(x^{k-1}) & \text{if } j \in A_k \\ V_j^{k-1} & \text{otherwise,} \end{cases} \quad (6)$$

$$Q_i^k = \begin{cases} \nabla F_i\left(\frac{1}{m} \sum_{j=1}^m V_j^k\right) & \text{if } i = i_k \\ Q_i^{k-1} & \text{otherwise.} \end{cases} \quad (7)$$

Finally, the update rule is obtained by substituting Eqs. (5)–(7) into Eq. (4):

$$\begin{aligned} x^k &= x^{k-1} - \alpha \nabla \hat{f}^k \\ \nabla \hat{f}^k &= (\partial \hat{G}(x^{k-1}))^T \nabla \hat{F}(\hat{G}(x^{k-1})) \\ \partial \hat{G}(x^{k-1}) &= \frac{1}{m} \sum_{j=1}^m J_j^k, \quad \hat{G}(x^{k-1}) = \frac{1}{m} \sum_{j=1}^m V_j^k, \quad \text{and} \quad \nabla \hat{F}(\hat{G}(x^{k-1})) = \frac{1}{n} \sum_{i=1}^n Q_i^k. \end{aligned} \quad (8)$$

In addition, due to the fact that the gradient estimate is biased in the case of FS-CEVF objective function [23], to improve the stability of the algorithm, we use a refresh mechanism to evaluate the exact full gradient periodically and update J , V , and Q . The frequency of refresh is controlled by a pre-specified parameter K . The complete algorithm is shown in Algorithm 1.

3.3 Oracle Query Complexity of C-SAG

In the algorithm, we define the cost of an oracle query as the unit cost of computing $\partial G_j(\cdot)$, $G_j(\cdot)$, and $\nabla F_i(\cdot)$. The runtime of the algorithm is proportional to the number of oracle queries. Evaluating the exact full gradient, such as line 3, 4, and 5 in Algorithm 1, takes m , m , and n oracle queries, for $\partial G(\cdot)$, $G(\cdot)$, and $\nabla F(\cdot)$ respectively. On the other hand, lines 9 to 11 yield $2 + |A_k|$ queries at each iteration, which is independent of the number of component functions.

We proceed to compare the query complexity between C-SAG and the state-of-the-art methods C-SVRG-1 and C-SVRG-2 [23]. C-SVRG-1 requires $2A + 4$ queries in each iteration (where A is the mini-batch size used by C-SVRG-1) except for the reference update iteration, in which exact full gradient is evaluated and requires $2m + n$ queries. The reference update in C-SVRG methods is equivalent to the memory refreshing mechanism in C-SAG. In addition, C-SVRG-2 takes $2A + 2B + 2$ queries per iteration (where B is a second mini-batch size used by C-SVRG-2) except for the reference update iteration. As aforementioned, the proposed C-SAG takes $A + 2$ queries per iteration except for the refresh iteration. The number of queries required by C-SAG during memory refresh iterations are the same as those for C-SVRG-1 and C-SVRG-2 during their reference update iterations. We conclude that C-SAG, in general, incurs only half the oracle query complexity of C-SVRG-1, and less than half that of C-SVRG-2.

4 Convergence Analysis of C-SAG

We proceed to establish the convergence and the convergence rate of C-SAG. Suppose the objective function in Eq. (1) has a minimizer x^* . In what follows, we make the same assumptions regarding the functions $f(\cdot)$, $F(\cdot)$, and $G(\cdot)$ as those used in [23] to establish the convergence of C-SVRG methods. We assume:

- (a) **Strongly Convex Objective:** $f(x)$ in Eq. (1) is strongly convex with parameter μ_f :

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu_f}{2} \|y - x\|^2, \quad \forall x, y. \quad (\Delta_1)$$

- (b) **Bounded Jacobian of Inner Functions:** We assume that the Jacobian of all inner component functions are upper-bounded by B_G ,

$$\|\partial G_j(x)\| \leq B_G, \quad \forall x, \forall j \in \{1, \dots, m\}. \quad (\Delta_2)$$

It naturally follows that since $J_j^k = \partial G_j^{k'}$ for some $k' < k$, we have:

$$\|J_j\| \leq B_G, \quad \forall j \in \{1, \dots, m\}. \quad (\Delta_3)$$

- (c) **Lipschitz Gradients:** We assume that there is a constant L_F such that $\forall x, \forall y, \forall i \in \{1, \dots, n\}$, the following holds:

$$\|\nabla F_i(x) - \nabla F_i(y)\| \leq L_F \|x - y\| \quad (\Delta_4),$$

Armed with these assumptions, we proceed to establish the convergence of C-SAG.

Theorem 1. (Convergence of C-SAG algorithm). For the proposed C-SAG algorithm, we have:

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E} \|x_k - x^*\|^2 \leq \frac{\gamma_1}{\gamma_2} \mathbb{E} \|\tilde{x} - x^*\|^2,$$

$$\begin{aligned} \gamma_1 &= \frac{1}{K} + \left(n\sigma_1 + 3m \left(1 - \left(\frac{a}{m} \right)^2 \right) \sigma_2 \right) B_G^4 L_F^2 \\ \gamma_2 &= \alpha\mu_f - \left(\frac{32\alpha(m-a)}{m\mu_f} + 3a \left(2 - \frac{a}{m} \right) \sigma_2 \right) B_G^4 L_F^2, \end{aligned}$$

$$\begin{aligned} \sigma_1 &= 9\alpha^2 \left((m-1)^2 \left(1 - \frac{1}{n} \right) (n+2) + (n-1)(4m-3) \right) + 16\alpha n \frac{(m-1)^2 \left(1 - \frac{1}{n} \right)^2 + \left(1 - \frac{1}{n} \right)^2}{\mu_f} \\ \sigma_2 &= \frac{9\alpha^2}{m} \left((m-1)^2 (2n-1) + n + 4n(m-1) \right) + 16\alpha \frac{\left(1 - \frac{1}{m} \right)^2 m + 16(m-a)}{m^2 \mu_f}. \end{aligned}$$

Corollary 1. (Convergence rate of C-SAG algorithm). Suppose we choose the user-specified parameters of C-SAG as following:

$$\begin{aligned} a &> m \left(1 - \frac{\mu_f^2}{128B_G^4 L_F^2} \right), \quad \alpha < \min \{ \alpha_1, \alpha_2, \alpha_3 \}, \quad \text{and } K > \frac{8}{\alpha\mu_f}, \\ \alpha_1 &= \frac{m \left(\frac{\mu_f}{12a \left(2 - \frac{a}{m} \right) B_G^4 L_F^2} - 16 \frac{\left(1 - \frac{1}{m} \right)^2 m + 16(m-a)}{m^2 \mu_f} \right)}{9 \left((m-1)^2 (2n-1) + n + 4n(m-1) \right)} \\ \alpha_2 &= \frac{\frac{\mu_f}{8nB_G^4 L_F^2} - 16n \frac{(m-1)^2 \left(1 - \frac{1}{n} \right)^2 + \left(1 - \frac{1}{n} \right)^2}{\mu_f}}{9 \left((m-1)^2 \left(1 - \frac{1}{n} \right) (n+2) + (n-1)(4m-3) \right)} \\ \alpha_3 &= \frac{m \left(\frac{\mu_f}{24m \left(1 - \left(\frac{a}{m} \right)^2 \right) B_G^4 L_F^2} - 16 \frac{\left(1 - \frac{1}{m} \right)^2 m + 16(m-a)}{m^2 \mu_f} \right)}{9 \left((m-1)^2 (2n-1) + n + 4n(m-1) \right)}, \end{aligned}$$

Then we have the following result that implies a linear convergence rate [23]:

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E} \|x_k - x^*\|^2 < \frac{3}{4} \mathbb{E} \|\tilde{x} - x^*\|^2.$$

Proof. The proof proceeds as follows: (i) express $\mathbb{E} \|x^k - x^*\|^2$ in terms of J^{k-1} , V^{k-1} , Q^{k-1} , and x^{k-1} ; (ii) use the assumptions (a) through (c) above to bound J^{k-1} , V^{k-1} , and Q^{k-1} in terms of x^{k-1} and \tilde{x} ; and (iii) use the preceding bounds to establish the convergence of C-SAG, and the choice of parameters to ensure convergence rate. A full proof is given in the Appendix of the arXiv preprint of this paper¹.

¹ arXiv:1809.01225.

5 Experimental Evaluation of C-SAG

We report results of comparison of C-SAG with two variants of C-SVRG [23], C-SVRG-1 and C-SVRG-2, which are among the state-of-the-art methods for FS-CEVF optimization, as well as the classical full gradient (FG) method, on the mean-variance portfolio optimization problem (see Sect. 2 for details).

Our experimental setup closely follows the setup in [23]. Synthetic data were drawn from a multi-variate Gaussian distribution and absolute values of the data samples were retained. We generated two synthetic data sets (D1 and D2). D1 consists of 2000 time points, and 200 assets, i.e., $n = 2000, N = 200$; D2 consists of 5000 time points, and 300 assets, i.e., $n = 5000, N = 300$. We also controlled κ_{cov} , the condition number of the covariance matrix used in the multi-variate Gaussian distribution to generate the synthetic data. The initialization and the step sizes were chosen to be identical for all algorithms. The parameters are set to their default values in the implementation provided by the authors of [23]: The size of the mini-batch was set to 20 ($a = 20$), the update period was set to 20 iterations ($K = 20$), and the constant step size was set to 0.12 ($\alpha = 0.12$).

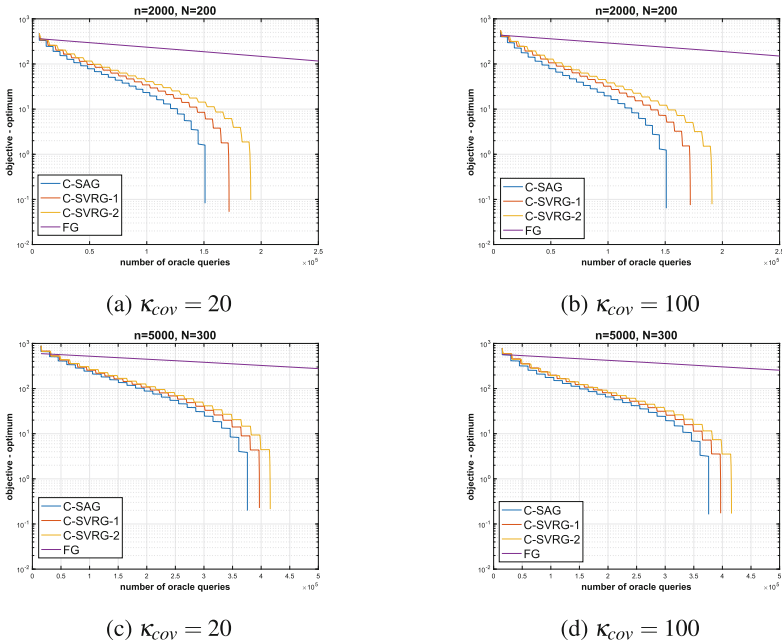


Fig. 1. Mean-variance portfolio optimization on synthetic data D1 with 200 assets ($N = 200$) and the number of time points $n = 2000$ (first row), and D2 with 300 assets and 5000 time points (second row). The logarithm of the objective value minus the optimum is plotted along the Y-axis and the number of oracle queries is plotted along the X-axis. The κ_{cov} is the conditional number of the covariance matrix of the corresponding Gaussian distribution used to generate reward vectors.

The experimental results comparing the different methods on data sets D1 and D2 are shown in Fig. 1. Note that the sudden drop at the tail of each curve results from treating the smallest objective value in the convergence sequence, which is usually found at the end of the sequence, as the optimum value. We observe that on both data sets, all stochastic gradient methods converge faster than full gradient (FG) method, and that C-SAG achieves better (lower) value of the objective function at each iteration. Although the three stochastic methods under comparison (C-SVRG-1, C-SVRG-2, and our method, C-SAG) all have linear convergence rates, C-SAG converges faster in practice by virtue of lower oracle query complexity per iteration as compared to C-SVRG-1 and C-SVRG-2.

6 Summary and Discussion

Many machine learning, statistical inference, and portfolio optimization problems require minimization of a composition of expected value functions. We have introduced C-SAG, a novel extension of SAG, to minimize composition of finite-sum functions, i.e., finite sum variants of composition of expected value functions. We have established the convergence of the resulting algorithm and shown that it, like other state-of-the-art methods e.g., C-SVRG, achieves a linear convergence rate when the objective function is strongly convex, while benefiting from lower oracle query complexity per iteration as compared to C-SVRG. We have presented results of experiments that show that C-SAG converges substantially faster than the state-of-the-art C-SVRG variants.

Work in progress is aimed at (i) analyzing the convergence rate of C-SAG for weakly convex problems; (ii) developing a distributed optimization algorithm by combining C-SAG with the ADMM framework [6]; and (iii) applying C-SAG and its variants to problems of practical importance in machine learning.

Acknowledgement. This project was supported in part by the National Center for Advancing Translational Sciences, National Institutes of Health through the grant UL1 TR000127 and TR002014, by the National Science Foundation, through the grants 1518732, 1640834, and 1636795, the Pennsylvania State Universitys Institute for Cyber-science and the Center for Big Data Analytics and Discovery Informatics, the Edward Frymoyer Endowed Professorship in Information Sciences and Technology at Pennsylvania State University and the Sudha Murty Distinguished Visiting Chair in Neuro-computing and Data Science funded by the Pratiksha Trust at the Indian Institute of Science [both held by Vasant Honavar]. The content is solely the responsibility of the authors and does not necessarily represent the official views of the sponsors.

References

1. Amari, S.I.: Backpropagation and stochastic gradient descent method. *Neurocomputing* **5**(4–5), 185–196 (1993)
2. Bishop, C.: *Pattern Recognition and Machine Learning*. Springer, New York (2006). <https://doi.org/10.1007/978-1-4615-7566-5>

3. Bottou, L.: Stochastic gradient learning in neural networks. *Proc. Neuro-Nimes* **91**(8), 12 (1991)
4. Bottou, L.: Large-scale machine learning with stochastic gradient descent. In: Lechevallier, Y., Saporta, G. (eds.) *Proceedings of COMPSTAT*, pp. 177–186. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-7908-2604-3_16
5. Bottou, L., Curtis, F.E., Nocedal, J.: Optimization methods for large-scale machine learning. *SIAM Rev.* **60**(2), 223–311 (2018)
6. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends® Mach. Learn.* **3**(1), 1–122 (2011)
7. Cauwenberghs, G.: A fast stochastic error-descent algorithm for supervised learning and optimization. In: *Advances in Neural Information Processing Systems*, pp. 244–251 (1993)
8. Dai, B., He, N., Pan, Y., Boots, B., Song, L.: Learning from conditional distributions via dual embeddings. *arXiv preprint [arXiv:1607.04579](https://arxiv.org/abs/1607.04579)* (2016)
9. Darken, C., Moody, J.: Fast adaptive K-means clustering: some empirical results. In: *International Joint Conference on Neural Networks*, pp. 233–238. IEEE (1990)
10. Dentcheva, D., Penev, S., Ruszczyński, A.: Statistical estimation of composite risk functionals and risk optimization problems. *Ann. Inst. Stat. Math.* **69**(4), 737–760 (2017)
11. Ermoliev, Y.: Stochastic quasigradient methods. In: Ermoliev, Y., Wets, R.J.-B. (eds.) *Numerical techniques for stochastic optimization*, no. 10. Springer, Heidelberg (1988)
12. Fagan, F., Iyengar, G.: Unbiased scalable softmax optimization. *arXiv preprint [arXiv:1803.08577](https://arxiv.org/abs/1803.08577)* (2018)
13. Friedman, J., Hastie, T., Tibshirani, R.: *The Elements of Statistical Learning*. Springer Series in Statistics, vol. 1. Springer, New York (2001). <https://doi.org/10.1007/978-0-387-21606-5>
14. Hu, J., Zhou, E., Fan, Q.: Model-based annealing random search with stochastic averaging. *ACM Trans Model. Comput. Simul.* **24**(4), 21 (2014)
15. Huo, Z., Gu, B., Huang, H.: Accelerated method for stochastic composition optimization with nonsmooth regularization. *arXiv preprint [arXiv:1711.03937](https://arxiv.org/abs/1711.03937)* (2017)
16. Jain, P., Kakade, S.M., Kidambi, R., Netrapalli, P., Sidford, A.: Accelerating stochastic gradient descent for least squares regression. In: *Conference on Learning Theory*, pp. 545–604 (2018)
17. Jin, C., Kakade, S.M., Netrapalli, P.: Provable efficient online matrix completion via non-convex stochastic gradient descent. In: *Advances in Neural Information Processing Systems*, pp. 4520–4528 (2016)
18. Johnson, R., Zhang, T.: Accelerating stochastic gradient descent using predictive variance reduction. In: *Advances in Neural Information Processing Systems*, pp. 315–323 (2013)
19. Kiefer, J., Wolfowitz, J.: Stochastic estimation of the maximum of a regression function. *Ann. Math. Stat.* **23**(3), 462–466 (1952)
20. Le, Q.V., Ngiam, J., Coates, A., Lahiri, A., Prochnow, B., Ng, A.Y.: On optimization methods for deep learning. In: *Proceedings of the 28th International Conference on Machine Learning*, pp. 265–272. Omnipress (2011)
21. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. *Nature* **521**(7553), 436 (2015)
22. LeCun, Y.A., Bottou, L., Orr, G.B., Müller, K.-R.: Efficient BackProp. In: Montavon, G., Orr, G.B., Müller, K.-R. (eds.) *Neural Networks: Tricks of the Trade*. LNCS, vol. 7700, pp. 9–48. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35289-8_3

23. Lian, X., Wang, M., Liu, J.: Finite-sum composition optimization via variance reduced gradient descent. In: *Artificial Intelligence and Statistics*, pp. 1159–1167 (2017)
24. Lin, T., Fan, C., Wang, M., Jordan, M.I.: Improved oracle complexity for stochastic compositional variance reduced gradient. arXiv preprint [arXiv:1806.00458](https://arxiv.org/abs/1806.00458) (2018)
25. Liu, L., Liu, J., Tao, D.: Variance reduced methods for non-convex composition optimization. arXiv preprint [arXiv:1711.04416](https://arxiv.org/abs/1711.04416) (2017)
26. Mandt, S., Hoffman, M.D., Blei, D.M.: Stochastic gradient descent as approximate Bayesian inference. *J. Mach. Learn. Res.* **18**(1), 4873–4907 (2017)
27. Needell, D., Ward, R., Srebro, N.: Stochastic gradient descent, weighted sampling, and the randomized Kaczmarz algorithm. In: *Advances in Neural Information Processing Systems*, pp. 1017–1025 (2014)
28. Ravikumar, P., Lafferty, J., Liu, H., Wasserman, L.: Sparse additive models. *J. R. Stat. Soc.: Ser. B (Stat. Methodol.)* **71**(5), 1009–1030 (2009)
29. Robbins, H., Monro, S.: A stochastic approximation method. In: Lai, T.L., Siegmund, D. (eds.) *Herbert Robbins Selected Papers*, pp. 102–109. Springer, New York (1985)
30. Schmidt, M., Le Roux, N., Bach, F.: Minimizing finite sums with the stochastic average gradient. *Math. Program.* **162**(1–2), 83–112 (2017)
31. Shamir, O.: Convergence of stochastic gradient descent for PCA. In: *International Conference on Machine Learning*, pp. 257–265 (2016)
32. Shapiro, A., Dentcheva, D., Ruszczyński, A.: *Lectures on Stochastic Programming: Modeling and Theory*. SIAM, Philadelphia (2009)
33. Sutton, R.S., Barto, A.G.: *Reinforcement Learning: An Introduction*. MIT Press, Cambridge (1998)
34. Tan, C., Ma, S., Dai, Y.H., Qian, Y.: Barzilai-Borwein step size for stochastic gradient descent. In: *Advances in Neural Information Processing Systems*, pp. 685–693 (2016)
35. Tibshirani, R.: Regression shrinkage and selection via the lasso. *J. R. Stat. Society. Ser. B (Methodol.)* 267–288 (1996)
36. Wang, L., Yang, Y., Min, R., Chakradhar, S.: Accelerating deep neural network training with inconsistent stochastic gradient descent. *Neural Netw.* **93**, 219–229 (2017)
37. Wang, M., Fang, E.X., Liu, H.: Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions. *Math. Program.* **161**(1–2), 419–449 (2017)
38. Wang, M., Liu, J., Fang, E.: Accelerating stochastic composition optimization. In: *Advances in Neural Information Processing Systems*, pp. 1714–1722 (2016)
39. Yu, Y., Huang, L.: Fast stochastic variance reduced ADMM for stochastic composition optimization. arXiv preprint [arXiv:1705.04138](https://arxiv.org/abs/1705.04138) (2017)
40. Yuan, M., Lin, Y.: Model selection and estimation in the Gaussian graphical model. *Biometrika* **94**(1), 19–35 (2007)
41. Zhang, T.: Solving large scale linear prediction problems using stochastic gradient descent algorithms. In: *Proceedings of the 21st International Conference on Machine Learning*, p. 116. ACM (2004)
42. Zhao, S.Y., Li, W.J.: Fast asynchronous parallel stochastic gradient descent: a lock-free approach with convergence guarantee. In: *AAAI*, pp. 2379–2385 (2016)
43. Zinkevich, M., Weimer, M., Li, L., Smola, A.J.: Parallelized stochastic gradient descent. In: *Advances in Neural Information Processing Systems*, pp. 2595–2603 (2010)



Instance-Based Stacked Generalization for Transfer Learning

Yassine Baghoussi^{1,2}(✉)  and João Mendes-Moreira^{1,3} 

¹ LIAAD-INESC TEC, Porto, Portugal
baghoussi.yassine@gmail.com

² Faculty of Science, University of Porto, Porto, Portugal

³ Faculty of Engineering, University of Porto, Porto, Portugal
jmoreira@fe.up.pt

Abstract. We present a method for improving the prediction accuracy using multiple predictive algorithms. Several techniques have been developed to tackle this issue such as bagging, boosting and stacking. In contrary to the first two that, usually, generate homogeneous ensembles of classifiers, stacking techniques have demonstrated success using heterogeneous ensembles. In our method, we adopt the stacking mechanism. Several models are generated using different learning algorithms. Forward stepwise selection is implemented to link each instance to its appropriate learning model. Experiments with three datasets benchmarked with four learning schemes show that this novel method improves prediction accuracy and can serve as a bridge to transfer knowledge between tasks given the same feature space but different data distributions.

Keywords: Ensemble learning · Stacking · Transfer learning
Operational planning

1 Introduction

Every supervised learning tasks, namely classification and regression consist of (1) data collection for the given task of interest; (2) learning a model on this data; (3) apply the model to new data/task. A major assumption in traditional supervised learning is that the training and test data must be in the same feature space and follow the same distribution. However, in real world applications, the learned-from dataset (source) may not have the same data distribution or feature space as the predicted-on dataset (target). As a result, most statistical models need to be rebuilt from scratch using newly collected training data. Overcoming

This work is funded by Project “TEC4Growth - Pervasive Intelligence, Enhancers and Proofs of Concept with Industrial Impact/NORTE-01-0145-FEDER-000020”, a project financed by North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF). We also thank STCP - Sociedade de Transportes Colectivos do Porto, SA, for the data used in this work.

such situation is a critical issue because usually it is expensive or impossible to re-collect the needed training data and rebuild the models. There are two approaches to reusing the model from the first task as a starting point for the new model of another different but related task: transfer learning and meta-learning.

1.1 Transfer Learning

In human psychology, Transfer Learning (TL) is defined as an embedded skill in human, it reads as the impact of one knowledge on another. For example, human can learn to recognize a pear by knowing apples; to speak Portuguese by knowing French, and so on. In machine learning, TL was defined in [1] as follows: given a source domain D_S and a learning task T_S , a target domain D_T and a learning task T_T , transfer learning aims to help improve the learning of the target predictive function $f_T(\cdot)$ in D_T using the knowledge in D_S and T_S , where $D_S \neq D_T$, or $T_S \neq T_T$. In contrast to traditional machine learning techniques, TL abandons the assumption that the source and target domain should be from the same feature space and data distribution. Hence, it reduces the dependencies of the target task and source data and widens the migration of knowledge. In other words, it leverages knowledge from one domain (source) to another domain (target). For example, suppose a transport company is planning a new bus route. The company has no data about the new route, however other data have been collected in another bus routes of the same network [2]. Intuitively, one can apply transfer learning techniques to transfer relevant travel information such as travel time, dwell time and delays from one bus line data to the new bus line.

Machine learning works on transfer learning can be organized into two categories: homogeneous and heterogeneous. In contrast to homogeneous, heterogeneous transfer learning does not assume an equivalence and an overlap of data distribution and feature space between source and target domain [3]. Most transfer learning solutions fall into two categories: feature-based or instance-based (symmetric or asymmetric). But, other solutions still exist such as model parameters transfer and relational-knowledge transfer approaches [3]. In Instance-based TL, instances from source are used as a support to target instances in order to improve the performance of the target model, as manifold alignment [4] does; In feature-based transfer learning, some approaches transfer feature-representation extracted from the source domain to construct feature representation of the target domain, such as heterogeneous spectral mapping [5], feature mapping [6] and feature projection and transformation [7]. In model parameter transfer, it is assumed that the models in both source and target domains share some parameters. A source model is used to extract these parameters which are later used to build a target model [8]. Finally, in relational knowledge transfer, knowledge about data is transferred between source and target domains [9].

1.2 Meta Learning

Selecting the appropriate predictive algorithm to use for a dataset of interest is challenging. The task of tackling this issue was pioneered by Schmidhuber in 1987 [10] and is called meta-learning. Ultimately, meta-learning approach aims at exploiting the repetitious use of a determined method over a set of similar tasks. Meta learning approaches are numerous. For instance, some meta-learning algorithms aim at managing bias and are commonly adopted for data streams (i.e. continuous and ordered sequence of instances). Data stream mining requires domain adaptation due to the fact that the domain is not static. In other case studies, meta-learning addresses meta-knowledge transfer across tasks. This latter knowledge transfer is often used for the Algorithm Selection Problem [11].

Another approach of meta learning aims at combining a number of base learners together in order to create a combined model that is able to improve the prediction accuracy. The idea consists on using knowledge collected about the performance of a set of learning algorithms at the base level; such knowledge is then referred to as meta-knowledge. Among the methods, one can find: stacked generalization.

Stacked Generalization. Meta-knowledge can combine predictions of base learners, a process known as stacked generalization [12]. The process is conceptualized as a set of layers. Each of a set of base-learners will be trained on a dataset; the original feature space is then extended to include the predictions of these learners. The output of each layer is given as an input of the immediately succeeding layer. A single learner will be produced at the topmost level for the final predictions. The knowledge gained through predictions issued by base learners is called meta knowledge and the predictions added to the original dataset are called meta-features. This is why stacked generalization is considered a meta-learning task. Works in this area study: the exploitation of base learners outputs (i.e. meta knowledge [13]); and how to define meta-features [14].

In the following sections, we will introduce a method named Stacked Generalization approach using Instance-based Best Algorithm Selection (SG-IBAS). We apply this approach to a real world data and discuss the results. We conclude by discussing future works.

2 SG-IBAS Implementation

The approach in Fig. 1 works under a layered architecture. A dataset of interest is split into three random subsets of equal size: train-1, train-2 and test.

In the **level-0** layer, for each train set we learn N base-learners and predict on the remaining train set and test set. The original feature representation is then extended to include the predictions of these base learners. The predictions, in train sets, are ranked with respect to a given error measure and, according to the best prediction, we store the ID of the best learner. As a result, the meta-datasets consist of: Best Algorithm ID, base learners predictions added to the

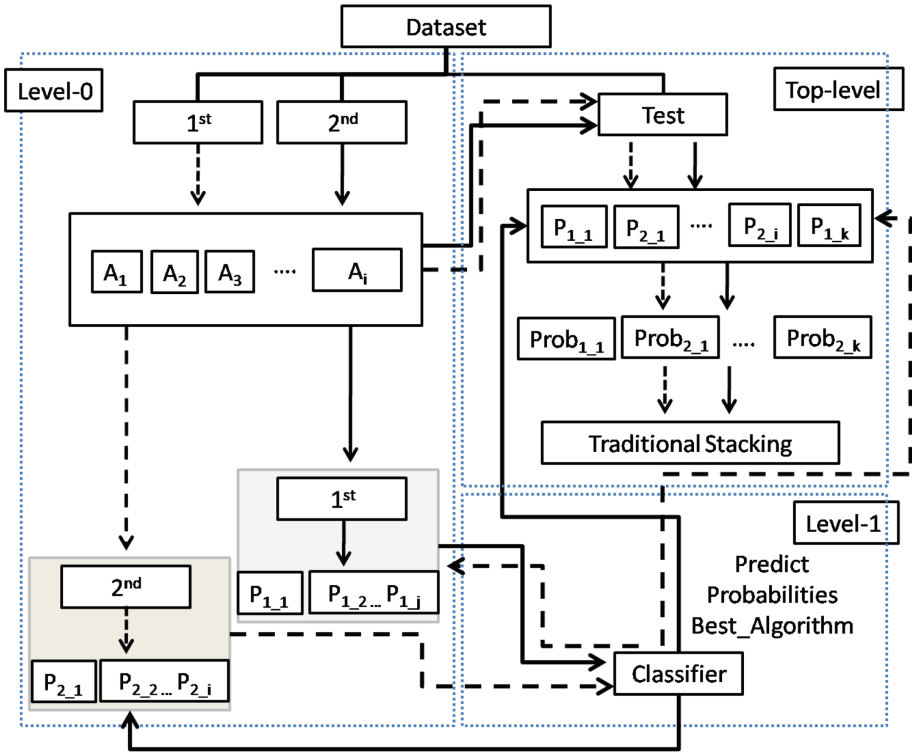


Fig. 1. Stacked Generalization approach using Instance-based Best Algorithm Selection.

original dataset (i.e. train-1 or train-2) while test set contains only predictions. In contrary to traditional stacking frameworks, we choose to learn and predict using the two train sets in order to avoid the loss of information when learning in only one dataset. Thus, the predictions ranking process, in test set, is made according to these two train sets (i.e. train-1 and train-2). In other words, we produce models corresponding to each learner in each train set: i.e. two pairs (N learners, train-1) and (N learners, train-2).

In the next **level-1**, we feed two classifiers, using train-1 and train-2 respectively. Each classifier learn to predict the Best Algorithm ID on the test set and the remaining train set. As a result, the meta-test set includes: the predictions of base learners, the output of each classifier (i.e. each classifier will learn from base learners predictions to predict the performance of each base learner in the test set. Each classifier will output probabilities).

Assuming each of the N learners outperformed the others in predicting the output of a number of instances, thus the **top-level** layer will produce N models: one from each N learner to predict on its test set sample. Each learning probability consists of the performance of a model constituted of one pair (learner,

train-1) or (learner, train-2). Thus, each of these N models must learn from the test instance corresponding train set (i.e. train-1 or train-2 depending on the probabilities of **level-1**). In Top-level layer, we pursue a traditional stacking ensemble for each train set. The predictions of each ensemble in the meta-test set will be due to a metric. In our experiments, we have averaged the two outputs but other metric can be used such as weighting according to each ensemble, in each instance.

Other related approaches have addressed the issue of adding a meta-layer in order to output instance-based best algorithm [15]. The method is called grading and is similar to stacking. The principle is to show to each classifier where it errs. However, this approach is used for classification problems only. In our approach, we build on the same principle. We combine grading and stacking by using the probabilities of each algorithm (i.e. the probability of how good an algorithm is to predict each instance) as inputs for a stacking ensemble. Most of the stacked generalization approaches implementing combination of base learners take advantage of the base learners predictions, directly or engineered for a particular need, and are given as input for the successive learners.

3 Experiments

Experiments in this section show that:

1. for a successful stacked generalization, we can use meta-features to perform instance-based algorithm selection;
2. k-NN is suitable for SG-IBAS level-1 classification task (i.e. to learn the probabilities for the best algorithm selection);
3. SG-IBAS may be applied to tackle transfer learning from a source domain to a target domain having equal feature spaces but different data distributions as it bypasses the distributions dependency.

The framework is tested on three real world datasets provided by STCP - Sociedade de Transportes Colectivos do Porto, the bus transport company of the city of Porto, Portugal. The dataset consists of Automated Vehicle Location data collected in the year 2010 from three bus lines: Line 200, 502 and 600. It gives information on the number of vehicle, travel date/time, number of bus line, number of shift, number of trip, number of the journey, number of stop order, direction and stop id. We reshape the AVL data in order to convert it from a set of arrivals to a set of trips. Then, we merge the trips data with the expected travel times using timetables of the same year 2010. Thus, the predictive task is a regression one and the predicted value is the $target = actualtime - expectedtime$. All the datasets have the same feature space but different data distribution. As pre-processing we used z-normalization. Our learning scheme was tested against the following methods: SVMradial, NeuralNet, Linear Regression (lm), Standard Stacked Generalization (SSG).

SG-IBAS setting in level-0 consists of SVMradial and neural network as base learners. In the level-1 layer, we have compared four classifiers applied to our

classification problem: k-nn, SVMlinear, random forest and generalized linear model. k-NN did it better in this case study as shown in Fig. 2. In the top-level, we adopt linear regression ensemble to be added in two SG-IBAS ensembles: SG-IBAS without base learners predictions (SG-IBAS-without) and SG-IBAS with base learners predictions (SG-IBAS-with).

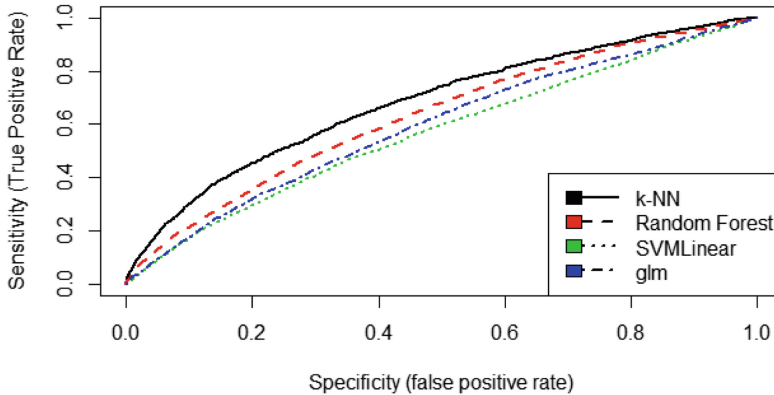


Fig. 2. ROC curves: k-NN holds the best performance.

The experiments are run in R using the caret package. In SVM, we used the radial kernel using grid search for hyper-parameter tuning. The optimal values obtained were: cost = 10000, gamma = 0.00001 and epsilon = 0.1. For Neural Net, we also tuned the hyper-parameters using grid search. The optimal values were decay = 0.1 and size = 4. For the stacking ensemble, we used the CaretEnsemble package available in R. It works as follows: (1) learn a number of base models; (2) use predictions as input for a meta learner.

Table 1 summarizes the RMSE values obtained for the bus lines 600, 502 and 200. Each row in the table, represents the RMSE errors resulted from each learning scheme. The results demonstrate that both SG-IBAS schemes outperformed the others in all the cases. For the lines 600 and 502, the base learners predictions slightly improved the accuracy.

Table 2 illustrates the decrease of RMSE errors during transfer learning tasks. To perform transfer learning, we have learned in each bus line data and tested the learning on the remaining bus line data sets. The datasets have the same feature space but different data distribution. According to the result, we notice that SG-IBAS is less sensitive to the difference of data distribution compared to traditional stacking. The reason is that according to the steps on SG-IBAS, we are reducing the training space and test set space into a partly-common subspace. For instance, in level-1, we use the predictions of base learners (i.e. both in train and test sets) to predict the probabilities of each learner to each instance. Even though, in the majority of the cases, SG-IBAS was able to decrease the error, in one case SG outperformed (i.e. learning in Line 600 and testing in Line 200).

Table 1. Root Mean Squared Error for the Best Single Models (i.e. SVM, NN, LM), Traditional (SG), Stacked Generalization IBAS without base learners predictions (SG-IBAS-without) and Stacked Generalization with base learners predictions (SG-IBAS-with).

Line	nnet	svmRadial	lm	SG	SG-IBAS-without	SG-IBAS-with
L600	0.1488	0.1563	0.1564	0.1488	0.1477	0.1476
L502	0.1255	0.1356	0.1358	0.1255	0.1258	0.1252
L200	0.1223	0.1324	0.1321	0.1224	0.1214	0.1218

Table 2. Transfer knowledge between bus lines 502, 600 and 200. Comparative RMSE errors results between Stacked Generalization (SG), Stacked Generalization Instance-based Best Algorithm Selection SG-IBAS with and without base learners predictions.

Source domain	200		502		600	
Target domain	600	502	600	200	502	200
SG	0.1670	0.2264	0.1874	0.2023	0.1615	0.1535
IBAS-without	0.1465	0.1768	0.1739	0.1655	0.1625	0.1621
IBAS-with	0.1496	0.1728	0.1775	0.1609	0.1587	0.1676

One possible difficulty of this approach occurs in the second layer. Depending on each domain task, the classifier should be selected. A cross validation and a comparative analysis is required. Moreover, the present approach is very sensitive to the training size: the largest the data is, the more efficient this approach is.

4 Conclusion

In this paper, we have addressed a new learning scheme of stacked generalization, namely, SG-IBAS. We demonstrated how the predictions of base learners can be deterministic in predicting the probabilities of each learner capability in predicting each instance on test set. We make use of these probabilities learned from the predictions instead of using these predictions directly as input attributes for higher-level learning. One of the challenges in stacked generalization approaches is the loss of information during the split of train sets. Thus, to overcome this issue we proposed to use both samples in order to increase the accuracy of predictions and probabilities in our meta-layer.

The method was tested on a regression problem. We have shown that SG-IBAS allowed us to improve the predictions by combining two base learners NeuralNet and SVM radial. We also opened an interesting discussion about transfer learning and showed that by testing the approach on datasets of the same feature spaces but different data distribution, we could decrease the errors.

For future works, we aim at analyzing the common subspace produced by the learning scheme. In this work, we have focused on homogeneous transfer learning

as the source and target domains must be from the same feature space in order to perform SG-IBAS. For future works, we tend to improve the learning scheme to heterogeneous transfer learning.

References

1. Pan, S.J., Yang, Q.: A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **22**(10), 1345–1359 (2010). <https://doi.org/10.1109/TKDE.2009.191>
2. Baghoussi, Y., Mendes-Moreira, J., Emmerich, M.T.M.: Updating a robust optimization model for improving bus schedules. In: 2018 10th International Conference on Communication Systems Networks (COMSNETS), pp. 619–624, January 2018
3. Day, O., Khoshgoftaar, T.M.: A survey on heterogeneous transfer learning. *J. Big Data* **4**(1), 29 (2017). <https://doi.org/10.1186/s40537-017-0089-0>
4. Wang, C., Mahadevan, S.: Heterogeneous domain adaptation using manifold alignment. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume, IJCAI 2011, vol. 2, pp. 1541–1546. AAAI Press (2011). <https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-259>
5. Shi, X., Liu, Q., Fan, W., Yu, P.S., Zhu, R.: Transfer learning on heterogeneous feature spaces via spectral transformation. In: 2010 IEEE International Conference on Data Mining, pp. 1049–1054, December 2010
6. Wu, X., Wang, H., Liu, C., Jia, Y.: Cross-view action recognition over heterogeneous feature spaces. In: 2013 IEEE International Conference on Computer Vision, pp. 609–616, December 2013
7. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 213–226. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15561-1_16
8. Gao, J., Fan, W., Jiang, J., Han, J.: Knowledge transfer via multiple model local structure mapping. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2008, pp. 283–291. ACM, New York (2008)
9. Wang, H., Yang, Q.: Transfer learning by structural analogy. In: Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, pp. 513–518. AAAI Press (2011). <http://dl.acm.org/citation.cfm?id=2900423.2900505>
10. Schmidhuber, J.: Evolutionary principles in self-referential learning. On learning how to learn: the meta-meta-meta...-hook. Diploma thesis, Technische Universität München, Germany, 14 May 1987
11. Brazdil, P., Giraud-Carrier, C.G., Soares, C., Vilalta, R.: Metalearning (2009)
12. Wolpert, D.H.: Stacked generalization. *Neural Netw.* **5**(2), 241–259 (1992)
13. Gama, J., Brazdil, P.: Cascade generalization. *Mach. Learn.* **41**(3), 315–343 (2000)
14. Seewald, A.K., Fürnkranz, J.: An evaluation of grading classifiers. In: Hoffmann, F., Hand, D.J., Adams, N., Fisher, D., Guimaraes, G. (eds.) IDA 2001. LNCS, vol. 2189, pp. 115–124. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44816-0_12
15. Duch, W., Maszczyk, T., Grochowski, M.: Optimal support features for meta-learning. In: Jankowski, N., Duch, W., Grabczewski, K. (eds.) Meta-Learning in Computational Intelligence. Studies in Computational Intelligence, vol. 358, pp. 317–358. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20980-2_10



Combined Classifier Based on Quantized Subspace Class Distribution

Paweł Ksieniewicz^(✉) 

Wrocław University of Science and Technology,
Wyb. Wyspiańskiego 27, 50-370 Wrocław, Poland
pawel.ksieniewicz@pwr.edu.pl

Abstract. Following paper presents *Exposer Ensemble* (EE), being a combined classifier based on the original model of quantized subspace class distribution. It presents a method of establishing and processing the *Planar Exposer* – base representation of discrete class distribution over given subspace, and a proposition how to effectively fuse discriminatory power of many *Planar Exposers* into a combined classifier. The natural property of the representation used in the following article is its resistance to the imbalance of training data, without the need to use over- or undersampling methods and the constant computational complexity of prediction. Description of proposed algorithm is complemented by a series of computer experiments conducted on the collection of balanced and imbalanced datasets with diverse imbalance ratio, proving its usefulness in a supervised learning task.

Keywords: Supervised learning · Classification · Classifier ensemble

1 Introduction

In the age of increased interest in the subject of *Machine Learning* problems, there may be observed a large intensification of research on neural models, with particular focus on the topics of *Deep* [20] and *Convolutional Neural Networks*. Except for the ability of effective feature extraction [13, 18] available for such systems, the most often considered application of NN is to solve the problem of *supervised learning* in the task of *classification* [5, 21], where the aim of the algorithm is to acquire knowledge necessary to assign new, unknown objects to one of the predefined groups basing on the labeled set of training data [14].

Since the beginning of *Artificial Intelligence* as a branch of science, various models used for *pattern classification* have been studied. Starting from the simplest solutions, such as *k-Nearest Neighbors* or the *Nearest Centroid Classifier*, where the decision about assigning an object to the class is based on a direct comparison of its similarity to data collected in the learning process, by probabilistic models, such as the *Naïve Bayes Classifier* [2], and *Decision Trees*, to the *Support Vector Machines*, whose learning process is looking for a hyperplane separating with the maximum margin patterns belonging to different classes, to enumerate only a few of the most commonly used approaches.

The variety of available classification models allows, among others, for the construction of effective *classifier ensembles*, where one of the most fundamental problems is to ensure a suitably diverse pool of classifiers making independent decisions [12]. *Ensembles* should be designed to provide a proper fusion of independent responses, allowing to achieve accuracy higher than each of their base classifiers. Diversification of the classifier pool may be ensured by a mutual *heterogeneity* of the models or by a *homogeneous* pool, in which each member builds knowledge on the basis of another subset of the training data.

One of the well-known approaches of this kind is *Random Subspace*, where each of the member classifiers receives a different subspace of features for learning [8]. This is a proper solution especially in the context of the *curse of dimensionality* [3], which tells us about the difficulties of classic approaches in the processing of patterns with many attributes. Building member classifiers in such manner allows us to break down the multidimensional problem of classification into many problems of lower dimensionality, which require more achievable computing power.

A typical path to classification is to assess the testing object probability of belonging to one of the problem classes, based on the accumulation of objects from the training set around the point representing it in the feature space [11]. This approach, however, is highly biased to the majority class of imbalanced data problems, where the minority class is represented only by a small percentage of the training set [10]. A frequent solution to this problem is *undersampling* of the majority class [17, 19] aiming to equalize the number of objects of different classes by eliminating portion of objects from the majority class, or *oversampling*, which target is to multiply the minority class patterns [6] or, in more sophisticated algorithms such as SMOTE [4] or ADASYN [9], we try to generate new, synthetic samples based on the existing ones.

Bearing in mind the above-mentioned problems and concepts, following paper proposes a classifier, whose model does not follow any of the commonly used approaches, aiming to extend the classification algorithms pool diversity. It is based on establishing the discriminatory power in the area of two-dimensional subspaces of features, also proposing a measure of their quality assessment in order to determine weights for a committee built of multiple projections of the original feature space to try to construct an effective combined classifier. In addition, the learning procedure of a proposition is not based on the number of patterns from different classes over the feature space, but on the image of their normalized distribution within the analyzed subspace, which is an attempt to construct a model that is resistant to imbalance of data without the need for under and oversampling methods.

2 Proposed Method

2.1 Preparing *Planar Exposer*

Elementary structure and the base model of the proposed solution is the *Planar Exposer*, which is a three-dimensional, discrete structure built on the basis of

information extracted from two dimensions of a feature space. Its size is determined by the quantum hyperparameter, called the *grain* (g) and the number of classes present in the training set.

To explain the principles of *Planar Exposer* construction, the example of the popular, benchmark *Iris* dataset will be used. It consists of 150 patterns, described by four features and equally divided into three classes. To establish the model of a *Planar Exposer*, first two features of the set will be used.

In the first processing step, each of the analyzed features, independently, need to be transformed with a *Min-Max Scaler* to the range from zero to the assumed g value – for the purpose of the example established at 8. To provide a clear visualization, each of classes is assigned to the components of the RGB color model. The *scatter plot* of objects rescaled with this approach is presented in Fig. 1(a).

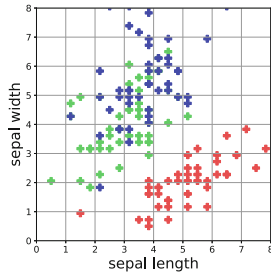


Fig. 1. Scatter plot of Iris dataset

After scaling, it is necessary to place a grid on the analyzed subspace, dividing it into number of rows and columns equal to g parameter. As a next step, in a matter similar to histogram calculation, we add up number of samples *falling into* each cell, independently for the each class. If the sum contained in each cell of the matrix is divided by the maximum count achieved within the class, we will establish information about the discrete probability distribution of its occurrence depending on the subspace location. The visualization of such result can be seen in Fig. 2.

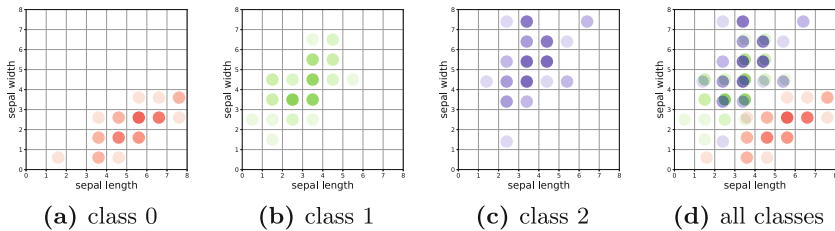


Fig. 2. Quantized distribution of iris dataset classes

The presence of class samples is marked by circles of different colors with brightness depending on the intensity of the patterns occurrence.

In a natural parallel, we can interpret every cell of such grid as a matrix of receptors, sensitive to the presence of objects of particular class. We call the above process a *sample exposition*.

The proposed representation may also be interpreted as an image with a resolution of the g hyperparameter and n color channels, where n is equal to the number of classes in the training set. With the *Iris* set, due to the presence of three classes, it is possible to directly visualize the *Exposer* in the RGB model, as is shown in Fig. 3.

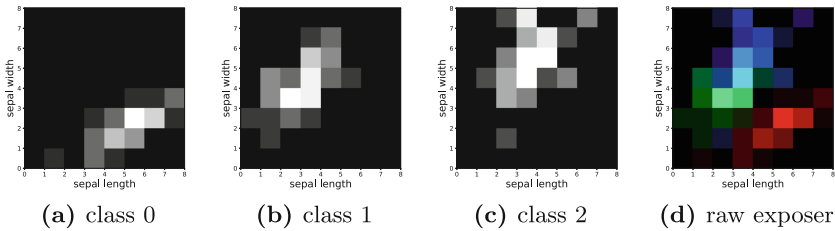


Fig. 3. The raw *Planar Exposer* and its layers

The limitation to only two dimensions, which allows a picture interpretation, leads to the possibility to employ image processing methods to improve established model. The proposal uses two algorithms. The first one is *anisotropic diffusion* [16], applied independently to each channel, which allows to join information between neighboring pixels while maintaining the shapes resulting from the exposition. It is calibrated using the iteration number parameter (a). The second algorithm, *Gaussian filter*, combines the information without taking into account the shape of the distribution, and is calibrated via the σ parameter. Such enhanced exposition is shown in Fig. 4.

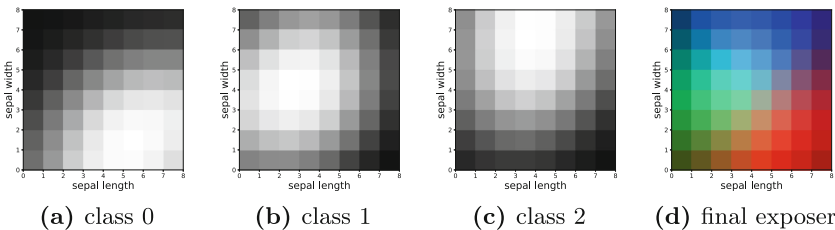


Fig. 4. The *Planar Exposer* enhanced by *anisotropic diffusion* and *Gaussian filter*

Established post-processing phase leads to the more general structure, describing not just the distribution of a training set, but also the approximate distribution of classes on whole available plane of given two-dimensional feature subspace.

2.2 Planar Exposer as a classifier

Each of the *Planar Exposer* layers, assigned to the n classes, is connected with each other by the same spatial dimensions, depending on the features on which it was described. Therefore, if we make the *exposition* of the training set, then for any test pattern we may determine the exact spatial location within the resulting structure, from which we can directly read n values meaning the local density of each class distribution. The *argmax* from this vector is the prediction of *Exposer*, enabling its usage as a classifier.

2.3 Approaches to Build an Exposer Ensemble

A *Planar Exposer*, despite the potential to use it as a classifier, limits its analysis to only two features of the dataset, so the natural move is to develop a method of information fusion from many such structures established on different subspaces of the feature space. An important element in each combined classifier is the assessment of the suitability of each member classifier for the committee's decision. So let's look at the examples presented on Fig. 5.

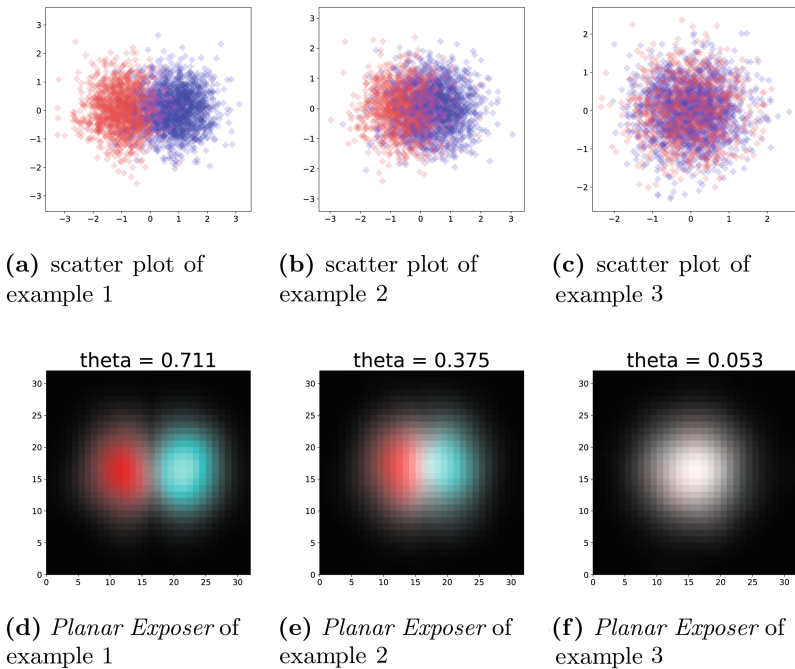


Fig. 5. Examples of scatter plots and Planar Exposers prepared for three datasets with different level of difficulty

The illustrations show the *Exposers* established for problems of varying difficulty. Example 1 shows a relatively easy problem in which the distributions of both classes overlap slightly, but most of them are disjoint, which leads to a structure, whose visual interpretation is characterized by high saturation. In Example 2 we have to deal with a more difficult problem, where overlapping distributions generate a white color at their contact, signifying the equal presence of objects of both classes in a given space element. Example 3 shows distributions almost completely overlapping, which leads to a completely desaturated display.

Each subspace of a dataset may be treated as a separate problem, which can be examined in terms of the degree of class distributions overlapping. So the measure θ was proposed, defined as the mean saturation value (understood as the difference between the maximum and minimum value in the point) of the *Exposer* elements with a maximum class intensity value greater than the threshold of 0.5. The larger it is, the higher is the information value of analyzed subspace. The θ values for the presented Examples have been written above the visualizations.

Bearing in mind the above property, three methods of constructing the pool of *Planar Exposers* were proposed:

- **Brute**, where we include all possible two-dimensional subspace in the pool,
- **Random**, where we include 16 randomly chosen two-dimensional subspaces in the pool,
- **Purified**, where we are establishing all possible *Planar Exposers*, but including only 16 of them with the highest θ measure.

2.4 Fusers

The last step necessary to construct a combined classifier from the pool of many *Planar Exposers* established on different dataset subspaces is to develop a rule for the combination of their answers to the final decision of the classifier ensemble. Two methods based on the accumulation of responses have been proposed:

- **Equal**, where we accumulate the class density vectors at the point of the test pattern taking the equal impact of each *Planar Exposer* in the pool on the decision,
- **Theta**, where we weigh the influence of each *Planar Exposer* on the decision, according to its θ measure.

3 Experimental Evaluation

3.1 Datasets

To experimentally evaluate proposition described in following paper, a collection of benchmark datasets from *KEEL Data Set Repository* [1] was used. Four groups of them were tested, from standard, balanced classification datasets, through imbalanced datasets with imbalance ratio lower than 9, to two groups of datasets

with imbalanced ratio greater than 9 [7]. In total, 77 problems were used (24 balanced, 14 with low imbalance ratio and 39 with high imbalance ratio), selecting both binary and multi-class datasets with only quantitative attributes. Overview of all datasets is available at KEEL repository website¹.

Remark 1. Data sets were also rejected due to too simple problems, in which almost all models, including *Exposer Ensemble*, gave 100% accuracy.

3.2 Design of Experiments

Implementation of *Exposer Ensemble* (EE) was prepared using Scikit-learn library [15]². Datasets from KEEL repository come with already prepared folds, which forced 5×2 CV as a method of dividing them into testing and training sets. To establish hyperparameters of proposed method, grid search was used. The hypothesis of statistical dependency was analyzed using Wilcoxon test. As the reference models, five popular classifiers with default hyperparameters were used:

- Multi Layer Perceptron (MLP),
- Naive Bayes (NB),
- Support Vector Machine (SVC),
- k-Nearest Neighbors (kNN),
- Random Forest (RF).

3.3 Results

Table 1 shows results of experiments conducted on balanced datasets, using the accuracy measure. Table 2 presents scores for imbalanced datasets with imbalance ratio lower than 9, using balanced accuracy score, more robust to such problems. Tables 3 and 4 shows results for imbalanced problems with imbalance ratio higher than 9.

Underline of score shows the highest quality achieved in pool of tested methods. Highlight emphasis the results statistically dependent to *Exposer Ensemble*. Its green color shows cases, where *Exposer Ensemble* was the best classifier in a pool, or was statistically dependent on the classifier with a highest score. Its blue color shows the other cases. Under each table is a summary of number of datasets where each tested classifier were the best or statistically dependent on the best.

Section *Exposer Ensemble* of the tables shows also the best configuration of proposed method for each dataset resulted from Grid Search.

¹ <http://sci2s.ugr.es/keel/datasets.php>.

² <https://github.com/w4k2/exposing>.

Table 1. Results obtained for balanced datasets.

Dataset	Exposer Ensemble						Reference models ACCs'				
	App.	Fuser	G	σ	A	ACC	RF	kNN	SVC	NB	MLP
<i>appendicitis</i>	brut	equal	32	4	0	0.877	0.887	0.868	0.812	0.858	0.859
<i>australian</i>	rand	equal	32	1	0	0.852	0.864	0.649	0.555	0.794	0.675
<i>balance</i>	brut	equal	16	4	0	0.654	0.813	0.830	0.901	0.893	0.952
<i>banana</i>	brut	equal	32	0	3	0.882	0.887	0.890	0.902	0.613	0.900
<i>bands</i>	brut	equal	32	1	1	0.710	0.705	0.630	0.627	0.483	0.606
<i>bupa</i>	rand	equal	32	2	3	0.658	0.681	0.670	0.591	0.554	0.733
<i>cleveland</i>	rand	equal	32	2	1	0.535	0.545	0.491	0.538	0.505	0.560
<i>contraceptive</i>	brut	equal	8	1	1	0.534	0.515	0.521	0.570	0.471	0.559
<i>ecoli</i>	puri	theta	16	2	2	0.827	0.809	0.816	0.426	0.601	0.822
<i>glass</i>	brut	theta	32	1	1	0.626	0.757	0.650	0.673	0.429	0.369
<i>heart</i>	rand	theta	16	2	2	0.804	0.789	0.674	0.556	0.841	0.819
<i>hepatitis</i>	rand	equal	32	3	1	0.821	0.815	0.802	0.834	0.656	0.820
<i>ionosphere</i>	brut	equal	32	1	1	0.880	0.926	0.843	0.935	0.895	0.923
<i>iris</i>	brut	equal	32	2	1	0.947	0.960	0.953	0.980	0.960	0.973
<i>led7digit</i>	brut	theta	8	1	1	0.734	0.700	0.722	0.728	0.610	0.738
<i>mammographic</i>	brut	theta	32	2	1	0.832	0.802	0.802	0.808	0.818	0.798
<i>monk-2</i>	brut	equal	16	1	1	0.974	0.988	0.988	0.972	0.919	0.924
<i>newthyroid</i>	brut	equal	32	4	2	0.958	0.958	0.930	0.749	0.963	0.884
<i>phoneme</i>	brut	equal	32	1	1	0.785	0.899	0.881	0.836	0.761	0.854
<i>ring</i>	brut	equal	8	2	1	0.847	0.934	0.687	0.505	0.980	0.771
<i>wdbc</i>	brut	theta	32	3	1	0.928	0.953	0.931	0.627	0.939	0.944
<i>wine</i>	brut	equal	16	3	2	0.972	0.966	0.691	0.438	0.983	0.713
<i>wisconsin</i>	brut	theta	8	1	0	0.975	0.962	0.975	0.963	0.962	0.962
<i>zoo</i>	brut	equal	8	3	1	0.950	0.950	0.861	0.931	0.960	0.960
# of best or statistically dependent on best						17	18	12	11	11	16

4 Critical Evaluation of Proposed Method

The results obtained by EE allow, with high probability, to consider it as a probabilistic pattern recognition method independent of the other considered solutions. In the case of the majority of analyzed datasets, it was included in the group of statistically the best classifiers.

The advantages of the proposed method include its approach to imbalanced data, making the decision of the classifier independent of the number of patterns from individual classes in the training set, deciding only on the basis of their probability distribution in the subspace.

The downside is the *Min-Max Scaler*, which, unlike the *Standard Scaler*, does not compensate for the impact of each of the features on the decision of the classifier. This approach may not be immune to outliers inside a training set, but filtering the established image of class distribution by anisotropic or Gaussian filter, makes highly likely resistance to the presence of noise.

Table 2. Results obtained for imbalanced datasets with imbalanced ratio lower than 9.

Dataset	Exposer Ensemble						Reference models BACS'				
	App.	Fuser	G	S	A	BAC	RF	kNN	SVC	NB	MLP
<i>ecoli-0-vs-1</i>	rand	theta	32	3	0	0.980	0.980	0.980	0.948	0.928	0.948
<i>ecoli1</i>	rand	theta	16	0	3	0.892	0.827	0.871	0.788	0.756	0.816
<i>ecoli2</i>	puri	theta	8	3	0	0.816	0.829	0.950	0.500	0.616	0.739
<i>ecoli3</i>	puri	equal	16	0	3	0.884	0.704	0.783	0.500	0.847	0.514
<i>glass-0-1-2-3-vs-4-5-6</i>	rand	theta	8	0	3	0.919	0.895	0.870	0.929	0.869	0.704
<i>glass0</i>	puri	equal	32	0	2	0.830	0.819	0.739	0.648	0.700	0.579
<i>glass1</i>	puri	equal	32	0	0	0.777	0.733	0.755	0.710	0.671	0.527
<i>glass6</i>	brut	equal	8	2	0	0.904	0.911	0.873	0.881	0.891	0.628
<i>new-thyroid1</i>	brut	equal	16	1	3	0.943	0.929	0.869	0.657	0.983	0.743
<i>new-thyroid2</i>	brut	equal	32	4	2	0.926	0.986	0.857	0.643	0.986	0.840
<i>pima</i>	rand	theta	32	3	0	0.715	0.749	0.721	0.651	0.756	0.677
<i>wisconsin</i>	brut	theta	8	1	0	0.975	0.962	0.975	0.963	0.962	0.962
<i>yeast1</i>	brut	theta	32	3	0	0.692	0.663	0.649	0.523	0.519	0.653
<i>yeast3</i>	brut	theta	8	2	1	0.914	0.814	0.828	0.500	0.605	0.750
# of best or statistically dependent on best						12	9	7	4	7	3

Table 3. First part of results obtained for imbalanced datasets with imbalanced ratio higher than 9.

Dataset	Exposer Ensemble						Reference models BACS'				
	App.	Fuser	G	S	A	BAC	RF	kNN	SVC	NB	MLP
<i>ecoli-0-1-3-7-vs-2-6</i>	rand	theta	8	4	3	0.845	0.550	0.850	0.500	0.825	0.700
<i>ecoli4</i>	brut	equal	16	4	0	0.937	0.850	0.848	0.500	0.878	0.500
<i>glass-0-1-6-vs-2</i>	rand	theta	32	2	1	0.748	0.553	0.555	0.500	0.580	0.500
<i>glass-0-1-6-vs-5</i>	brut	equal	16	0	2	0.940	0.750	0.739	0.500	0.941	0.500
<i>glass2</i>	rand	equal	16	4	3	0.636	0.500	0.485	0.500	0.591	0.500
<i>glass4</i>	brut	equal	32	3	4	0.963	0.692	0.781	0.728	0.587	0.533
<i>glass5</i>	puri	equal	16	0	0	0.887	0.748	0.695	0.500	0.938	0.500
<i>page-blocks-1-3-vs-4</i>	puri	theta	8	0	3	0.893	0.906	0.808	0.500	0.763	0.892
<i>shuttle-c2-vs-c4</i>	brut	equal	16	0	0	0.900	0.950	0.600	0.500	0.996	0.800
<i>vowel0</i>	brut	theta	32	4	4	0.955	0.955	0.977	0.983	0.917	0.999
<i>yeast-0-5-6-7-9-vs-4</i>	puri	theta	16	4	3	0.795	0.648	0.667	0.500	0.504	0.500
<i>yeast-1-2-8-9-vs-7</i>	rand	equal	16	3	0	0.701	0.564	0.499	0.500	0.544	0.500
<i>yeast-1-4-5-8-vs-7</i>	puri	theta	8	0	0	0.616	0.499	0.499	0.500	0.547	0.500
<i>yeast-1-vs-7</i>	brut	equal	16	2	0	0.727	0.546	0.517	0.500	0.604	0.500
<i>yeast-2-vs-4</i>	puri	equal	32	4	4	0.892	0.810	0.819	0.500	0.561	0.619
<i>yeast-2-vs-8</i>	brut	equal	8	4	2	0.779	0.650	0.774	0.725	0.657	0.725
<i>yeast4</i>	puri	equal	8	1	0	0.836	0.528	0.574	0.500	0.551	0.529
<i>yeast5</i>	brut	equal	16	4	4	0.963	0.723	0.850	0.500	0.831	0.644
<i>yeast6</i>	rand	equal	8	1	2	0.869	0.625	0.739	0.500	0.650	0.499
# of best or statistically dependent on best						18	8	8	2	10	5

Table 4. Second part of results obtained for imbalanced datasets with imbalanced ratio higher than 9.

Dataset	<i>Exposer Ensemble</i>						Reference models BACS'				
	App.	Fuser	G	S	A	BAC	RF	kNN	SVC	NB	MLP
<i>ecoli-0-1-4-6-vs-5</i>	brut	equal	32	2	0	<u>0.962</u>	0.746	0.898	0.500	0.877	0.771
<i>ecoli-0-1-4-7-vs-2-3-5-6</i>	rand	equal	8	0	1	0.846	0.813	0.847	0.500	0.630	<u>0.880</u>
<i>ecoli-0-1-4-7-vs-5-6</i>	rand	equal	8	0	1	<u>0.846</u>	0.798	0.838	0.500	0.735	0.777
<i>ecoli-0-1-vs-2-3-5</i>	puri	theta	16	1	4	<u>0.912</u>	0.763	0.830	0.500	0.638	0.578
<i>ecoli-0-1-vs-5</i>	brut	equal	8	1	0	<u>0.966</u>	0.850	0.900	0.500	0.782	0.568
<i>ecoli-0-2-3-4-vs-5</i>	puri	equal	8	1	0	<u>0.964</u>	0.800	0.894	0.500	0.754	0.692
<i>ecoli-0-2-6-7-vs-3-5</i>	brut	equal	8	0	1	<u>0.828</u>	0.808	0.787	0.500	0.563	0.700
<i>ecoli-0-3-4-6-vs-5</i>	puri	equal	8	1	1	<u>0.959</u>	0.820	0.875	0.500	0.784	0.822
<i>ecoli-0-3-4-7-vs-5-6</i>	brut	equal	16	0	2	0.839	0.838	<u>0.876</u>	0.500	0.775	0.578
<i>ecoli-0-3-4-vs-5</i>	puri	theta	8	1	1	<u>0.939</u>	0.850	0.875	0.500	0.817	0.669
<i>ecoli-0-4-6-vs-5</i>	rand	equal	16	1	3	<u>0.961</u>	0.897	0.900	0.500	0.854	0.814
<i>ecoli-0-6-7-vs-3-5</i>	puri	equal	8	0	0	0.835	<u>0.845</u>	0.835	0.500	0.508	0.765
<i>ecoli-0-6-7-vs-5</i>	rand	equal	8	1	1	<u>0.872</u>	0.847	0.847	0.500	0.780	0.645
<i>glass-0-1-4-6-vs-2</i>	rand	equal	8	0	1	<u>0.658</u>	0.553	0.512	0.500	0.577	0.500
<i>glass-0-1-5-vs-2</i>	brut	equal	32	2	0	<u>0.642</u>	0.554	0.527	0.500	0.519	0.497
<i>glass-0-4-vs-5</i>	puri	theta	16	2	1	0.901	<u>1.000</u>	0.850	0.600	0.994	0.650
<i>glass-0-6-vs-5</i>	puri	equal	16	0	0	0.830	0.895	0.745	0.600	<u>0.945</u>	0.500
<i>yeast-0-2-5-6-vs-3-7-8-9</i>	puri	theta	8	3	2	0.761	0.710	<u>0.762</u>	0.509	0.670	0.583
<i>yeast-0-2-5-7-9-vs-3-6-8</i>	puri	theta	8	1	2	0.861	0.847	<u>0.902</u>	0.500	0.577	0.681
<i>yeast-0-3-5-9-vs-7-8</i>	brut	theta	32	3	0	<u>0.687</u>	0.562	0.639	0.520	0.557	0.559
# of best or statistically dependent on best						20	13	19	1	9	11

The quantization of features makes the decision space discrete, which may negatively affect the quality of classification. In the future, the precise location of the tested pattern in the context of the *Exposer* should be considered, so also the values of neighboring cells should be taken into account during calculation of the support vector.

However, due to structure of *Planar Exposer*, discrete decision space is always stored in the memory, which makes EE a classifier with very favorable, constant computational complexity.

The random approach of ensemble construction can not be a stable solution, and a brutal overview of all possible subspaces leads to the storage of redundant information. Therefore, heuristic methods of choosing the optimal combination of subspace should be considered in the future.

5 Conclusion

This work presented *Exposer Ensemble*, being a combined classifier based on the original model of quantized subspace class distribution. The property of its representation, being resistance to the imbalance of training data were shown by

a series of computer experiments conducted on the collection of various datasets with diverse imbalance ratio. On the other hand, its quality in balanced datasets were comparable to the other popular classification models, showing no stable particular dependency to any of them.

The Exposer Ensemble is currently the work in progress solution, having the potential for solving both balanced and imbalanced classification problems. Its main future goal is to find application as well in streaming data. In other current works, its ability to regenerate data is tested, which may be used to balance datasets, whether static or streaming. On the other hand observation of saturation changes during the processing of subsequent stream chunks can be used also in the problem of drift detection of the concept.

Acknowledgment. The work was funded by the statutory funds of Department of Systems and Computer Networks (Faculty of Electronics, Wrocław University of Science and Technology) during realization of Młoda Kadra 2017/2018 task.

References

1. Alcalá-Fdez, J., et al.: Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *J. Mult.-Valued Log. Soft Comput.* **17**, 255–287 (2011)
2. Rish, I.: An empirical study of the naive Bayes classifier. In: *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence* (2001)
3. Bellman, R.E.: *Adaptive Control Processes: A Guided Tour*, vol. 2045. Princeton University Press, Princeton (2015)
4. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **16**, 321–357 (2002)
5. Dolph, C.V., Alam, M., Shboul, Z., Samad, M.D., Iftekharruddin, K.M.: Deep learning of texture and structural features for multiclass Alzheimer’s disease classification. In: *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2259–2266, May 2017. <https://doi.org/10.1109/IJCNN.2017.7966129>
6. Estabrooks, A., Jo, T., Japkowicz, N.: A multiple resampling method for learning from imbalanced data sets. *Comput. Intell.* **20**(1), 18–36 (2004)
7. Fernández, A., del Jesus, M.J., Herrera, F.: Hierarchical fuzzy rule based classification systems with genetic rule selection for imbalanced data-sets. *Int. J. Approx. Reason.* **50**(3), 561–577 (2009)
8. Gu, J., Jiao, L., Liu, F., Yang, S., Wang, R., Chen, P., Cui, Y., Xie, J., Zhang, Y.: Random subspace based ensemble sparse representation. *Pattern Recognit.* **74**, 544–555 (2018)
9. He, H., Bai, Y., Garcia, E.A., Li, S.: ADASYN: adaptive synthetic sampling approach for imbalanced learning. In: *2008 IEEE International Joint Conference on Neural Networks, IJCNN 2008 (IEEE World Congress on Computational Intelligence)*, pp. 1322–1328. IEEE (2008)
10. He, H., Garcia, E.A.: Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.* **9**, 1263–1284 (2008)
11. Kuncheva, L.: *Fuzzy Classifier Design*, vol. 49. Springer Science & Business Media, Heidelberg (2000). <https://doi.org/10.1007/978-3-7908-1850-5>

12. Kuncheva, L.I.: *Combining Pattern Classifiers: Methods and Algorithms*. John Wiley & Sons, Hoboken (2004)
13. Liu, B., Yu, X., Zhang, P., Yu, A., Fu, Q., Wei, X.: Supervised deep feature extraction for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **56**(4), 1909–1921 (2018). <https://doi.org/10.1109/TGRS.2017.2769673>
14. Mitchell, T.M., et al.: *Machine learning*. WCB (1997)
15. Pedregosa, F.: *Scikit-learn: machine learning in Python*. *J. Mach. Learn. Res.* **12**(Oct), 2825–2830 (2011)
16. Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach. Intell.* **12**(7), 629–639 (1990)
17. Ramentol, E., Caballero, Y., Bello, R., Herrera, F.: SMOTE-RSB*: a hybrid pre-processing approach based on oversampling and undersampling for high imbalanced data-sets using SMOTE and rough sets theory. *Knowl. Inf. Syst.* **33**(2), 245–265 (2012)
18. Romero, A., Gatta, C., Camps-Valls, G.: Unsupervised deep feature extraction for remote sensing image classification. *IEEE Trans. Geosci. Remote Sens.* **54**(3), 1349–1362 (2016). <https://doi.org/10.1109/TGRS.2015.2478379>
19. Yen, S.J., Lee, Y.S.: Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Syst. Appl.* **36**(3), 5718–5727 (2009)
20. Yin, F.L., Pan, X.Y., Liu, X.W., Liu, H.X.: Deep neural network language model research and application overview. In: 2015 12th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), pp. 55–60, December 2015. <https://doi.org/10.1109/ICCWAMTIP.2015.7493906>
21. Yousif, H., Yuan, J., Kays, R., He, Z.: Fast human-animal detection from highly cluttered camera-trap images using joint background modeling and deep learning classification. In: 2017 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1–4, May 2017. <https://doi.org/10.1109/ISCAS.2017.8050762>



A Framework for Form Applications that Use Machine Learning

Guilherme Aguiar^(✉) and Patrícia Vilain

Universidade Federal de Santa Catarina, Florianópolis, Brazil
guilherme.aguiar@posgrad.ufsc.br,
patricia.vilain@ufsc.br

Abstract. Machine Learning (ML) has been used efficiently in applications across multiple domains. As a consequence, there is a growing interest in ML techniques and artifacts that facilitate its use. However, most of them are aimed at researchers and experienced users. In addition, few artifacts provide more than ready-made algorithms. In this work, we present a framework capable of delivering ready-to-use ML algorithms, as well as the code to be reused by form applications that use ML algorithms. We also show an example where we used the framework to build two form applications. The results show that the framework is able to reduce approximately 50% of the effort when building a new application. In addition, the framework allows to include new state-of-the-art ML algorithms in an easy way, as well as it provides a simple flow control that assists inexperienced users in the use of ML algorithms.

Keywords: Machine learning · Framework · Form application
Form

1 Introduction

Machine Learning (ML) is a subfield of Computer Science that gives computers the ability to learn without being explicitly programmed [1]. ML induces results from previously known examples. Such examples are described by values for a set of attributes.

ML techniques are used in different domains. ML has been used in the estimation of effort for software development [2], as well as a support for the evaluation of the software development process [3]. Recommendation systems are also adopting ML in a predominant way, as demonstrated in [4]. ML is also presented in domains such as image classification [5], in economy [6], predicting the behavior of the financial market, and helping to optimize the business process decisions [7]. We can also find ML in many other areas such as health [8], improving user experience [9] and entertainment [10], for example.

Such versatility, along with the promising results, expanded the interest in ML by non-specialists. Programmers of all levels and researchers from various fields seek to use such techniques. The need for open source solutions to be adopted to diffuse the use of ML is highlighted by Sonnenburg et al. [11]. In fact, several initiatives have emerged to disseminate, facilitate the use and evolve the ML techniques.

Collobert et al. [12] proposed ready-to-use ML algorithms. Gashker [13] provided facilities varying from ready-to-use algorithms to a wizard used from command line. However, none of these libraries helps novice users to get involved with ML. Konkol [14] created a library of implemented learning algorithms; however, its major contribution was the sophisticated system of definition and management of attributes. This system helps users to provide their numerical data to the algorithms and also avoid much effort to model the data.

Kotthoff et al. [15] addressed this issue in a different way. They created a tool that facilitates the use of the WEKA, a widely used open source ML platform. This tool helps less experienced users to choose between learning algorithms and parameterize them.

Although these works facilitate the use of ML algorithms by non-specialists, the technologies mentioned so far do not provide code to be reused in the domain of form applications. Their efforts focus only in the implementation of ML algorithms and their parameterization. So, any form application to be created using such algorithms must be built from scratch.

In order to use ML algorithms, applications must be able to carry data consistent through variables used during their training. This data can be collected in different ways, but one simple and widely used way is the forms. These forms can contain fields of direct filling, or even be the interface of applications with more elaborate business rules. However, despite their simplicity, forms are an extremely efficient way of collecting such data.

This paper presents a framework that provides flow controls to guide the programmer during the creation of a ML logic. It reduces efforts during the implementation of applications that bring the data to the ML algorithms by filling forms. As it was designed using design patterns, the framework is easily extended. It also allows programmers to use ready-made libraries containing ML algorithms without much effort. Thus, any application that intends to use ML and that uses forms as input to supply the data to the trained algorithm can be easily produced using framework.

The rest of the paper is organized as follows: Sect. 2 presents the related works. Section 3 presents the proposed framework. Sections 4 and 5 present, respectively, the methodology and an example using the framework. Section 6 presents the threats to the validity. Finally, Sect. 5 presents the conclusions.

2 Related Works

There are several initiatives to produce technologies that bring more efficiency and productivity to the development of applications that use ML. Some produce ready-to-use implementations of known algorithms [12, 13], others adapt such algorithms to specific domains or programming languages [16], and some address their efforts to facilitate the understanding and application of these techniques.

Konkol [14] shows an example of artifact that promotes facilities for the use of ML techniques by non-specialists. Brainy was the name given to a library that brings an interface to the most popular ML algorithms. In addition, it has a complex mathematical infrastructure to support its sophisticated feature definition and management

system. According to the authors, such a system is the great scientific contribution of the research.

Bischl et al. [16] did something similar; however their focus is on a well-known tool called WEKA. WEKA is an open-source platform widely used by novices in ML due to its intuitive interface. However, because it has a wide range of algorithms in place, it is common for users to have difficulty to choose the best approaches to deal with their data set. So, Kotthoff et al. [15] defines the Auto-Weka 2.0, which assists users in this task.

Lauer [17] presents an open source software toolkit for ML on the Web. This tool is composed of three components. The first one is a JavaScript API for scientific computing. The second component is an extension of this library with ML algorithms. These two components deliver the benefits of offline processing and data privacy, since client-side processing removes the need to navigate the data over the Internet. The third component is an online development environment based on the components already described, with a user-friendly interface and examples of each function.

Costa et al. [18] defines a framework that combines ML techniques with the advantages of code reuse and effort reduction. The framework allows building web applications, aimed at mining blog information. It has two sub-frameworks, one to navigate and get data from blogs, and another dedicated to mining information from that data. In addition, it also has a third component, which starts the process of extracting information from blogs and provides an interface for the applications to request the services. This framework restricts its domain to e-commerce-driven WEB applications, and is designed to work with text rather than structured numerical data.

As we have seen, the vast majority of works focuses on providing ready-made algorithms, facilitating the parameterization and data modeling. However, nowadays, with the high availability of these algorithms, we also need technologies that are capable of facilitating other levels of using of ML algorithms.

The framework presented in this paper has the advantage of facilitating the application implementation. It allows users to have easy access to ready-made algorithms, a clear interface to use them, and ready code to be used in applications that uses these algorithms. Therefore, the user can reduce efforts in the development of applications that use ML.

3 Framework

According to Fayad and Schmidt [19], a framework is a semi-complete, reusable application that can be specialized to create specific applications and, as opposed to class libraries, is geared towards a specific application domain. Through these characteristics, frameworks promote benefits such as modularity, reuse, extensibility and inversion of control [19].

The framework presented in this paper is designed to provide advantages both in the use of ML techniques and in the creation of the applications that will use them.

After training a ML algorithm, the development of an application that uses ML may contain unexpected challenges for non-specialist users of such techniques. Bringing the formatted data to an algorithm interface and getting its results can be a daunting task.

Information that will be the input data of the algorithm must be collected and handled properly.

Forms are a simple and efficient way to gather information in applications. However, in spite of presenting such characteristics, the forms may represent more complex concepts than simple registers or fill-in fields for users. Applications can have your control flow and the input and output of all your data made through forms. In the case of ML applications, user-filled questionnaires, or data about some domains, when inserted into forms can easily be taken to trained algorithms if handled correctly.

This framework was developed to take data obtained through forms and used them as input to the ML algorithms. The framework fulfills its effort reduction approach through code reuse by providing ready code both in (1) the scope of ML algorithms and (2) the application flow control integrated with the use of forms.

The classes of the framework are divided into two packages as shown in Fig. 1.

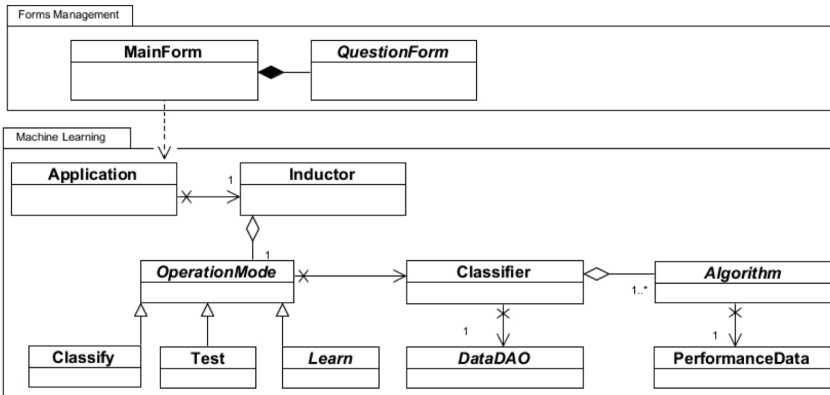


Fig. 1. Framework package diagram.

The Forms Management package contains the classes responsible for configuring and initializing the application and for managing application forms. The Machine Learning package contains all the classes that control the data and classes for the selected ML techniques. This package provides algorithms, data handling, and machine learning services for the Forms Management package.

3.1 Forms Management Package

An application that uses a trained ML algorithm requires the data created during its execution be consistent with the mass of data that was used to train such an algorithm since these data were not necessarily originated by it. One way to ensure this is keeping the forms consistent with the data loaded during the training phase of the algorithm. Besides to automatically create user-defined forms, the classes of the Forms Management package also propagate their settings to the Machine Learning package. Thus

it is possible to guarantee that the forms will be according to the data used by the algorithms.

In addition, this package relies on algorithms for component distribution and flow control of forms, reducing effort and bringing security during the creation of applications that manipulate them.

3.2 Machine Learning Package

The Machine Learning package contains the classes responsible for receiving form data that will be the input of the algorithms, as well as for the message flow control for it. The package has a class that is responsible for initializing the chosen ML technique and serves as a facade for the rest of the framework. All data traverse in the form of vectors and numerical arrays, creating a simple and unique interface, so new algorithms can be easily integrated into the framework.

The Machine Learning package also loads data to be used during the algorithm training. This is also a point where programmers have complete freedom to use the technology they want, or data persistence format. However, such data must be formatted following the definition of the framework, with matrices and numerical vectors. Thus, all the facilities for checking consistency and data exchange and performance verification will be done in a simple and automatic way according to the parameters defined by the users of the framework.

4 Methodology

The first step in the design a framework is the definition of the scope of the applications it will support. The framework presented in this paper is designed to encompass applications that combine two main characteristics: the use of ML and the collection of data from forms.

Some examples of applications that contain such characteristics are questionnaires that identify profiles from user responses. Vocational and personality tests are among the most common applications of this kind. One application of this kind will be described in Sect. 5.

Another possibility of great applicability of this framework is the creation of forms as interface to consult ML algorithms. Calderon-Vilca et al. [20] show a promising application of ML in the identification of suicidal tendencies. After the evaluation of results, and the definition of the ML algorithm, a desktop application was propose to determine if an adolescent has a suicidal tendency or not. This kind of application can be developed using the framework.

In addition to the scope of the applications, we also have to consider the user profile of a framework during its design. The main users of this framework are non-specialists in the use of ML. Often, such users may not be aware of features that are important to the applications, such as consistency between training and classification data, for example. Other issues are also addressed by the framework, such as the normalization of data and its adaptation to certain algorithms. However, considering that the concepts of control inversion and object orientation are extremely important to frameworks, it is

expected that users will have the necessary knowledge to properly handle these concepts.

The framework was codified using the Java language. This language was chosen because it has a great acceptance in the industry, besides being widely used in the academic area. In addition, important platforms which offer a huge variety of ML algorithms, like WEKA, are also codified using Java.

The framework was initially conceived from abstractions carried out by the authors. In order to construct its structure, we considered the domain and the scope of the applications addressed by the framework. Design patterns were thoroughly scrutinized in order to make the framework as reusable as possible, and to enable it to be extended more easily by users.

Then we created applications to improve the initial abstractions. After the creation of the applications, the lines of code of each one were analyzed to evaluate the reuse of code proposed by the framework. We counted the lines of code of each class of each application. After that, those lines that were already present in the code developed for the framework were considered reused. The remaining rows were considered new and also classified as reusable or not. Reusable lines represent those that implement features that could be present in more than one application, such as an algorithm that loads data from a spreadsheet.

The concept of reusable lines is useful since the framework contains concepts represented by abstract classes, which must be specialized in concrete classes by the applications created. An example is the Algorithm class, which represents ML algorithms and serves as interface with the rest of the application classes. It is expected that the programmer will specialize such a class to use the algorithms presented in others libraries. In this case, the specialized class can be reused by other applications that use the same algorithm.

Lines of code of the framework that are not considered reusable are usually related to the overwriting methods or the specialization of concrete classes that represent application characteristics. As an example, we can mention the checkAnswerRules method of the QuestionForm class. This method has as a function to verify the possible rules of filling the forms of the application. Such rules are characteristics of the application that hardly match those of another application.

In the case of class specialization, we can mention the subclasses of QuestionForm. These classes are responsible for representing the application forms. These forms generally consist of formatted fields, statements or descriptions, as well as other components that make these classes virtually unique to the applications.

5 Examples of Use

This section is dedicated to briefly illustrate two applications created during the development of the framework. Through the development of such applications, common requirements and concepts were generalized to be incorporated into the framework, such as a mechanism to facilitate bidirectional navigation between data entry forms.

The applications were chosen in order to demonstrate the larger possible number of characteristics of the framework. Thus, one application represents a problem of classification, while the other one represents a problem of regression. Likewise, completely different domains characteristics were chosen, showing the versatility and possibilities covered by the framework.

The WEKA platform was chosen as the supplier of the algorithms used in both applications. This decision was motivated by the wide acceptance of the WEKA in academic and industry areas. In addition, since it is an open source tool, WEKA is constantly updated with algorithms and techniques belonging to the state of the art.

In the following, both applications and their results are shown. The source code and documentation are available at <https://github.com/gui-aguiar/Ideal-Applications>.

5.1 Personality Assessment Test

The first application created was a test for personality assessment. There are numerous of such tests available for free online, and their format fit within the framework proposal. Therefore, one of these tests was chosen and reproduced using the framework.

As the logic to produce the personality diagnosis is not public, we used machine to simulate the application. It is common in the cases where a specialist indicates results from data, without a clear and explicit rule. In these cases ML techniques are used in an attempt to obtain a rule for generating results.

Thus, forty-eight simulations of the personality assessment test were done and their results were collected. These data were used as the basis for the training of the chosen algorithm. To be loaded by the application, such data were represented in csv format and loaded in such a way that they were transformed into vectors and numerical matrices.

Each test question consists in judging an affirmation and one answer the question choosing an option from seven levels according to his/her personality. The first level represents total agreement with the question while the seventh level represents total disagreement. At the end of the test the user receives a feedback on which personality type, among four possibilities, is most suitable to his/her profile.

Values from zero to seven were stipulated to represent each of the user's answer for each question. In addition, each personality category was represented as an integer from zero to three. In this way, the problem is defined as a classification problem.

Then, efforts were taken for the creation of the application. The forms were created according to the framework. With a simple design, the reuse of the navigation logic of the framework has proved of great value for the application.

As said before, the WEKA platform was used to provide the chosen ML algorithm. Through code integration, an implementation of logistic regression was used to classify the results. In this way, one can reduce effort in the use of ML techniques.

We have the following results about the first developed application. The total of 1340 lines of code was produced to create the application. Approximately 610 lines (45%) were part of the framework and, consequently, were reused. From the 730 new lines of code added to the creation of the application, 140 lines were incorporated into the framework. Thus, approximately 19% of the effort spent during the development of the first application was reused into the framework. This code refers to expected efforts,

such as the class `DataDOCSV` that loads data using a csv file, and can be reused by other applications. However, from the 730 new lines of code, 490 lines (74%) were not incorporated to the framework since this code basically refers to the creation and configuration of the application forms. Such forms are particularly relevant to the application and their reuse was not expected.

5.2 A Game for Managers

The second application created was a game aimed at production managers of companies. In this game, a player places himself/herself in the role of the production manager of a company. The player has to analyze a given scenario and distribute investments in some areas of the productive system of the company. The total amount of investments is stipulated initially and must be distributed according to the analysis of the player. Taking into account these investments, a final monetary value is calculated and returned as the result of the game. The higher this value, the more appropriate the investment actions taken by the player. The goal is to obtain the highest value.

In this game, the input and output values are numbers that represent monetary values, which clearly represents a regression problem.

The creation of this application follows the same sequence of the first application. The ML algorithm used in the application is from WEKA. As the game was initially designed by its creators using a spreadsheet, the CSV format was again chosen to represent the historical data of the application. This choice brought a complete reuse of the code created for this task in the first application.

As expected, during the development of the application, the main efforts were concentrated on the creation of the forms. In this application, 1146 lines of code were produced. From the total of 1146 lines, 610 lines of code are part of the framework, representing more than 53% of the application. From the remaining 536 lines of code, 50 lines were reused from a class created in the first application. This represents 9% of the code that was not reused from the framework; however if it was previously integrated to the framework it could be reused from the framework. In relation to the remaining 486 lines (approximately 42% of the entire application), 188 lines could also be incorporated into the framework, as the `DataDAOCSV` class, representing 35% of the new lines of code.

6 Threats to Validity

The framework has shown promising results, but some factors may threaten such results. The first one is that the author of the framework developed both applications used to measure such results. Thus, although it is expected that the programmer will be able to extract the maximum advantages and reuse of the framework some design decisions may have been influenced and aligned directly with the proposal of the framework, achieving the maximum reuse.

Another threat is related to the requirements of the developed applications. Although they are within the domain supported by the framework, its requirements do not exploit all the features provided by it. For example, both applications use only one

ML algorithm, but the framework predicts a flow of control capable of working with multiple algorithms simultaneously. Therefore, the reuse rate could be changed according to the application requirements chosen, as well as the algorithm used by it.

In addition, to use ML algorithms in the state of the art, the framework has been integrated only with WEKA. Integration with a distinct framework would inevitably change the results obtained.

7 Conclusions

The framework presented in this paper is capable of reducing effort and simplify the creation of form applications that uses ML. With an easy-to-maintain and simple-to-extend implementation, the framework allows to use third-party algorithms or algorithms implemented by the programmer. In the first case, if a chosen ML technique or library evolves, the user can easily update the application.

Another advantage obtained with the use of this framework is the predefined logical control of the application. Its inversion of control was designed to prevent errors that may occur when the application call the ML algorithms. This brings quality to the applications.

Finally, the framework reduces effort in the creation of forms that communicate with the classes that represents ML algorithms. Forms are a simple way to produce the data need to be consumed by ML algorithms, and are extremely versatile and can be used in various domains. The facilities imposed by the framework ensure that such forms will be in accordance with the data requested by the ML algorithms used, besides allowing the reuse of the code.

As future work we can mention the portability of the framework to other programming languages, the evolution of it through new applications produced, as well as the integration test with different ML artifacts.


References

1. Samuel, A.L.: Some studies in machine learning using the game of checkers. *IBM J. Res. Dev.* **44**(1.2), 206–226 (2000)
2. Wen, J., et al.: Systematic literature review of machine learning based software development effort estimation models. *Inf. Softw. Technol.* **54**(1), 41–59 (2012)
3. Chen, N., Hoi, S.C.H., Xiao, X.: Software process evaluation: a machine learning framework with application to defect management process. *Empir. Softw. Eng.* **19**, 1531–1564 (2013)
4. Portugal, I., Alencar, P., Cowan, D.: The use of machine learning algorithms in recommender systems: a systematic review. *Expert Syst. Appl.* **97**, 205–227 (2018)
5. Peng, Y., Yin, H.: Markov random field based convolutional neural networks for image classification. In: Yin, H., et al. (eds.) *IDEAL 2017*. LNCS, vol. 10585, pp. 387–396. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68935-7_42
6. Zhang, D., Zhou, L.: Discovering golden nuggets: data mining in financial application. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **34**(4), 513–522 (2004)
7. Apte, C.: The role of machine learning in business optimization. In: *Proceedings of the 27th International Conference on Machine Learning*, pp. 1–2 (2010)

8. Jiang, Y., Zhu, G., Lin, L.: Research of dengue fever prediction in san juan, puerto rico based on a KNN regression model. In: Yin, H., et al. (eds.) IDEAL 2017. LNCS, vol. 10585, pp. 146–153. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68935-7_17
9. Zhan, L., Zhang, J., Yang, Q., Lin, Y.: Applying random forest to drive recommendation. In: Yin, H., et al. (eds.) IDEAL 2017. LNCS, vol. 10585, pp. 470–480. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68935-7_51
10. Hsieh, J.-L., Sun, C.-T.: Building a player strategy model by analyzing replays of real-time strategy games. In: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence) (2008)
11. Sonnenburg, S., et al.: The need for open source software in machine learning. *J. Mach. Learn. Res.* **8**, 2443–2466 (2007)
12. Collobert, R., Bengio, S., Mariéthoz, J.: Torch: a modular machine learning software library. *Idiap* (2002)
13. Gashker, M.: Waffles: a machine learning toolkit. *J. Mach. Learn. Res.* **12**, 2383–2387 (2011)
14. Konkol, M.: Brainy: a machine learning library. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2014. LNCS (LNAI), vol. 8468, pp. 490–499. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07176-3_43
15. Kotthoff, L., Thornton, C., Hoos, H.H., Hutter, F., Leyton-Brown, K.: Auto-WEKA 2.0: automatic model selection and hyperparameter optimization in WEKA. *J. Mach. Learn. Res.* **18**, 1–5 (2013)
16. Bischl, B., et al.: Machine learning in R. *J. Mach. Learn. Res.* **17**, 1–5 (2016)
17. Lauer, F.: MLweb: a toolkit for machine learning on the web. *Neurocomputing.* **282**, 74–77 (2018)
18. Costa, E., et al.: A framework for building web mining applications in the world of blogs: a case study in product sentiment analysis. *Expert Syst. Appl.* **39**, 4813–4834 (2012)
19. Fayad, M., Schmidt, D.C.: Object-oriented application frameworks. *Commun. ACM* **40**(10), 32–38 (1997)
20. Calderon-Vilca, H.D., Wun-Rafael, W.I., Miranda-Loarte, R.: Simulation of suicide tendency by using machine learning. In: 2017 36th International Conference of the Chilean Computer Science Society (SCCC) (2017)



CGLAD: Using GLAD in Crowdsourced Large Datasets

Enrique G. Rodrigo¹, Juan A. Aledo², and Jose A. Gamez¹

¹ Computer Systems Department, Castilla-La Mancha University,
Albacete, Spain

{enrique.grodrigo, Jose.Gamez}@uclm.es

² Mathematics Department,
Castilla-La Mancha University, Albacete, Spain

JuanAngel.Aledo@uclm.es

Abstract. In this article, we propose an improvement over the GLAD algorithm that increases the efficiency and accuracy of the model when working on problems with large datasets. The GLAD algorithm allows practitioners to learn from instances labeled by multiple annotators, taking into account the quality of their annotations and the instance difficulty. However, due to the number of parameters of the model, it does not scale easily to solve problems with large datasets, especially when the execution time is limited. Our proposal, CGLAD, solves these problems using clustering from vectors coming from the factorization of the annotation matrix. This approach drastically reduces the number of parameters in the model, which makes using GLAD strategy for solving multiple annotators problems easier to use and more efficient.

Keywords: Non-standard classification · Crowdsourcing
Multiple annotators · Weakly supervised

1 Introduction

The GLAD algorithm [5] allows us to tackle problems of binary classification from multiple annotators (for example, those from crowdsourcing platforms) [7]. Learning from instances labeled by multiple annotators of unknown quality belongs to the set of non-standard classification problems [2], problems which differ from the typical classification problem in certain aspects. Specifically, when learning from multiple annotators, we do not have a true label for the instances of the training dataset. Instead, we receive a set of annotations of unknown quality for each instance as in Table 1.

Besides these annotations, it is possible to obtain features for each instance, although algorithms that take them into account [4] are not as common as

This work has been partially funded by the Spanish Research Agency (AEI) and FEDER (UE) through project TIN2016-77902-C3-1-P. Enrique G. Rodrigo has also been funded by the FPU scholarship FPU15/02281 by MECED.

© Springer Nature Switzerland AG 2018

H. Yin et al. (Eds.): IDEAL 2018, LNCS 11314, pp. 783–791, 2018.

https://doi.org/10.1007/978-3-030-03493-1_81

Table 1. Dataset of annotations

Annotator ₁	Annotator ₂	...	Annotator _N
0	0	...	1
1	1	...	-
0	-	...	0
1	0	...	1
1	-	...	1
...

algorithms that tackle the label aggregation process separately [8]. The most straightforward approach to estimate the true class from annotations is to use the most frequent class usually known as MajorityVoting. However different methods exist to estimate not only the labels but also the annotator quality and even the difficulties of instances given only the set of annotations [1,5]. This kind of algorithms is especially attractive when working with large unannotated datasets, as using experts is usually impractical.

In this article we expose some scalability problems of the GLAD algorithm, the main algorithm to use when one is interested in estimating the difficulty of the instances. In addition to this study, we propose an improvement to GLAD, CGLAD, which increases the size of problems that it can tackle. It also makes using the method easier by providing stability against changes in the parameters of the algorithm.

2 The GLAD Algorithm

The GLAD algorithm [5] allows us to estimate the true labels for instances labeled by multiple annotators, as well as the quality of these annotators and the difficulty of the instances, unlike other algorithms [1,3,4]. In this section, we introduce this algorithm, and its scalability problems.

2.1 Model

Apart from the true label, the annotation model depends on two elements:

- **Difficulty of each example**, represented as $1/\beta_i \in [0, \infty)$, for each example i , where β_i is positive. If $1/\beta_i$ is close to 0, the instance tends to be trivial to label while if it approaches ∞ , the example is so difficult that even experienced annotators have only a 50% to label correctly.
- **Annotator quality**, represented as $\alpha_j \in (-\infty, \infty)$, for each annotator j . If α_j approaches ∞ , the annotator always labels the instances correctly. If α_j is close to $-\infty$ the annotator always labels the instances incorrectly. This means that the annotator is as good as the previous one distinguishing between classes, but he is flipping the annotations (he may be adversarial or could

have misunderstood the instructions). If α_j is close to 0, the annotator cannot distinguish between the classes (random annotator or spammer).

The generative annotation model uses the last two parameters and obtains the probability that an annotator labels an instance correctly, using the following expression

$$c_{ji} = p(y_i^j = y_i | \alpha_j, \beta_i) = \frac{1}{1 + e^{-\alpha_j \beta_i}}$$

where y_i^j is the label given by annotator j to the instance i , and y_i is the true label for the instance.

2.2 Inference

The observable variables are the labels provided by the annotators. The hidden variables are the true labels and the parameters for the annotator (α) and the difficulty (β) models. The Expectation-Maximization (EM) algorithm can be used to obtain maximum likelihood estimates of the values of the hidden variables. First, the algorithm initializes the true label estimates by MajorityVoting. Then, this estimates (M step) are used to learn the parameters of the model (α and β). Lastly the new parameters are used to estimate the true label of the examples (E Step). The details of this algorithm can be seen in [5].

2.3 Scalability Problems

In this section, we provide results of the execution of the algorithm varying the number of instances of synthetic datasets. We also test different values for the learning rate of the gradient descent algorithm. This parameter seems to be the most related to the ability of the algorithm to converge, especially if we keep unaltered the gradient descent convergence threshold and the maximum number of gradient descent iterations. For the experimentation, we used synthetic datasets with the following features:

- **Dataset size.** We generated datasets of the following number of instances¹: 5K, 10K, 20K, 40K, 80K, 160K, 320K, 640K, 1.28M, 2.56M.
- **Annotations.** 10 annotations were simulated for each instance of the previous datasets. Each synthetic annotator is modeled using a discrete probability distribution (Fig. 1). Six of them are modeled with high accuracy (*good*), two are modeled as random annotators (*random*) and the last two as adversarial (*adversarial*).

We collect the accuracy obtained by the model (using learning rates: 0.1, 0.01, 0.001) for each dataset against the true label data. Since datasets are synthetic, this information is available to us, which would not be the case otherwise. The results are summarized in Fig. 2. We can observe that as the number of instances

¹ The datasets are available in `.csv` and `.parquet` format in the following link: <http://bit.ly/ideal2018-cglad>.

Class	Negative	Positive
Negative	0.8	0.2
Positive	0.1	0.9

(a) *good*

Class	Negative	Positive
Negative	0.5	0.5
Positive	0.5	0.5

(b) *random*

Class	Negative	Positive
Negative	0.2	0.8
Positive	0.8	0.2

(c) *adversarial*

Fig. 1. Annotator models

increases, the method fails to provide an accurate solution. As we alter the learning rate, we manage to tackle bigger datasets but more time is required, as more iterations are needed until convergence.

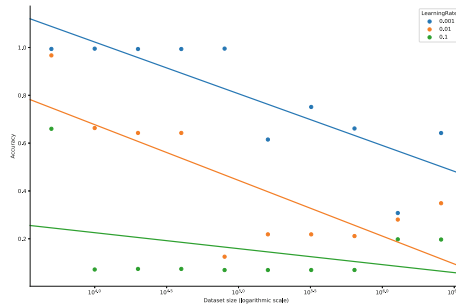


Fig. 2. Scalability problems of GLAD algorithm

We believe that the main reason for this problem is the increase in the number of parameters that have to be estimated. As all the datasets keep the same number of annotators, the number of parameters to be estimated for the annotators. However, as the number of instances of a dataset increases, the number of difficulty parameters increases linearly, as GLAD needs one parameter for each instance. As derived from Fig. 2, the gradient descent algorithm manages to converge for problems with a small number of instances but fails with larger ones. Therefore, it seems necessary to reduce the number of difficulty parameters to obtain a more stable algorithm (concerning the learning rate) and, mainly, to be able to use this method with more massive datasets.

3 CGLAD

In this section we describe the CGLAD algorithm, an improvement over the GLAD algorithm to reduce the number of parameters of the model. For this, we add an initial step previous to the EM algorithm that obtains clusters of similar instances, and then the difficulty of those clusters is used in the EM process. We explain the details of the full algorithm next.

3.1 Model

The model uses the same idea as GLAD, but it estimates difficulties for instance clusters instead of for every instance.

- **Cluster difficulty**, modeled using a parameter $1/\beta_t \in [0, \infty)$ for each cluster t , where β_t is positive. As was the case with GLAD, if $1/\beta_t$ is close to 0, the instances in *cluster* t are easier, while they are more complex as the parameter approaches ∞ .
- **Annotator quality**. It is represented as $\alpha_j \in (-\infty, \infty)$ for each annotator j , as it was the case in the GLAD algorithm.

The annotation model is similar to GLAD, but it uses difficulty parameters for each cluster. If we denote as $\phi(i)$ the function that maps the instance i to one of the clusters, the annotation model is

$$c_{ji} = p(y_i^j = y_i | \alpha_j, \beta_{\phi(i)}) = \frac{1}{1 + e^{-\alpha_j \beta_{\phi(i)}}}$$

where y_i^j is the label given by annotator j to instance i , and y_i is the true label of the instance. Except for the use of difficulties for each cluster, the interpretation of the expression is the same as in GLAD.

3.2 Inference

We use the EM approach to estimate the parameters of the model (before using EM, the cluster mapping is known).

- **E-step**: We call $\mathcal{Y}_i = \{y_i^j\}$ the set of annotations for the instance i (not every annotator labels all the instances). If we know α, β and \mathcal{Y}_i , we can determine the probability of the true label y_i using the following expression:

$$p(y_i = k | \mathcal{Y}_i, \alpha_j, \beta_{\phi(i)}) \propto p(y_i = k) \prod_j p_{ji}^k$$

where

$$\begin{aligned} p_{ji}^k &= p(y_i^j = k | y_i, \alpha_j, \beta_{\phi(i)}) \\ &= \begin{cases} (c_{ji})^k \cdot (1 - c_{ji})^{1-k} & \text{si } y_i = 1 \\ (c_{ji})^{1-k} \cdot (1 - c_{ji})^k & \text{si } y_i = 0 \end{cases} \end{aligned}$$

- **M-step**: The expression below is maximized using gradient descent with respect to the values of α and β

$$Q(\alpha, \beta) = \sum_i E[\ln(p(y_i = k))] + \sum_{ij} E[\ln(p_{ji}^k)].$$

where E is the expected value with respect to the estimations of the previous step.

3.3 Initialization

CGLAD proposes a special initialization that not only obtains a first estimation for the true labels (using majority voting) but also divides the instances into clusters, using matrix factorization and K-Means. This combination of methods allows the algorithm to obtain clusters of similar instances even when the features for the dataset are not available, only using the annotations.

- **Matrix factorization of the annotations dataset.** As occurs in collaborative filtering, we can visualize the dataset of annotations as a matrix where annotators are represented as rows and instances are represented as columns. Matrix factorization provides us with two matrices that allow us to estimate the matrix of annotations. This method gives us vectors of size R (a parameter that must be set by the user) for each instance (and also for the annotators) that capture differences between instances. For this step we used ALS [9] that can be used at scale. After the factorization, we obtain matrices A and D , that represent annotators and instances respectively.
- **Clustering.** Once the vectors are estimated, we can apply whichever clustering algorithm we choose over these vectors to obtain instance groups. In this proposal, we use K-Means, although other methods may also be applicable [6]. As our goal is not to interpret the clusters but to use them to reduce the number of parameters to be inferred, we can use a high number of clusters (the selection of the number of clusters depends on the problem to solve).

Once the previous steps are completed, we obtain the function ϕ , which is the last component needed to use the algorithm.

3.4 Scalability

We applied CGLAD to the same datasets that we used when analyzing the scalability problems of GLAD. We also used the same configuration for the gradient descent algorithm and EM algorithm. The results can be found in Fig. 3, where CGLAD shows a more stable performance across datasets and learning

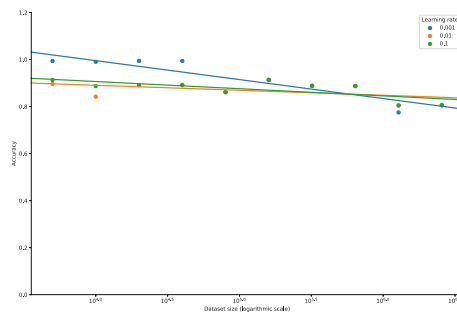


Fig. 3. Scalability of CGLAD in terms of quality

Table 2. Results from GLAD and CGLAD

(a) Accuracy and F-score					(b) Execution time		
Instances	Accuracy		F-score		Instances	CGLAD (Speedup)	GLAD
	CGLAD	GLAD	CGLAD	GLAD			
5000	0.9940	0.9940	0.9941	0.9941	5000	2797s (1.14)	3208s
10000	0.9904	0.9952	0.9903	0.9952	10000	1862s (2.03)	3781s
20000	0.9941	0.9937	0.9942	0.9938	20000	719s (5.52)	3973s
40000	0.9941	0.9937	0.9942	0.9938	40000	723s (5.43)	3925s
80000	0.8731	0.8521	0.8796	0.8520			
160000	0.9132	0.6151	0.9191	0.6153			
320000	0.8879	0.7513	0.8939	0.7516			
640000	0.8873	0.6613	0.8974	0.6566			
1280000	0.7750	0.3077	0.7500	0.3086			
2560000	0.8056	0.6423	0.7844	0.6418			

rate values. CGLAD seems to scale better to bigger datasets, and the difference in performance regarding accuracy seems negligible (we show further comparisons between the models in the next section).

4 Experimentation

In this section, we compare CGLAD against GLAD and also to other algorithms in the field with a similar goal.

4.1 CGLAD vs GLAD

We use the same datasets as in the previous sections to collect results about accuracy and execution time. As shown in Table 2, CGLAD obtains better results in most of the datasets, especially when their size tends to grow. The results were collected using the following configuration²:

- EM parameters: 5 iterations and 0.1 as threshold.
- Gradient descent: max. number of iterations 100, 0.1 as threshold and 0.001 as learning rate.
- Vector size: 8
- Cluster number: 32

It is also interesting to analyze the execution time for some of the datasets. Specifically, the first four, where GLAD and CGLAD obtain similar results³ (see Table 2).

Even though our proposal achieves a similar accuracy in the smaller cases, its execution time is highly reduced, as can be observed in Table 2.

² In this paper we did not use parameter tuning as this is not trivial to do in real datasets (because of the lack of ground truth) and also in order to obtain more general results.

³ In problems with large datasets GLAD fails to converge due to the high number of parameters to optimize, so it does not reach a good enough solution and the iteration process stops quickly. Because of that, CGLAD and GLAD cannot be fairly compared regarding execution time for bigger datasets.

4.2 CGLAD Against Other Algorithms

There exist other algorithms in this area able to estimate the ground truth labels from multiple annotators. However, we should take into account that these algorithms do not provide the user with the difficulties for each example. They do provide, usually, the quality of the annotators. Next, we compare our proposal, CGLAD, against two of these algorithms: majority voting, which could be considered as the baseline, as it supposes that all the annotators provide the same quality and the Dawid-Skene algorithm, which obtains state of the art results in most problems. The latter also uses the EM algorithm, but it only estimates the quality of the annotators using a discrete probability distribution. For this comparison, we use synthetic datasets in which annotations do not only depend on the instance class but also on a difficulty parameter (generated randomly). We use the following dataset sizes: 5000, 10000, 20000, 40000 y 80000. We adopt the following configuration for the algorithm:

- EM parameters: 5 iterations and 0.1 as threshold.
- Gradient descent: max. number of iterations 100, 0.1 as threshold and 0.0003 as learning rate.
- Vector size for factorization: 8
- Number of clusters: 32

CGLAD obtains comparable results (and even the best results) in several datasets as shown in Table 3. It also surpasses majority voting on every problem. However, CGLAD is not systematically better than Dawid-Skene, although this method is less complex. Anyway, CGLAD achieves comparable results and also allows the user not only to obtain the quality of the annotator but also the difficulty of the examples, which can be interesting in some problems.

Table 3. Comparative with other methods (Accuracy)

Instancias	Methods		
	MajorityVoting	DawidSkene	CGLAD
5000	0.8102	0.8462	0.8610
10000	0.8130	0.8443	0.8469
20000	0.8141	0.8478	0.8286
40000	0.8161	0.8494	0.8465
80000	0.8161	0.8494	0.8465

5 Conclusions and Future Work

In this article, we propose an improvement over the GLAD algorithm for the problem of learning from multiple annotators, CGLAD. We show through several experiments that our proposal achieves better scalability and stability when dealing with large datasets, improving upon not only GLAD execution time but

also its accuracy in several problems. As derived from the experimentation, for large datasets CGLAD is more robust against changes in the learning rate parameter, and it obtains better results overall, than the GLAD algorithm. Moreover, it obtains comparable results in smaller datasets. We have also compared CGLAD against other related algorithms. In this sense, our proposal obtains comparable results to these algorithms so it should be an algorithm to consider when solving multiple annotation problems. However, there exist other algorithms more accessible that should also be taken into consideration, and that obtain good results. CGLAD should especially be considered if we suspect that the difficulty of the instances plays an important role or if we are also interested in collecting estimations of the difficulty of the instances.

This improvement, as well as the analysis of the problems of GLAD, opens the door to future lines of work. First, in this article, we only consider simple algorithms in the initialization step, such as K-Means for clustering. There exist several algorithms that could be interesting to try for improving upon the results of the algorithm. Likewise, we could transfer the idea of estimating the difficulty of the examples to other models that can tackle classification problems with more than two classes or, even, regression problems. Lastly, we want to point out that this proposal does not make use of features to learn the cluster mapping. CGLAD could be extended to consider the features of the instances in problems where they are available, together with the annotations.

References

1. Dawid, A.P., Skene, A.M.: Maximum likelihood estimation of observer error-rates using the em algorithm. *Appl. Stat.* **2**, 20–28 (1979)
2. Hernández-González, J., Inza, I., Lozano, J.A.: Weak supervision and other non-standard classification problems: a taxonomy. *Pattern Recogn. Lett.* **69**, 49–55 (2016)
3. Li, Q., Li, Y., Gao, J., Zhao, B., Fan, W., Han, J.: Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In: *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, pp. 1187–1198. ACM (2014)
4. Raykar, V.C., et al.: Learning from crowds. *J. Mach. Learn. Res.* **11**(Apr), 1297–1322 (2010)
5. Whitehill, J., Wu, T., Bergsma, J., Movellan, J.R., Ruvolo, P.L.: Whose vote should count more: optimal integration of labels from labelers of unknown expertise. In: *Advances in Neural Information Processing Systems*, pp. 2035–2043 (2009)
6. Xu, R., Wunsch, D.: Survey of clustering algorithms. *IEEE Trans. Neural Netw.* **16**(3), 645–678 (2005)
7. Zhang, J., Wu, X., Sheng, V.S.: Learning from crowdsourced labeled data: a survey. *Artif. Intell. Rev.* **46**(4), 543–576 (2016)
8. Zheng, Y., Li, G., Li, Y., Shan, C., Cheng, R.: Truth inference in crowdsourcing: is the problem solved? *Proc. VLDB Endow.* **10**(5), 541–552 (2017)
9. Zhou, Y., Wilkinson, D., Schreiber, R., Pan, R.: Large-scale parallel collaborative filtering for the netflix prize. In: *Fleischer, R., Xu, J. (eds.) AAIM 2008. LNCS, vol. 5034*, pp. 337–348. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-68880-8_32



Extending Independent Component Analysis for Event Detection on Online Social Media

Hoang Long Nguyen and Jason J. Jung^(✉)

Department of Computer Engineering, Chung-Ang University,
84 Heukseok-ro, Dongjak-gu, Seoul, Korea
longnh238@gmail.com, jjjung@gmail.com

Abstract. In this paper, we propose a new approach for filtering noises from social event signals by using independent component analysis technique. With a case study about two practical events, we prove that our approach is feasible. Besides, this idea is also to acquire high adaptability and extensibility property because we use only temporal signals of social events as inputs without requiring any more information. Moreover, our approach can be used as pre-processing step for other signal-based event discovering algorithms.

Keywords: Social event · Independent component analysis
Noise reduction · Signal separation

1 Introduction

Social networking service (SNS) is an effective web application that allows users to interact with each other and share digital contents [5]. This explains why SNS is becoming so popular among Internet services. Whenever there is an event happen in our society, users promptly react by distributing it on SNSs. By gaining convenience from smart devices, social media contents are published expeditiously and accurately. One of the most representative examples of SNSs is Twitter. Twitter is currently one of popular SNSs that allows users to post their statuses (which is called as tweets) from various services (e.g., website, application, and sharing from other services by using Twitter API) [9]. Although tweets are limited in size with 140 characters, Twitter is updated with more than 8,000 tweets per second¹ from more than 300 million users as the statistics in June 2018. Besides, its content is very diverse and can give us an overview of what is happening around us. Various kinds of models have been studied to discover social events [1, 4, 14]. They can be supervised, unsupervised, or even hybrid approaches.

¹ <http://www.internetlivestats.com/>.

A hashtag is a keyword which is started by # and is used to describe topic on Twitter (e.g., #WorldCupRussia2018 mentions about a quadrennial international football tournament at Russia in 2018). Twitter users put hashtags in their tweet in order to classify them and make it easy for other users to follow a specific topic. Therefore, it is a powerful feature that enables researchers to discover events on Twitter [2]. In order to create effective hashtags for an event, they should be unique, related, and memorable. However, data which is collected by using hashtag contains a lot of noise and even inaccurate or counter-factual information because it is self-generated by users. People may use hashtags which do not concentrate on a particular topic. For example, instead of directly using hashtag #WorldCupRussia2018, some users use only general hashtag #Football to mention about this sport event. Besides, #Football is also an ambiguous hashtag because it represents two different sports. Moreover, user also uses many hashtags in a single tweet to get more exposure. This leads to the fact that some hashtags are not relevant to what you are tweeting about. This causes uncomfortable to users because they have to receive much irrelevant information.

Our target in this paper is to reduce the influence of noise tweets by applying Independent Component Analysis (ICA) algorithm. There is also a traditional filtering technique which is Principal Component Analysis (PCA). However, it is only able to process on data with (i) small noise level and (ii) signal and noise subspace are orthogonal [13]. ICA is a proficient method to overcome these aforementioned problems and was applied to reduce noise in EEG [8], image [3], or speech [7]. Nevertheless, applying ICA to handle social media data is still a new idea. This could be considered as our contribution in this research as well.

The remainder of this paper is organized in the following manners. In Sect. 2 we define our main problem. Further, Sect. 3 mentions about ICA method to reduce noise which is caused by Twitter hashtags. Next, we discuss about a case study in Sect. 4 for expressing about the utility of this research. Finally, we conclude and state some future works in Sect. 5.

2 Problem Definition

This research attempts to filter out the noise from social events on Twitter which is caused by failure hashtags. Although various studies focus on utilizing hashtags but few of them realize the occurrence of misunderstanding hashtags. In this section, we clearly define the problem and give a formalization of importance definitions.

Definition 1 (Event). *An event E is a particular action which happens in our society. It includes a collection of tweets Ψ which mention a topic Θ in a duration of time Υ . It can be formulated as:*

$$E = \langle \Psi, \Theta, \Upsilon \rangle \quad (1)$$

$$\Psi = \langle \psi_1, \dots, \psi_i, \dots, \psi_n \rangle \quad (2)$$

$$\Upsilon = [\varepsilon_s, \varepsilon_e] \quad (3)$$

where ψ_i is used to denote for an i -th tweet that belongs to event E , ε_s is the starting time of event E and ε_e is the ending time of this event.

Each tweet ψ_i is featured by a sequence of words W_i and social attribute A_i . A tweet may contain lots of information, however, the triple of *topic*, *location*, and *time* are essential attributes for representing social events.

$$\psi_i = \langle W_i, A_i \rangle \quad (4)$$

$$W_i = \{w_i \mid w_i \in \mathbb{T} \text{ or } w_i \in \mathbb{H}\} \quad (5)$$

$$A_i = \langle \tau_i, \lambda_i, \varepsilon_i \mid \tau_i \in \Theta, \varepsilon_i \in \Upsilon \rangle \quad (6)$$

where w_i denotes for a word that belongs to tweet ψ_i , w_i can be a normal term \mathbb{T} or a hashtag \mathbb{H} . $\tau_i, \lambda_i, \varepsilon_i$ are topical, spatial, temporal attribute respectively of tweet ψ_i .

Topic τ_i of tweet ψ_i is indicated by the set of hashtags that belongs to tweet ψ_i . In spite of the fact that normal term can be used to represent a topic, we utilize the expressing topic ability of hashtag in this paper.

$$\tau_i = \{w_i \mid w_i \in W_i \text{ and } w_i \in \mathbb{H}\} \quad (7)$$

Definition 2 (Topic of Event). *Topic of event Θ is specified by the set of hashtags which is exploited from the set of tweet Ψ .*

$$\Theta = \{w_i \mid w_i \in \Psi \text{ and } w_i \in \mathbb{H}\} \quad (8)$$

However, the set of hashtags between two events is always overlapped because they are self-generated by users. In this paper, we focus on reducing the noise of two events E_a and E_b which have similar topic and happen at similar time because they will have higher probability to create noise to each other. Because people may use hashtags which belong to both events, we will get some failure tweets from event E_b when we collect data for event E_a and vice versa.

Definition 3 (Event Noise). *Event noise is set of tweets that belongs to both event E_a and E_b because of the overlap hashtags between topics of two events.*

$$N_a = \{\psi_a \mid \exists w_a, w_a \in \Theta_a \cap \Theta_b \text{ and } w_a \in \mathbb{H}\} \quad (9)$$

where N_a denotes as noise of event E_a , ψ_a is a tweet of event E_a , w_a is a hashtag that belongs to tweet ψ_a , Θ_a and Θ_b are topics of event E_a and event E_b respectively.

Our task in this paper is to minimize the noise of two events N_a and N_b by applying the idea which will be explained in detail in Sect. 3.

3 Applying Independent Component Analysis for Reducing Social Event Noise

In this section, we explain in detail about our proposing method for reducing noise in two events E_a and E_b which have similar topic and happen at similar time. The noise is caused by using misunderstanding hashtags as we mentioned in previous section. Our algorithm includes two steps: (i) determining the temporal distribution of tweets and (ii) applying ICA for separating observed temporal signals in step (i) for obtaining source signals.

ICA is a famous algorithm to solve the blind source separation problem (i.e., estimating originals signal from observed signals including the mixing between original signals and noise). In this research, our observed signals is the tweet distribution over time which is calculated from above step (1). For the sake of simplicity, we propose general formula and each event can be effortlessly formulated by applying this one.

Giving the set of tweets Ψ which belongs to event E , we first need to compute the distribution of tweet for an event. We divide the event time \mathcal{T} as time windows and the number of tweets in each time windows is used as value at each sample point of the time series. Temporal signal of an event can be written as a sequence:

$$x(t) = \{x_{t_1}, x_{t_2}, \dots, x_{t_n}\} \tag{10}$$

The value of x_{t_i} at each sample point is computed as follows.

$$x_{t_i} = \{N_{t_i}(\psi) \mid \forall \varepsilon \in [t_{i_s}, t_{i_e}]\} \tag{11}$$

where $N_{t_i}(\psi)$ is the number of tweets at sample point t_i , ε is published time of tweets belongs to sample point t_i , and t_{i_s} , t_{i_e} and starting time, ending time of time windows correspondence with t_i .

The temporal signals of two events $x_a(t)$ and $x_b(t)$ are used as input for ICA algorithm which is proposed by Jutten and Herault [12]. This method uses statistical independence as the principle for separating sources without knowing mixing coefficients. The criterion of ICA is to find a transformation that can minimize mutual information between elements. Therefore, it is a novel method for eliminating noise from signals. Figure 1 represents the principle of ICA.

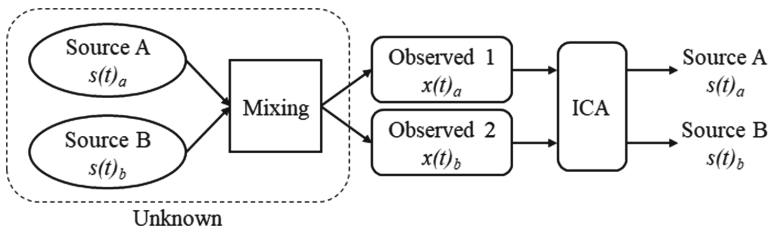


Fig. 1. ICA modeling.

Returning back to our problem, what we have are only two observed signals $x(t)_a$ and $x(t)_b$. They are temporal signals which consist of original signal and noise of two event E_a and event E_b respectively. In order to apply ICA, the length of $x(t)_a$ and $x(t)_b$ must be equal. The relationship between observed signals and source signals can be expressed as follows.

$$\begin{aligned} x(t)_a &= a_{11}s(t)_a + a_{12}s(t)_b \\ x(t)_b &= a_{21}s(t)_a + a_{22}s(t)_b \end{aligned} \tag{12}$$

where $s(t)_a$ and $s(t)_b$ are original signals, and a_{11} , a_{12} , a_{21} , a_{22} are mixing coefficients.

We assume that $s(t)_a$ and $s(t)_b$ are statistically independent of each other. Let's denote A as the mixing matrix with size 2×2 as follows.

$$A = \begin{bmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \end{bmatrix} \tag{13}$$

For the sake of simplicity, Eq. (12) can be re-written as:

$$x = As \tag{14}$$

In order to solve the ICA problem, we need to estimate A and using x . After obtaining matrix A , we can compute matrix W which is the inverse of matrix A . Then, s is calculated easily by:

$$s = Wx \text{ with } A = W^T \tag{15}$$

There are various methods to estimate matrix A , however, we select Fast ICA algorithm due to its computationally efficient and less memory requirement. To apply Fast ICA, observed signals first need to be pre-processed by whitening to reduce the problem complexity. Eigen-value decomposition (EVD) is an effective whitening algorithm by using eigen values and eigen vectors of the covariance matrix $E(xx^T)$ for creating whitening matrix Q as follows.

$$Q = ED^{-1/2}E^T \tag{16}$$

where E is eigen vector matrix of $E(xx^T)$ and D is diagonal matrix of eigen values.

After whitening, we acquire new ICA task as following formula.

$$y = Qx = ED^{-1/2}E^TAs = A's \tag{17}$$

$$E(A'A^T) = I \tag{18}$$

where y is whitened data.

Fast ICA is aimed to find the maximization of non-gaussianity of w^Ty which can be based on kurtosis, negentropy, mutual information, and so on. In this paper, we consider to maximize the approximations of negentropy $J(w^Ty)$

because it is simple and fast computation [6]. The approximations of negentropy function is as follows.

$$J(z) \approx [E\{G(z)\} - E\{G(v)\}]^2 \quad (19)$$

where v is a Gaussian random variable with zero mean and unit variance, G denotes for quadratic functions.

Non-quadratic function G can be belongs to the possible choices:

$$\begin{aligned} G_z &= \frac{1}{a_1} \log \cosh(a_1 z) \\ G_z &= -\exp(z^2/2) \end{aligned} \quad (20)$$

where a_1 is an appropriate constant with $1 \leq a_1 \leq 2$.

Our object is to maximize the below function:

$$J(w^T y) = [E\{G(w^T y)\} - E\{G(v)\}]^2 \quad (21)$$

To maximize function (21), we need to find direction of weight vector w in order for projection $w^T y$ maximizes nongaussianity. We conduct the Fast ICA algorithm as follows [11].

1. Initializing a random value for weight vector w .
2. Setting $w^+ = E\{yg(w^T y)\} - E\{g'(w^T y)\}w$ with $g(z) = \tanh(z)$ and $g(z) = \exp(-z^2/2)$.
3. Normalizing w by calculating $w = w^+ / \|w^+\|$.
4. Returning back to step 2 if not converged (i.e., convergent means current value and previous value of w in same direction. This can be fomulated as $|w^+ - w| > \epsilon$).
5. Output is the final weight vector w . This is a signal which is separated: $s_i = w^T .y_i$, $i = (1, 2, \dots, n)$ with n is the number of independent components.

In our problem, we need to estimate two independent components. Hence, the above algorithm must be run two times for obtaining two separated signals.

4 Case Study with Summit Event

This section aims to present the case study to explain about our proposal in a practical scenario. We select two summit events which are the G7 summit and the Trump-Kim summit as our case study. The topics of these two events are similar to each other. Both events are all related to an international meeting between heads of government. Besides, they are also organized at contiguous time (i.e., 9th June 2018 with the G7 summit event and 12th June 2018 with the Trump-Kim summit event).

We first obtained the dataset by using our SocioScope framework [10]. This system supports crawling streaming data from Twitter by passing the list of keywords. In this paper, we do not focus on how to determine top hashtags

of an event. We collected 288,121 tweets related to the G7 summit event and 1,191,536 tweets related to the Trump-Kim summit event. This result shows that people feel more interested in the Trump-Kim summit rather than the G7 summit. It is very interesting that there exists some hashtag which belongs to both events (i.e., #trumpkimsummit, #trump, #summit, #presidenttrump, #maga) because these two events share similar topic. Because the keyword #summit is too general, some data that contains this hashtag is about the G7 summit, some other is about the Trump-Kim summit. We therefore found it very difficult to categorize and to obtain the information that we need.

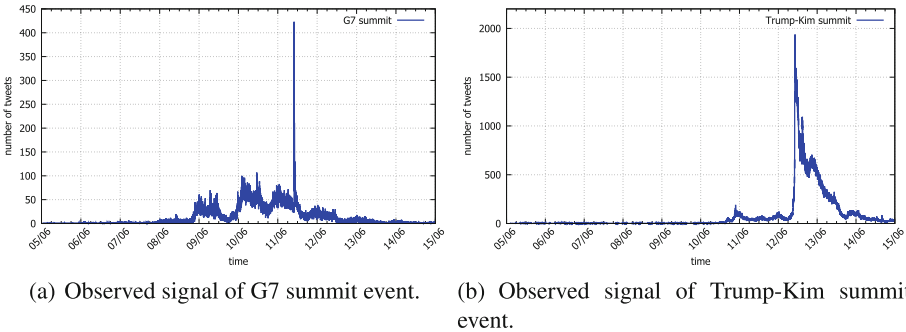


Fig. 2. Observed signal of two summit events.

Let's assume that we have a group of users A that is interested in the G7 summit event and they want to share their consideration by posting tweets to Twitter. Similarly, a group of users B pays attention to Trump-Kim summit and has same posting behavior with a group of users A. Nevertheless, there are overlap hashtags as we mentioned. Due to this reason, the set of tweets about the G7 summit contains tweets (which are considered as noise) about the Trump-Kim summit and vice versa.

Figure 2(a) and (b) represent about the distribution of tweet of two events. It is necessary to emphasize that what we need to use as inputs for our proposed algorithm are only temporal signals of two events. This proves for the high adaptability of our method. In case of losing all other information except temporal signals (i.e., distribution of tweet by time), our proposed method still well performs. After conducting the ICA on observed signals, we got filtered signals as shown in Fig. 3(a) and (b). The noise is eliminated at many points, however, we recognize that it is higher from 11th June 2018 to 12th June 2018. This is pretty reasonable because it is the time between two events, when G7 summit has just finished and Trump-Kim summit is going to happen. At this period, users tend to mention about both events simultaneously.

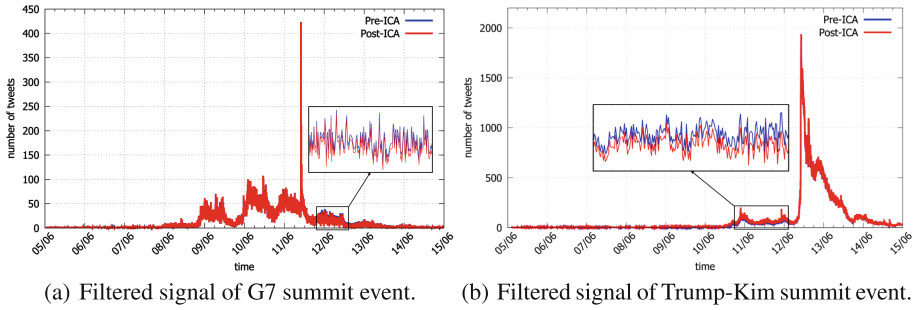


Fig. 3. Experiment result.

5 Conclusion and Future Work

In this paper, we proposed the idea of reducing noise of social events by applying ICA algorithm. With a case study of summit events on Twitter, we demonstrated the feasibility of our research. Besides, this study also aims to open a new idea for applying ICA to handle social media data in general, not only focus on social event. For the future work, we plan to research about applying ICA for frequency domain to solve the problem. Time domain signal is difficult to do more deeper analysis compared to frequency domain. In addition, frequency analysis is very effective for filtering the noise from signals.

Further, we are also interested in studying about separating single-channel signal because we only can obtain single signal in some situations. To adapt with this issue, our target is to propose an algorithm that: *(i)* must support separating single-channel data signal to original signals for increasing the generality, *(ii)* can be able to cope with non-stationary signal because social data is not stable in a duration of time, and *(iii)* be unsupervised approach since no dataset can be obtained for training.

Acknowledgment. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (NRF-2017R1A4A1015675, NRF-2018K1A-3A1A38056595).

References

1. Becker, H., Naaman, M., Gravano, L.: Selecting quality twitter content for events. In: Proceedings of the 5th International Conference on Weblogs and Social Media (ICWSM 2011), Barcelona, Catalonia, Spain, 17–21 July 2011, pp. 442–445. The AAAI Press (2011)
2. Cui, A., Zhang, M., Liu, Y., Ma, S., Zhang, K.: Discover breaking events with popular hashtags in twitter. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, Maui, HI, USA, 29 October–02 November 2012, pp. 1794–1798. ACM (2012)

3. Ding, W.: A new method for image noise removal using chaos-PSO and nonlinear ICA. *Procedia Eng.* **24**, 111–115 (2011)
4. Fung, G.P.C., Yu, J.X., Yu, P.S., Lu, H.: Parameter free bursty events detection in text streams. In: *Proceedings of the 31st International Conference on Very Large Data Bases*, Trondheim, Norway, 30 August–2 September 2005, pp. 181–192. ACM (2005)
5. Hoang Long, N., Jung, J.J.: Privacy-aware framework for matching online social identities in multiple social networking services. *Cybern. Syst.* **46**(1–2), 69–83 (2015)
6. Hyvärinen, A., Oja, E.: Independent component analysis: algorithms and applications. *Neural Netw.* **13**(4–5), 411–430 (2000)
7. Kim, C.M., Park, H.M., Kim, T., Choi, Y.K., Lee, S.Y.: FPGA implementation of ICA algorithm for blind signal separation and adaptive noise canceling. *IEEE Trans. Neural Netw.* **14**(5), 1038–1046 (2003)
8. Li, Y., Ma, Z., Lu, W., Li, Y.: Automatic removal of the eye blink artifact from EEG using an ICA-based template matching approach. *Physiol. Meas.* **27**(4), 425 (2006)
9. Nguyen, H.L., Jung, J.E.: Statistical approach for figurative sentiment analysis on social networking services: a case study on twitter. *Multimedia Tools Appl.* **76**(6), 8901–8914 (2017)
10. Nguyen, H.L., Jung, J.E.: Socioscope: a framework for understanding internet of social knowledge. *Future Gener. Comput. Syst.* **83**, 358–365 (2018)
11. Patil, D., Das, N., Routray, A.: Implementation of fast-ICA: a performance based comparison between floating point and fixed point DSP platform. *Meas. Sci. Rev.* **11**(4), 118–124 (2011)
12. Peng, H., Zhu, S.: Handling of incomplete data sets using ICA and SOM in data mining. *Neural Comput. Appl.* **16**(2), 167–172 (2007)
13. Vorobyov, S., Cichocki, A.: Blind noise reduction for multisensory signals using ICA and subspace filtering, with application to EEG analysis. *Biol. Cybern.* **86**(4), 293–303 (2002)
14. Yang, Y., Pierce, T., Carbonell, J.: A study of retrospective and on-line event detection. In: *Proceedings of the 21st International Conference on Research and Development in Information Retrieval*, Melbourne, Australia, 24–28 August 1998, pp. 28–36. ACM (1998)



Framework for the Training of Deep Neural Networks in TensorFlow Using Metaheuristics

Julián Muñoz-Ordóñez¹, Carlos Cobos¹✉, Martha Mendoza¹, Enrique Herrera-Viedma², Francisco Herrera², and Siham Tabik²

¹ Information Technology Research Group (GTI), University of Cauca, Popayán, Colombia

{julianfer, ccobos, mmendoza}@unicauca.edu.co

² Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain

{viedma, herrera}@decsai.ugr.es, siham@ugr.es

Abstract. Artificial neural networks (ANN) again are playing a leading role in machine learning, especially in classification and regression processes, due to the emergence of deep learning (ANNs with more than four hidden layers), allowing them to encode more and more complex features. The increase in the number of hidden layers in ANNs has posed important challenges in their training. Variations (e.g. RMSProp) of classical algorithms such as backpropagation with its stochastic gradient descent are the state of the art for training deep ANNs. However, other research has shown that the advantages of metaheuristics need more detailed study in this area. We summarize the design and use of a framework to optimize learning of deep neural networks in TensorFlow using metaheuristics, a framework implemented in Python that allows training of the networks in CPU or GPU depending on the TensorFlow configuration and allows easy integration of diverse classification and regression problems solved with different neural networks architectures (conventional, convolutional and recurrent) and new metaheuristics. The framework initially includes Particle Swarm Optimization, Global-best Harmony Search, and Differential Evolution. It further enables the conversion of metaheuristics into memetic algorithms including exploitation processes using the algorithms available in TensorFlow: RMSProp, Adam, Adadelta, Momentum, and Adagrad.

Keywords: Deep learning · Neuroevolution · TensorFlow · Metaheuristic Framework

1 Introduction

Artificial neural networks (ANN) have again acquired a leading role in the area of machine learning, especially in classification and regression processes, due to the emergence of deep learning [1]. The “deep” quality in ANNs is acquired increasing the hidden layers in their architecture, allowing them to encode increasingly complex features. Deep learning architectures are based on a series of approaches [2, 3] that solve problems related to image classification (Deep Convolutional Neural Networks),

and analysis and recognition of text and voice (Deep Recurrent Neural Networks), among others.

Historically, ANNs are trained using the backpropagation algorithm (BP) that makes use of stochastic gradient descent (SGD) to adjust weights of neurons in the training process [1], reducing total error rate. A disadvantage in the learning process of a deep neural network (DNN) is SGD stagnation due to the high dimensionality produced by the increase in connections and weights to be defined in the network. Reduction of the error rate is slowed, with consequent low values of accuracy and/or precision in classification problems, or high error values in regression problems.

Although SGD was initially thought to be affected by the multiple local optima in the high dimensionality, multiple escape routes were shown to benefit the SGD, but there are also areas where the gradient becomes zero (saddle point) meaning that the SGD cannot find an escape route and stagnates [4]. Algorithms such as RMSProp [5], have been proposed to prevent SGD being trapped in saddle points.

Metaheuristic algorithms have furthermore shown advantages in solving complex discrete problems and binary or continuous problems of high dimensionality. Previous studies show metaheuristic advantages over SGD in training DNNs [1]. A framework for optimizing the learning of DNNs was thus designed through modifying weights and biases of the different layers that make up the network using various metaheuristics, DNN architectures and datasets or problems.

The rest of the document is organized as follows: Sect. 2 presents related prior work; Sect. 3 the framework design; Sect. 4 then shows experimental results using the metaheuristics initially provided in the framework on a dataset/problem solved with a specific DNN architecture; Finally, conclusions are presented along with the main work the group intends to carry out in future.

2 Related Work

Although neural networks and now deep neural networks were traditionally trained with gradient-based linear approximations such as backpropagation, Quickprop, Rprop, Conjugate Gradient, etc. [6] and SGD optimizers such as RMSProp [5], Adagrad [7], Adadelta [8] and Momentum [9], neuroevolution since the 1980s has worked on solutions to the learning process of neural networks, understanding them as a continuous optimization process, trying to improve the fact that SGD based algorithms are local search algorithms that, through exploitation, generate a new individual or solution and lack the process of exploring wide space of high dimensional solutions.

Thus, in 1989, Montana and Davis [10] trained an ANN to perform SONAR image classification, obtaining better accuracies than BP. Yao and Floreano then showed how a combination of learning and evolution in ANNs is considered fundamental for adaptation, taking into account the weights of the connections of the network, architectures, input features and learning rules.

In 2015, Morse et al. [1] trained a DNN using a simple genetic algorithm (GA), competitive against SGD and RMSProp. This network solved three problems: time series, approximation of a function and the California housing dataset.

In 2017, it was increasingly seen that DNNs could be trained using GAs [6]. In [11] a DNN was trained using a GA on deep reinforcement learning problems, successfully solving such problems as playing Atari and humanoid locomotion. In 2018 [12] created EvoDeep, which focuses on optimizing parameters and architecture of DNNs, maximizing classification accuracy on MNIST. The research backs evolutionary algorithms as another solution to the DNN training process.

The rise of DNNs brought libraries and frameworks responsible for creating the different deep architectures and implementation of the classic training methods [13]. TensorFlow is a library used widely for constructing DNNs. Many applications have been developed with it, ranging from recognition of visualized fragment molecular orbital (FMO) results [14] to face recognition based on CNNs [15]. Among the many studies in neuroevolution, no frameworks exist for learning optimization of DNNs based on metaheuristics that work in conjunction with TensorFlow. A framework is herein proposed that facilitates integration between deep architectures, various metaheuristics and datasets/problems used to evaluate and compare the performance of deep ANN training algorithms using TensorFlow.

3 Deep Neuroevolution Framework

Neuroevolution has provided new solutions for training DNNs using metaheuristics or evolutionary algorithms able to modify weights, biases, activation functions and/or architecture of the different ANNs. Neuroevolution options combine global search (exploration) and local search (exploitation) over the wide multi-dimensional solution space, taking a population or set of solutions iteration after iteration towards areas where better solutions are located.

In this research, a framework called Deep Neuroevolution Framework (DNF) was designed. Through manipulation of weights and biases of the network. DNF enables learning process optimization using a range of different metaheuristics. The main aim is to facilitate integration between the metaheuristics, classification or regression problems and their different architectures. It does not yet seek to automatically define the network architecture.

DNF is built in Python and TensorFlow library. It works with both CPU and GPU because it uses TensorFlow. To achieve the easy implementation of a neuroevolution application for DNNs, it was constructed based on three fundamental concepts, self-contained and relating to one another: Problem, Solution and Metaheuristics. A general class diagram of DNF is presented in Fig. 1.

As metaheuristics, DNF initially includes Particle Swarm Optimization (PSO), Global-best Harmony Search (GBHS) and Differential Evolution (DE), showing the parameters used to configure them. Three problems shown from the `problem_MNIST` class really correspond to a single problem, MNIST, classic in deep learning. The difference from one to another corresponds to the ANN architecture used to solve it. The first architecture is an ANN of one conventional hidden layer (`Problem_MNIST_one_hidden_layer`), the second is an ANN of five hidden layers (`Problem_MNIST_five_hidden_layers`) and a convolutional ANN (`Problem_MNIST_CNN`). The three key components of the DNF are described below. As the `Problem_MNIST`

class represents a specific problem with three ANN solution architectures, as more problems are added, different solution architectures can be added correspondingly.

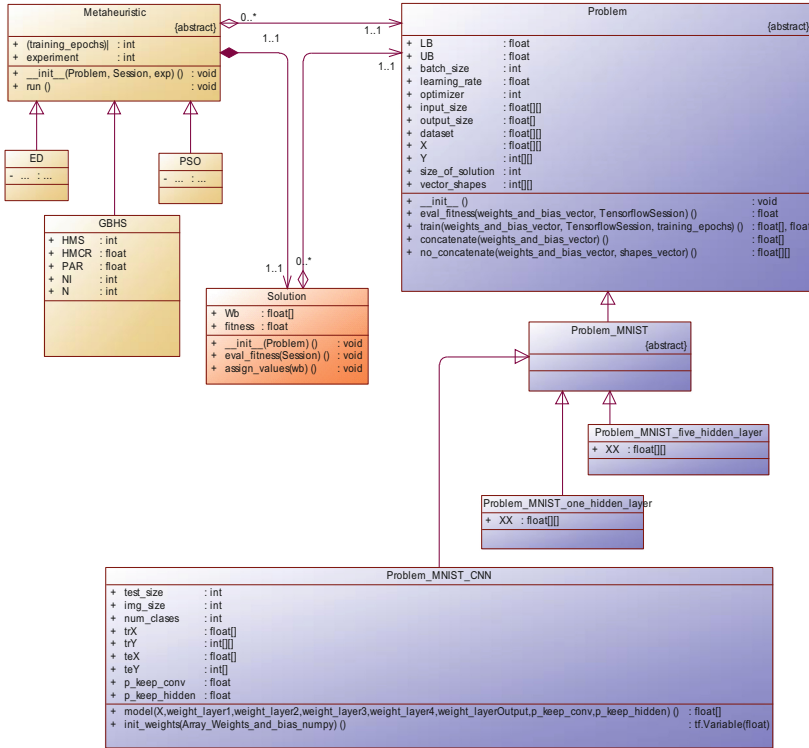


Fig. 1. Diagram of classes of the Deep Neuroevolution Framework (DNF)

3.1 Problem

The problem expressed through the abstract “Problem” class allows easy implementation of different architectures of neural networks for its solution. It was considered necessary to build two key methods: “eval_fitness”, evaluation of network fitness by TensorFlow session using weights and biases available in the solutions; and “train”, responsible for local search or exploitation on the individual and delivering the fitness value, making use of a specific number of training epochs (training_epochs). Within the concrete class of MNIST, the graphs and variables required to create the network architecture are constructed. Two additional methods appear in “Problem”: “concatenate” and “non_concatenate”. These allow us to concatenate the tensors that represent weights and biases of network layers in a unidimensional NumPy array that facilitates modification of the vector solution in “Metaheuristic”. When modifications are made through the specific metaheuristic, “non_concatenate” converts the NumPy array into different tensors that meet form conditions to function in TensorFlow and thus perform

the exploitation process (if “train” is used) or evaluation of the quality of the solution (“eval_fitness”).

3.2 Solution

The “Solution” class incorporates the “Problem” class. This relationship reveals the ranges of values where weights and biases should move (Lower Bound, LB and Upper Bound, UB). It also reveals the total size required for the NumPy array of the solution, depending on the number of neurons in the input, hidden and output layers, a value reported in the problem through the `size_of_solution` attribute. This allows each metaheuristic to modify each value of the unidimensional array of the solution independently and without the need to make direct modifications to values of tensors that represent the ANN that solves the specific problem to be solved.

The separation between “Solution” and “Problem” facilitates the inclusion of more metaheuristics, based on the use of a standard vector representation widely used in the research area. In addition, it gives DNF the ability to grow in problem solving as in the ANN architectures used to solve these problems.

To include PSO, it is required that a particle include, in addition to information of the solution, information related to the best position visited by the particle in its evolutionary process and the velocity of the particle. DNF makes it possible to inherit from “Solution” a new type of class called “Particle” (not presented in the diagram). Finally, in “Solution”, a method of assigning values (`assign_values`) is included to assign the solution vector of each solution.

3.3 Metaheuristics

The “Metaheuristic” class offers ample design possibilities. To connect any metaheuristic with the networks implemented in TensorFlow to solve a specific problem, the first requirement is to start from this base class, considering a vector representation (unidimensional array). The second is that every solution should be an instance of the “Solution” class or classes derived from it. The only further method necessary to implement is called “run” and is the focus of the execution of the logic of the metaheuristic within DNF of a specific experiment identifier. The builder of the class receives a problem object, the TensorFlow session and the seed of initialization represented through an experiment number. This last parameter seeks to ensure the repeatability of the experiments. Thus, all random values generated in metaheuristics or the solution must use a single random object initialized in the metaheuristic with the value of the experiment.

3.4 Integrated Local Search Using TensorFlow Optimizers

SGD, Adam, Adagrad, Momentum, RMSProp, among others, available in TensorFlow comprise a local search scheme that allows us to arrive at better solutions starting from an initial solution generated randomly. These exploitation algorithms work on a single individual and the results to date in DNN training are good. However, some problems

have occurred when the solution space is very large, which usually occurs in this context due to the increase of ANN hidden layers.

These algorithms only perform exploitation, while exploration is carried out in a naive way, executing the algorithm multiple times and choosing the best solution found. Metaheuristics meanwhile carry out the tasks of exploration and exploitation at the same time on one or a set of solutions, which allows them to find better solutions in the same or less time. DNF thus combines the best elements of both by performing exploration processes through metaheuristics and optimizing individuals of the population using TensorFlow optimizers.

Figure 1 shows that “Solution” adds an object of the abstract problem class and the concrete problems that are implemented to be optimized have a “train” method that allows the metaheuristics not only to evaluate the quality of the solution, but also run the local optimization algorithms a specific number of times (`training_epochs`) defined in the ANN that solves the problem. These kinds of concrete problems also include in their configuration the list of optimizers used to do the local search on the solution. This allows metaheuristics to become memetic algorithms, which are the state of the art today in several optimization problems, especially those that are very complex and with high dimensionality.

3.5 RAM or VRAM Memory Considerations

Management of RAM or VRAM (GPU memory) is vital. TensorFlow philosophy for implementing ANNs is based on designing the graph set required and executing the graphs via a work session. When done iteratively, TensorFlow accumulates variables in the memory, a problem when the process is run thousands of times. The same applies when defining average behavior of an optimizer, with E ($E > 30$) executions. The algorithm is executed several times without closing TensorFlow but objects are also accumulated, filling the RAM. One experiment was carried out measuring memory consumption executing a process that performs design and execution of a graph iteratively with data loaded in RAM, and a second performing several separate processes with the same operation. In the first, despite closing the session each time the experiment ends, RAM memory increases, since the constant calling to the design of the graphs produces leaks in RAM memory. In stark contrast, when it makes the call of experiments through independent processes, consumption remains constant. This indicates that TensorFlow releases resources only when the execution of a program ends completely. DNF makes use of this and through a class called `Experimenter` makes calls of independent experiments to calculate the average performance of a metaheuristic.

4 Experimentation

In order to evaluate the behavior of DNF in the optimization of deep neural networks, two experiments were carried out: (1) a comparison between Adam and RMSProp optimizers available in TensorFlow (integrated with the same DNF architecture to ensure a fair comparison) and metaheuristics implemented using only one (1) epoch of

local optimization, and (2) A comparison of the results of the metaheuristics with the best results against their memetic version, varying the number of epochs (intensification) of the local search.

4.1 Problem

The experimentation was conducted seeking to train a deep neural network with three hidden layers in the classification of digits written freehand known as MNIST and whose data are available through TensorFlow. The architecture of the network is: five hidden layers, one to five, containing 200, 100, 60, 30, and 10 neurons respective. The activation function of the first four layers is Sigmoid, while the fifth layer uses Soft-Max. In the two experiments, the aim is to minimize the cost function known as cross entropy. This architecture is chosen because it is reported in several documents with good results for accuracy and the training process is not costly in time, requiring approximately 1000 iterations with the RMSProp optimizer.

4.2 Results on Experiment 1

For this research, the three metaheuristics available in DNF were executed, namely: Global-best Harmony Search (GBHS), Particle Swarm Optimization (PSO) and Differential Evolution (DE). In GBHS, a population of five solutions is created (Harmony Memory Size), 50 iterations are made (Number of Improvisations, NI), Pitch Adjustment Rate (PAR) is set at 0.35 and Harmony Memory Consideration Rate is 0.9. In PSO, the size of the initial population is five, constant inertia weight equal to 0.5 and cognitive and social constants equivalent to 1 and 2 respectively. For DE, the initial population is five, the mutation rate equal to 0.5, and recombination rate equal to 0.7. For the experimentation, the values of the parameters recommended in the literature were used. No process of refining parameters was carried out at this time.

DNF stores the cross entropy values every certain number of evaluations of the objective function (EOFs) and with that, then proceeds to make a convergence curve of the best solution found up to a certain number of EFOs. For this experiment it was decided to execute the optimizers Adam and RMSProp with a total of 1000 epochs, a value that enables very good entropy results to be obtained. As mentioned, DNF allows metaheuristics to perform a local search on the solutions using TensorFlow optimizers. For this experiment the Adam optimizer was used, and it could perform only one (1) epoch (intensification of the local search) of optimization on the solutions.

To be fair when comparing metaheuristics with the optimizers of TensorFlow, using Eq. (1) the maximum number of iterations made with the population metaheuristics is defined, to perform the same 1000 evaluations of the objective function that are carried out using RMSProp and Adam.

$$NI = \frac{1000 - P_0 * NELS}{NEFOs} \tag{1}$$

Where NI is the number of iterations to be performed by the population metaheuristic (GBHS, PSO or DE), P0 is the number of solutions in the initial population,

NELS is the number of epochs in local search and NEFOs is the number of EFOs performed within the block of iterations of each metaheuristic. Thus: GBHS uses two EFOs (one for the local optimization of the new improvise and another for the best individuals of the population), PSO and DE perform five EFOs (one for each solution of the new population). Therefore, 1000 times in Adam or RMSProp equals 498 iterations (NI) for GBHS and 199 for PSO and DE. Figure 2 shows the convergence curve of the algorithms (Adam, RMSProp, PSO with Adam, DE with Adam, GBHS with Adam and GBHS with RMSProp) using the average of 30 experiments.

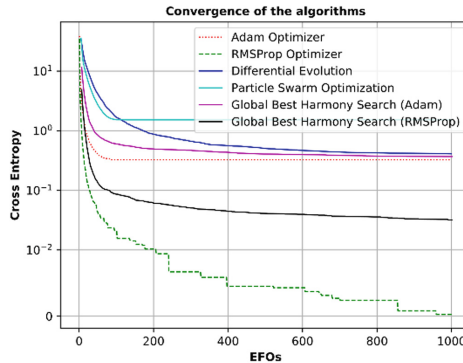


Fig. 2. Convergence curves for 1000 evaluations of the objective function between the Adam and RMSProp optimizers and the metaheuristics GBHS, PSO and DE

In Fig. 2 we see how metaheuristics are competitive against the RMSProp optimizer. Even when GBHS uses RMSProp as its local optimizer, it has a better performance than when using the Adam optimizer. In Table 1, the average entropy value is shown with different values of EFOs in this experiment.

Of the three metaheuristic populations, the best behavior during the learning process is GBHS (red color), for this reason it is chosen to carry out the second experiment in which it is sought to evaluate the impact on intensification of the local search in metaheuristics.

Table 1. Average of cross entropy (minimize) for different EFOs during the training process of the neural network (the best results are shown in bold)

Algorithm	EFOs			
	100	200	500	1000
RMSProp	0.021136	0.010380	0.004532	0.000261
GBHS/RMSProp	0.086010	0.061008	0.040913	0.031804
Adam	0.325254	0.325254	0.325254	0.325254
GBHS/Adam	0.599207	0.494281	0.406116	0.368744
DE/Adam	1.683368	0.873293	0.503006	0.410012
PSO/Adam	1.528210	1.522163	1.522163	1.522163

4.3 Results on Experiment 2

This experiment seeks to analyze the impact of varying the number of training periods (intensification) in the local search within GBHS (1, 2, 4, 8 and 20 training periods). In this sense, we seek to find the value of the training period parameter that provides the best performance to GBHS in its memetic version. To achieve this, the convergence curves of GBHS with its five series of values for the training periods and their comparison with RMSProp are presented in Fig. 3.

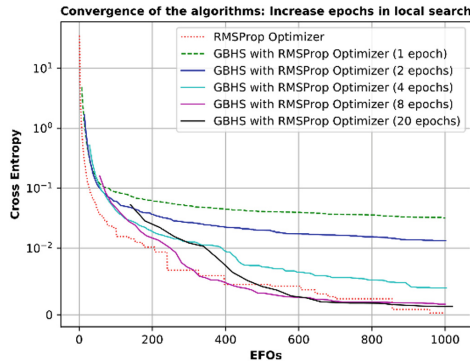


Fig. 3. GBHS convergence curves with different number of training epochs (exploitation) in the local search

In Fig. 3 it is observed that as the number of training epochs increases in the local search, the convergence curve of GBHS obtains better results. In Table 2 the GBHS with eight times in its local search surpasses RMSProp when it uses 420 EFOs, then GBHS with 20 epochs is converted to 856 EFOs, at which time RMSProp manages to overcome the metaheuristics until EFO 1000. It should be noted that the GBHS with 20 epochs continues to produce competitive results against RMSProp and behaves as the configuration with the best average entropy preserved (red color) after EFO 620.

Table 2. Convergence of cross entropy in the optimizer RMSProp vs. GBHS/RMSProp with different levels of intensification in the local search (1, 2, 4, 8 and 20 epochs)

Algorithm	EFOs				
	100	420	620	856	1000
RMSProp	0.021136	0.004532	0.003730	0.000794	0.000261
GBHS/RMSprop (20 epochs)	0.053043	0.006345	0.002545	0.001514	0.001237
GBHS/RMSprop (8 epochs)	0.044487	0.004311	0.002552	0.001825	0.001578
GBHS/RMSprop (4 epochs)	0.042615	0.008954	0.006255	0.004889	0.004006
GBHS/RMSprop (2 epochs)	0.059845	0.021795	0.017199	0.014530	0.013361
GBHS/RMSprop (1 epoch)	0.086010	0.043396	0.038824	0.034029	0.031803

Finally, the results of the training are evaluated on the set of tests available for the MNIST problem. The results based on Experiment 2 are shown in Table 3.

Table 3. Accuracy of the test set of the MNIST problem using the results of RMSProp and GBHS with 1, 2, 4, 8 and 20 epochs of local search

Algorithm	Average accuracy	Standard deviation
RMSProp	0.97637	0.00099
GBHS (20 epochs)	0.97538	0.00138
GBHS (8 epochs)	0.97476	0.00117
GBHS (2 epochs)	0.97463	0.00167
GBHS (4 epochs)	0.97449	0.00155
GBHS (1 epoch)	0.97389	0.00212

In Table 3, it can be seen how the accuracy values of GBHS are very close to those of RMSProp, with a decrease that varies between 0.1% and 0.25%. In addition, it is observed that as the number of training periods (intensification) increase in the local search, the standard deviation of the results tends to zero. The foregoing is promising and allows us to think that it is possible to remain competitive with metaheuristics against the classical algorithms of the state of the art in the training of neural networks. Moreover, other metaheuristics specialized in high dimensionality continuous optimization problems with multimodal functions that resemble the landscape of deep RNA training functions should be evaluated.

5 Conclusions and Future Work

DNF is a framework for optimizing DNN training using metaheuristics based on TensorFlow and implemented in Python. DNF facilitates the inclusion of various problems and different architectures of DNNs to solve each problem. It focuses optimization on defining the best values of weights and biases and uses a classic one-dimensional vector representation. DNF also allows metaheuristics to easily include local search operations using the optimizers available in TensorFlow.

DNF was designed with a weakly coupled architecture considering three main classes: Problem, Solution and Metaheuristic. This facilitates the integration of new metaheuristics without modifying the already written code. Currently, DNF includes PSO, GBHS and DE and different neuronal network architectures to solve problems. DNF was evaluated through comparison between two optimizers (Adam and RMSProp) versus PSO, GBHS and DE. The results show that metaheuristics were competitive in the DNN learning process. Furthermore, it is shown that increasing the number of iterations in local searches on the best GBHS metaheuristics allows us to increase the accuracy on MNIST, making it competitive with RMSProp.

As future work, we intend to include the following in DNF: (1) create new metaheuristics of large-scale global optimization to improve optimization of the DNN learning process; (2) carry out tests with more complex problems and deeper

architectures, for example the best architecture found by EvoDeep; (3) include the possibility of executing the training or evaluation of solutions in different graphic cards; and (4) implement multiobjective metaheuristics to modify DNN weights and biases.

References

1. Morse, G., Stanley, K.O.: Simple evolutionary optimization can rival stochastic gradient descent in neural networks. In: Proceedings of the 2016 on Genetic and Evolutionary Computation Conference - GECCO 2016, pp. 477–484 (2016)
2. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. *Neural Comput.* **18**, 1527–1554 (2006)
3. Bengio, Y., Lecun, Y.: Scaling learning algorithms towards AI. In: Bottou, L., Chapelle, O., DeCoste, D., Weston, J. (eds.) *Large-Scale Kernel Machines*, pp. 321–360. MIT Press (2007)
4. Dauphin, Y.N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., Bengio, Y.: Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In: Proceedings of the 27th International Conference on Neural Information Processing Systems - NIPS 2014, Montreal, Canada, vol. 2, no. 9, pp. 2933–2941. MIT Press, Cambridge (2014). <http://dl.acm.org/citation.cfm?id=2969033.2969154>
5. Hinton, G.E.: Neural networks for machine learning lecture 6a overview of mini-batch gradient descent reminder : the error surface for a linear neuron. http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf
6. Ojha, V.K., Abraham, A., Snášel, V.: Metaheuristic design of feedforward neural networks: a review of two decades of research. *Eng. Appl. Artif. Intell.* **60**, 97–116 (2017)
7. Duchi, J.C., Bartlett, P.L., Wainwright, M.J.: Randomized smoothing for (parallel) stochastic optimization. In: Proceedings of IEEE Conference on Decision and Control, vol. 12, pp. 5442–5444 (2012)
8. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. [arXiv:1212.5701](https://arxiv.org/abs/1212.5701) [cs] (2012)
9. Bengio, Y., Boulanger-Lewandowski, N., Pascanu, R.: Advances in optimizing recurrent networks. In: ICASSP, Proceedings of the IEEE International Conference on Acoustics Speech Signal Processing, pp. 8624–8628 (2013)
10. Montana, D.J., Davis, L.: Training feedforward neural networks using genetic algorithms (1989). <http://dl.acm.org/citation.cfm?id=1623876>
11. Such, F.P., Madhavan, V., Conti, E., Lehman, J., Stanley, K.O., Clune, J.: Deep Neuroevolution: Genetic Algorithms Are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning (2017)
12. Martín, A., Lara-Cabrera, R., Fuentes-Hurtado, F., Naranjo, V., Camacho, D.: EvoDeep: a new evolutionary approach for automatic deep neural networks parametrisation. *J. Parallel Distrib. Comput.* **117**, 180–191 (2018)
13. Deep Learning Frameworks|NVIDIA Developer. <https://developer.nvidia.com/deep-learning-frameworks>
14. Saitou, S., et al.: Application of TensorFlow to recognition of visualized results of fragment molecular orbital (FMO) calculations. *Chem-Bio Inform. J.* **18**, 58–69 (2018)
15. Yuan, L., Qu, Z., Zhao, Y., Zhang, H., Nian, Q.: A convolutional neural network based on TensorFlow for face recognition. In: Proceedings of 2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference, IAEAC 2017, pp. 525–529 (2017)



New Fuzzy Singleton Distance Measurement by Convolution

Rodrigo Naranjo and Matilde Santos^(✉)

Complutense University of Madrid,
Computer Architecture and Automatic Control Department,
28040, Madrid, Spain
rnanranjo.ina@gmail.com, msantos@ucm.es

Abstract. This article proposes a new method to calculate the distance between fuzzy singleton variables. It uses a measure of generalized fuzzy numbers based on the center of gravity. The fuzzy signals are transformed by applying convolution. To prove the effectiveness of this method, it is applied to a pattern recognition problem that deals with stock markets. Comparison with other classical distance measurements shows that this approach provides a consistent and reliable distance measure for the stock market scenario and can be generalized for any pattern recognition problem.

Keywords: Fuzzy distance · Convolution · Pattern recognition
Stock markets

1 Introduction

Since Zadeh raised fuzzy sets in 1965 [1], many researchers have exploited their potential to model human thinking and apply it to the most diverse scenarios [2, 3].

One of the most important issues when dealing with fuzzy sets is to obtain a reliable calculation of the similarity or the distance between fuzzy variables. In the literature, there are some approaches to address this problem. For example, in [4] authors propose a similarity measure based on arithmetic operations of the values which define the fuzzy sets; specifically, they use the average of the distances between the four values that define a trapezoidal set. In [5] these arithmetic operations are extended to emphasize certain parts of the fuzzy regions and to try to adjust the result of the measurements. There is a qualitative leap in [6], where authors calculate the similarity using the simple center of gravity method (SCGM). Afterwards, in [7], that center of gravity is replaced by a gravity radius, or geometric measures are used instead of arithmetic [8].

Nevertheless, the methods these papers propose to calculate the similarity measure do not properly work in certain cases, or they are not particularly sensitive. That is why in [9] the authors calculate the similarity of generalized fuzzy numbers by a new approach and compare their work with others for different examples.

These methods are designed to determine the similarity between fuzzy sets, but not to calculate the similarity (or distance) between fuzzy singletons that have been obtained as a result of the fuzzification of crisp variables (Fig. 1). In this Fig. 1, the

fuzzy singletons A and B (fuzzy numbers) are obtained after the fuzzification of two variables (a and b), which belong to the same fuzzy set X, with different membership degrees to that fuzzy set (μ_A and μ_B , respectively).

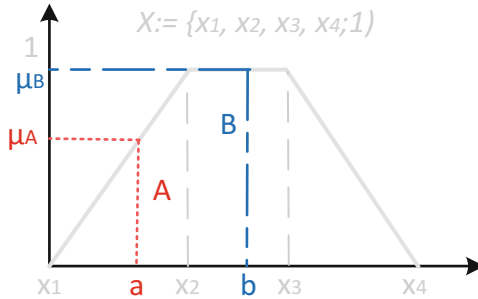


Fig. 1. Fuzzification of two variables

Therefore, in this paper we propose a new method to calculate the distance between two fuzzy singletons obtained as a result of the fuzzification.

The interest of this study lies in the fact that in many pattern recognition applications fuzzy variables are used to characterize and then detect a specific pattern. An example of stock market forecasting based on this approach will be shown.

This work is organized as follows. In the next Section, the method proposed to calculate the distance between fuzzy variables is explained. Section 3 presents the application of the new method to a general case and compares it with two classical distance definitions. In Sect. 4, the calculation of the distance by the convolution is applied to a pattern recognition problem. The conclusions and future work end the paper.

2 Proposed Method

The method here proposed to calculate fuzzy distance is based on the generalized fuzzy number similarity calculation reported by Sridevi and Nadarajan [9]. It provides great sensitivity and coherence regarding the measurements, besides some basic properties analyzed in [5], such as:

- *Reflexivity*: two generalized fuzzy numbers (A and B) are identical if, and only if their similarity is one, i.e., $S(A, B) = 1$.
- *Symmetry*: the similarity of two generalized fuzzy numbers $S(A, B)$ is equal to $S(B, A)$, i.e., $S(A, B) = S(B, A)$.

Besides, other properties can considerably reduce the computation time and are very useful when implementing a pattern recognition system with massive data. Among them, the method developed by [8] make easier the calculation of the similarity in certain cases; for example, the similarity of real numbers that represent trapezoidal fuzzy

sets ($A := (a, a, a, a; 1)$, $B := (b, b, b, b; 1)$), similarity of equal real numbers with different membership function ($A := (a, a, a, a; w_A)$, $B := (a, a, a, a; w_B)$), or in the latter, the similarity when one of the two membership functions (w_A or w_B) is equal to 1.

Specifically, the formula for the calculation of the similarity for two generalized fuzzy numbers according to [8] is given by (1) is (2):

$$A := \{a_1, a_2, a_3, a_4; w_A\}, \quad B := \{b_1, b_2, b_3, b_4; w_B\} \tag{1}$$

$$S(A, B) = \frac{1}{4} \sum_{i=1}^4 \mu(x) (1 - |x_A^* - x_B^*|)^{B(S_A, S_B)} \frac{\min(y_A^*, y_B^*)}{\max(y_A^*, y_B^*)} \tag{2}$$

Where $\mu(x)$ represents the membership function that measures the difference of the distance of two generalized fuzzy numbers (GFN):

$$\mu(x) = \begin{cases} 1 - \frac{x}{d} & 0 \leq x \leq d \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $0 \leq d \leq 1$ and $x = |ai - bi|$; d is usually set to 0.5, as we will do in the experiments.

The B function for two similarity measures S_A and S_B is defined as,

$$B(S_A, S_B) = \begin{cases} 1 & S_A + S_B > 0 \\ 0 & S_A + S_B = 0 \end{cases} \tag{4}$$

This function $B(S_A, S_B)$ is 0 or 1 according to whether the centre of gravity (COG) is considered or not. Finally $S_A = a_4 - a_1$ and $S_B = b_4 - b_1$, being the values $y_A^*, y_B^*, x_A^*, x_B^*$, defined in (5-6).

$$y_A^* = \begin{cases} \frac{w_A(a_3 - a_2 + 2)}{6} & a_1 \neq a_4 \\ \frac{w_A}{2} & a_1 = a_4 \end{cases} \quad y_B^* = \begin{cases} \frac{w_B(b_3 - b_2 + 2)}{6} & b_1 \neq b_4 \\ \frac{w_B}{2} & b_1 = b_4 \end{cases} \tag{5}$$

$$x_A^* = \frac{y_A^*(a_3 + a_2) + (a_4 + a_1)(w_A - y_A^*)}{2w_A} \quad x_B^* = \frac{y_B^*(b_3 + b_2) + (b_4 + b_1)(w_B - y_B^*)}{2w_B} \tag{6}$$

Then, according to (2), the measure of the distance will be,

$$D(A, B) = 1 - S(A, B) \tag{7}$$

However, this measure of the distance applies for two generalized fuzzy numbers, and not for two fuzzy singletons (Fig. 1). That is why an intermediate step that converts the problem of measuring two singletons into measuring two equivalent fuzzy numbers is necessary.

It consists of transforming the characteristics of a fuzzy singleton (its own fuzzy value and its membership value to a certain fuzzy set) to the characteristics of the

generalized fuzzy number that are used in the distance calculation, that is, the scaling of the generalized fuzzy number and the separation between them.

To do this, from the signal processing point of view and for the fuzzy number A (Fig. 2), we can consider that there are two time signals, one of them trapezoidal (fuzzy set X) and the other one an impulse function (A). That is, the fuzzification operation (Mamdani type) can be interpreted as an impulse signal.

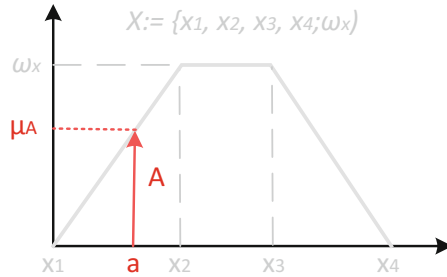


Fig. 2. Time signals that represent the fuzzification

The unit impulse function, δ , is defined as (8):

$$T\delta(t - t_0) = \begin{cases} T & t = t_0 \\ 0 & t \neq t_0 \end{cases} \quad T = \mu_A, \quad t_0 = a \quad (8)$$

The composition of those two functions is obtained by the convolution operation. Thus, the convolution of the trapezoidal function and the unit impulse is given by (9):

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t - \tau)d\tau \quad (9)$$

However, one of the properties of the impulse function is that when convolving with any other signal, the latter is scaled (T) and delayed (t_0).

Therefore, when applying the convolution, Fig. 2 is transformed to Fig. 3 (right).

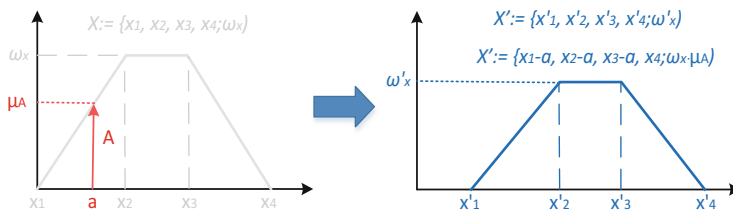


Fig. 3. Signal convolution

As a result, a twofold effect is obtained. First, the amplitude of the fuzzy set is scaled to the membership value of the fuzzy variable (the amplitude of the singleton). Second, the fuzzy set is delayed, that is, it is shifted.

Therefore, we can say that the new generalized fuzzy sets obtained, A' and B' , include the effects of the fuzzification. The fuzzy numbers obtained by this method are represented in Fig. 4.

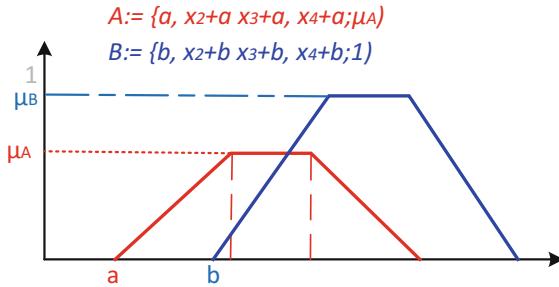


Fig. 4. Result of applying the convolution to the fuzzy singletons of Fig. 1

On these fuzzy numbers, A' and B' , we can now apply the distance definition given by (7).

3 Comparison with Classical Distances

Our proposal is compared with two classical distance measurements, the Euclidean distance and a modified version of this one, the weighted distance.

The main problem that arises in [9] is due to the fact that the distance measurement they apply is not efficient with the nearest neighbor k - nn algorithm. This would have required the use of other classical distance implementation, such as the Euclidean distance. The Euclidean distance applied to the values obtained by the fuzzification is (10),

$$D_{euc}(\{a_1, a_2, \dots, a_n\}, \{b_1, b_2, \dots, b_n\}) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \dots + (a_n - b_n)^2} \tag{10}$$

If we apply this distance definition to the values of Fig. 1, as there is only one element, the distant is $b-a$.

However, it is clear by (10) that this method does not take into account the membership value of the variable. For this reason, another method that weights the distance according to this value (weighted distance) is proposed (11).

$$D_{weig}(\{(\mu_{a1}, a_1), (\mu_{a2}, a_2) \dots, (\mu_{an}, a_n)\}, \{(\mu_{b1}, b_1), (\mu_{b2}, b_2) \dots, (\mu_{bn}, b_n)\}) = \sqrt{(\mu_{a1}a_1 - \mu_{b1}b_1)^2 + (\mu_{a2}a_2 - \mu_{b2}b_2)^2 + \dots + (\mu_{an}a_n - \mu_{bn}b_n)^2} \tag{11}$$

A priori, this last distance measurement should provide good results because it takes into account the two distance key factors; on the one hand, it considers the distance between the obtained values and, on the other hand, it also includes their membership value.

To illustrate the performance of the distance measurement proposed in this article the following example has been used. The main objective of this example is to see how these three similarity measures (Euclidean, weighted, and the proposed fuzzy distance measures) perform when they measure the distance between two fuzzy singletons that are very close to each other, throughout the domain [0, 1]. The most relevant points of the interval under study are the transitions between fuzzy sets. Supposedly, if the fuzzy singletons belong to different fuzzy sets, that should be somehow shown in the distance measure, even if they are close regarding the domain.

First, five trapezoidal (triangular) fuzzy sets, *A*, *B*, *C*, *D* and *E*, have been defined,

$$A := (0, 0, 0, 0.25; 1)$$

$$B := (0, 0.25, 0.25, 0.50; 1)$$

$$C := (0.25, 0.50, 0.50, 0.75; 1)$$

$$D := (0.50, 0.75, 0.75, 1; 1)$$

$$E := (0.75, 1, 1, 1; 1)$$

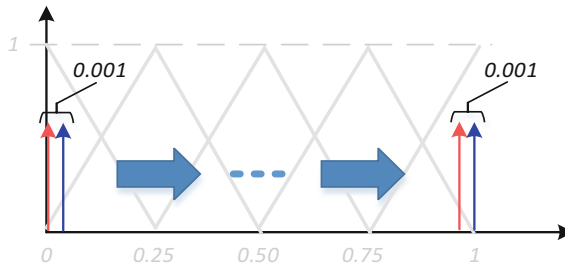


Fig. 5. Displacement of the two fuzzy variables

The initial and final values of two fuzzy singletons, separated by a constant distance of 0.001 between them, that have been shifted along the domain interval, are shown in Fig. 5.

The three distance calculation methods have been applied to those fuzzy singletons. Figure 6 shows how each fuzzy distance calculation method performs. To see it more

clearly, the distances have been multiplied by a constant factor of 30, to be able to show them on the same graph.

According to this figure, the Euclidean distance (red line) shows the worst behavior in this example because it maintains constant the separation between the fuzzy variables.

The weighted distance (green line) neither has the desired performance. In this case, small jumps are obtained at the intersections of the fuzzy sets, but the distance should be much greater since a variable belongs to one fuzzy set and the other to the adjacent fuzzy set. Besides, there are some peaks in these fuzzy regions where they should be minimum. Finally, there is an upward trend due to the fact that, as the membership

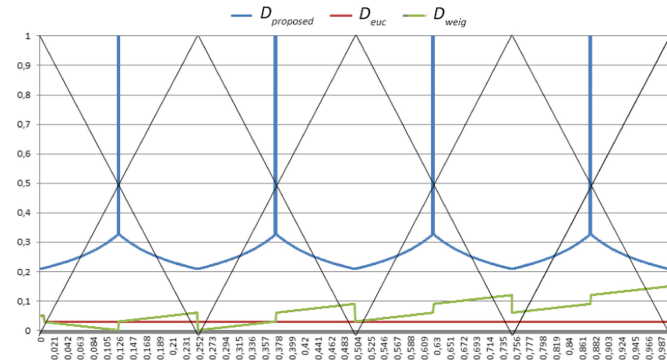


Fig. 6. Comparative of the three distances applied to fuzzy values. (Color figure online)

function is multiplied by the value of the x -axis, as it increases, the differences tend to be larger.

On the contrary, the proposed method (blue line) has a consistent behavior throughout the variable domain, obtaining the maximum distance at the intersections of the fuzzy sets, and presenting a perfectly periodic behavior.

The observed performance of the proposed distance measure solves one of the main problems in the forecasting systems based on comparison of fuzzy patterns, where similar values are obtained for fuzzy variables that belong to different fuzzy sets.

We have proved that classical distance measures fail to calculate the similarity in this sense. They give a small distance between fuzzy singletons that belong to different fuzzy sets. On the contrary, our proposal takes that into account and performs as expected.

4 Pattern Recognition Problem Application

To prove the effectiveness of the proposed model, a real pattern recognition problem is used. Particularly, a stock market application has been chosen. In [10], Japanese candlesticks pattern recognition was implemented in order to forecast the next

investment session. In that paper, three fuzzy variables were defined, namely *Lupper*, *Lbody* and *Llower*. They represent the upper shadow, lower shadow and the body size of the candle (percentage). Five triangular fuzzy sets, X_1 to X_5 , were assigned to each variable in the interval $[0, 1]$.

As a result of the fuzzification there are three fuzzy singletons, A , B and C . The input A (0.5) only belongs to the central fuzzy set X_3 with membership value = 1; input B (0.3775) belongs to the central fuzzy set, X_3 , with membership value of 0.51 and also to the previous X_2 fuzzy set. The input variable C (with value of 0.6275) belongs to X_3 fuzzy set with a membership of 0.51 and it also belongs to the adjacent fuzzy set X_4 . Therefore, both variables (B and C) have the same membership value, but they belong to different fuzzy sets, B belongs to X_2 and X_3 and C belongs to X_3 and X_4 .

The three distance definitions have been applied. The Euclidean distances $B-A$ and $A-C$ are almost identical, 12.25 and 12.75, respectively. The weighted distance is also calculated, obtaining $D_w(A, B) = 0.3074$ and $D_w(A, C) = 0.1799$. Finally, the new proposed distance is obtained, $D_{\text{new}}(A, B) = 0.6621$ and $D_{\text{new}}(A, C) = 0.9222$.

The value of the distance calculated by D_w is wrong. It means that the distant from B to A is larger than from C to A . This is not correct because A and B belong to the same fuzzy set. This is because the product of $\mu_A \cdot a$ is greater than $\mu_C \cdot c$, when it should not be so. However, with the proposed distance calculation method the measurements are at least coherent. The fact that they belong to different fuzzy sets make the distances consistent (unlike D_w) and considerably different (unlike D_e).

With these two examples, the theoretical one and the pattern recognition application, it has been shown that the calculation of fuzzy distances should have into account some factors that influence the expected result.

5 Conclusions and Future Works

In this paper we propose a new method to calculate the distance between fuzzy singletons in a fuzzy domain. It has been compared with other classical definitions of distance, such as the Euclidean and the weighted ones.

The obtained results show that the proposed method gives coherent results regarding the expected distance within the fuzzy frame, unlike the other distance definitions. As the distance is used as a measure of the similarity in some pattern recognition problems, we think this is an interesting contribution.

As future work, it would be interesting to compare the results with the ones reported in [10], using the same pattern recognition system but applying the new distance calculation.

References

1. Zadeh, L.: Fuzzy sets. *Inf. Control* **8**, 338–353 (1965)
2. Santos, M.: Intelligent control: Applied approach. *Revista Iberoamericana de Automática e Informática Industrial RIAI* **8**(4), 283–296 (2011)

3. López, V., Santos, M., Montero, J.: Fuzzy specification in real estate market decision making. *Int. J. Comput. Intell. Syst.* **3**(1), 8–20 (2010)
4. Chen, S.M.: New methods for subjective mental workload assessment and fuzzy risk analysis. *Cybern Syst.: Int. J.* **27**(5), 449–472 (1996)
5. Hsieh, C.H., Chen, S.H.: Similarity of generalized fuzzy numbers with graded mean integration representation. In: 8th International Proceeding on Fuzzy Systems Association World Congress, vol. 2, Taipei, pp. 551–555 (1999)
6. Chen, S.J., Chen, S.M.: Fuzzy risk analysis based on similarity measures of generalized fuzzy numbers, *IEEE Trans. Fuzzy Syst.* **11**, 45–56 (2003)
7. Yong, D., Wenkang, S., Feng, D., Qi, L.: A new similarity measure of generalized fuzzy numbers and its application to pattern recognition. *Pattern Recogn. Lett.* **25**, 875–883 (2004)
8. Chen, S.J.: New similarity measure of generalized fuzzy numbers based on geometric mean averaging operator. In: IEEE International Proceedings Conference on Fuzzy Systems, Vancouver, Canada, pp. 1879–1886 (2006)
9. Sridevi, B., Nadarajan, R.: Fuzzy similarity measure for generalized fuzzy numbers. *Int. J. Open Probl. Comput. Sci. Math.* **2**(2), 240–253 (2009)
10. Naranjo, R., Santos, M.: Fuzzy candlesticks forecasting using pattern recognition for stock markets. In: Graña, M., López-Guede, J.M., Etxaniz, O., Herrero, Á., Quintián, H., Corchado, E. (eds.) ICEUTE/SOCO/CISIS -2016. AISC, vol. 527, pp. 323–333. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-47364-2_31



Peak Alpha Based Neurofeedback Training Within Survival Shooter Game

Radu AbuRas¹(✉), Gabriel Turcu¹, Ilkka Kosunen²,
and Marian Cristian Mihaescu¹

¹ University of Craiova, Craiova, Romania
raduradu0410@yahoo.com, gabi_turcu_1997@yahoo.com,
mihaescu@software.ucv.ro

² University of Tallinn, Tallinn, Estonia
ilkka.kosunen@gmail.com

Abstract. Neurofeedback has been proven to be a useful tool in games, music or video applications for both treating various disorders and disabilities, as well as potentially enhancing cognitive functions of healthy individuals. However, most neurofeedback protocols are tedious to use, and the usefulness of the results is difficult to validate. In this paper, we present a framework that allows connecting various types of neurofeedback protocols inside a computer game in a way that the user can gain full benefits of the neurofeedback based training by simply playing a game. The paper outlines the full potential of cognitive training and its effectiveness when implemented in an entertaining scenario.

Keywords: Neurofeedback training · Shooter game
Alpha and theta training

1 Introduction

While the neural mechanisms that involved neurofeedback training (NFT) are still unexplained [4], the benefits of the technique have been proven both in clinical situations where neurofeedback has been successfully used to treat disorders such as epilepsy [22], learning disabilities [7], Attention Deficit/Hyperactivity Disorder (ADHD) [2, 3, 17], and autism spectrum disorders [15].

Neurofeedback has also shown potential to be a useful tool for healthy individuals to reach their peak potential [9], as well as to improve and evaluate working memory and memory performance for stroke patients [10, 13] or attentional processes such as sustaining, orienting and executive attention [9]. Neurofeedback has even been used to help meditation by combining virtual reality and neurofeedback training while measuring the level of concentration and relaxation [14].

One particular approach with reasonable effectiveness has been taken into consideration: individualization of theta/beta ratio (TBR) for ADHD children that are characterized with decreased individual alpha peak frequency (iAPF),

alpha bandwidth and alpha amplitude suppression magnitude. In this individualized context, there were identified various (adjusted) alpha activity metrics with a close relationship with ADHD symptoms [3].

Currently, one approach for coping with ADHD symptoms or other medical situations by using NFT is by developing EEG driven games. Unfortunately, some clinical studies had the goal to entertain children rather than apply and evaluate a learning procedure [2]. From this perspective, development of NFT driven games with clearly defined goal and mechanisms for evaluating learning outcomes represents a challenge that needs the researcher's attention. Recently, various dynamic difficulty adjustment strategies have been proposed by [21] triggered by EEG in multi-player games. Still, training and validation remain the vital critical issues for EEG driven games.

As main application domain of NFT [18] we outline clinical applications for treatment of ADHD, anxiety, depression and many other dysfunctions highlighting the advantages and limitations of proposed therapies regarding the effectiveness of treatment of such neurofeedback software. Still, the latest research does not provide conclusive results regarding the efficiency of NFT.

The goal of this paper is to build a data analysis pipeline within an NFT tool that is designed as an interactive game. The tool may integrate various cognitive training methods and approaches in order to achieve positive results for various patients: healthy people, people that have suffered chronic strokes or people with ADHD.

The chosen method for NFT is peak alpha and alpha/theta ratio because these two bands are more prone to manipulation while manipulation of beta band amplitudes are independent from specified training in visual attention processing scenarios [11].

2 Related Work

Various NFT protocols have been widely used for a wide range of practical problems such as recovery of memory, improving working memory, improving the sustained attention and orientation, improving the level of concentration and relaxation, treatment of ADHD or improving cognitive and affective gains. Among the most used NFT protocols there are SMR (Sensory Motor Rythm), TBR (theta/beta ratio), upper-alpha, high alpha, peak alpha frequency, gamma, theta and many other [9].

Among many available NFT protocols, the alpha training (enhancement and suppression) has been investigated by [8] in order to improve motor skills. Their study showed that alpha suppression might be more effective in the consolidation of motor memory.

Another example of NFT aimed to increase the individual upper alpha power in the parieto-occipital area of the scalp has been reported by [6] in their attempt to investigate if working memory performance can be improved for patients with a major depressive disorder. The experimental results in performed studies showed a positive correlation between the improvement in processing speed and

the increase of alpha power and thus the effectiveness of the proposed protocol. Specifically, EEG peak alpha frequency (PAF) has been shown to correlate positively with cognitive performance and to correlate negatively with age after childhood [1]. The same interrelation has been found between sensorimotor abilities, cognitive performance and individual alpha peak frequency (iAPF) in [19].

Biofeedback has been combined with (computer) games in various ways. In a comprehensive study, Kuikkaniemi et al. studied how biofeedback would enhance the game experience both when the players were aware of the feedback as well as when the game adapted implicitly without players knowing it [16].

Various BCI techniques have been used for creating neuroadaptive games: Pineda et al. proposed a system where the player could navigate in a 3D game by controlling their sensorimotor mu-rhythm [20]. Larol et al. utilized Steady State Visual Evoked Potentials (SSVEP) for controlling a game character: the system would show checkerboards flashing at different frequencies and, by observing the user's visual cortex the BCI system could detect which of the checkerboards the user was concentrating and use this information as control signal.

3 Proposed Approach for the NFT

The NFT is based on a data analysis pipeline that is presented in Fig. 1. It consists of several modules that are designed to process the raw input (i.e., EEG signals) and finally update the GUI of survivor shooter game.

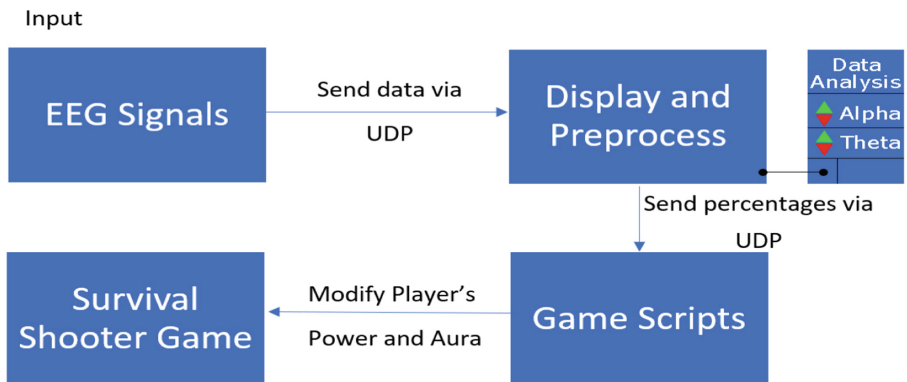


Fig. 1. Data analysis pipeline

The module responsible for receiving the raw EEG signals from Nautilus headset is based on Matlab and Simulink scripts that obtain the analog signals. From available signals, we save only the alpha and theta signals for later use. This module is also responsible for sending via UDP two different packets, one for alpha signal and one for theta signal. Once the signals are logged they are further

available for display and process by other modules within the data analysis pipeline.

Once the data is available in *.csv* format it is ready to be processed and displayed. All processing is performed in a custom desktop application (i.e., the game) that uses *qt* framework along with other external libraries such as *qcustomplot* for displaying the dynamic signal graph and *qtcsv* for writing data in a *.csv* file for later processing. The game provides two modes (i.e., high alpha and alpha/theta ratio) corresponding to the logged signals through User Datagram Protocol (UDP) ports. The signals are interpreted such that the byte array received from Simulink is transformed to a decimal number while storing the global smallest and largest values within the current running session. The module further computes the percentage of newly received signal value from the current global smallest and largest values and saves it into a *.csv* file and forwards it to Unity to update the GUI accordingly.

The data analysis module for the survival shooter game has been designed to accept two types of data from the headset: alpha and alpha/theta.

In the **Alpha mode** mode of the data analysis module the alpha band (8–12 Hz) are extracted into four 1 Hz bands:

- $Alpha_1 = 8\text{--}9\text{ Hz}$
- $Alpha_2 = 9\text{--}10\text{ Hz}$
- $Alpha_3 = 10\text{--}11\text{ Hz}$
- $Alpha_4 = 11\text{--}12\text{ Hz}$

The goal of peak alpha training is to increase the activity in the upper two alpha bands while decreasing it in the lower. Thus, we combined all the four channels into the following formula that gives an overall score on how well the training is going:

$$value = -(alpha_1)^2 - alpha_2 + alpha_3 + alpha_4^2$$

Notice that $alpha_1$ and $alpha_2$ have negative signs and that the $alpha_1$ and $alpha_4$ are squared to emphasize the highest and lowest part of the alpha band.

In **Alpha/Theta mode** [5] the data analysis module receives the same input and performs the same transformation, but for computing the final value the following formula is used:

$$value = \frac{alpha_1}{theta_1}$$

where $alpha_1$ and $theta_1$ are the highest alpha and theta signals.

Once the current and preceding values are determined, the percentage may be further computed and forwarded to the game such that the halo lightning intensity and the damage output of the character may be updated.

The value for the **percentage** from the current input is computed taking into account *min*, *MAX* values with the following formula:

$$percentage = \frac{currentValue - min}{MAX - min} * 100$$

If parasite values (i.e., values smaller or larger than a specific threshold value) occur then these values are discarded. When the values received are not parasite but are larger than MAX or smaller than min then the MAX and min values are updated and the returned percentage is 100 or 0, respectively.

In-game damage output and halo are updated with the following procedures:

Algorithm 1. updateGunPower

```

if  $newPercentage \leq (oldPercentage + sensitivity)$  then
     $gunPower \leftarrow (newPercentage - oldPercentage) / sensitivity$ 
else
    if ( $power \geq 0$ ) then
         $power \leftarrow power - 1$ 
    end if
end if
 $oldPercentage \leftarrow newPercentage$ 

```

Where the variables have the following meaning: **power** is the damage output by the gun (also displayed in game), **sensitivity** is a constant number that can be set in the options of the game whose purpose is to decrease or increase the amount of brain activity variation needed in order to see changes in game, **new percentage/old percentage** hold the last signal received on the UDP channel as well as the current one so that a comparison between them can be made.

Algorithm 2. enableHaloVisualEffect

```

 $lightIntensity \leftarrow power$ 
if  $lightIntensity \geq 50$  then
    # enable halo
end if

```

4 Sample Usage Scenario

The goal of the player's character is to constantly fight enemies that spawn around the map, survive them and finally get as many points as possible. The GUI displays a health bar such that when an enemy comes close to the player's health is decreased. When the player's health reaches 0 the game ends. There are several types of enemies, each with a different health amount. At any moment, the damage the player's gun does is influenced by the power, which acts as a multiplier to the player's damage. The power is positively correlated to the NFT through $alpha$ and $theta$ signals.



Fig. 2. GUI representing the difference between minimal and maximal brain activity

Figure 2 presents the GUI in two sample usage scenarios. The above figure presents the GUI for minimal brain activity while below figure presents the GUI for high brain activity.

The player mounts the headset and starts the game. Once the game starts it receives the UDP datagrams containing *alpha* and *theta* signals. At this step, the shooter's strength and aura have default values and are not altered by the EEG signals. Once an increase in the *percentage* value is detected the GUI is updated and a larger strength and aura are being displayed. In a similar way, a decrease in the *percentage* value triggers a decrease in the power of the shooter. The update of the strength and aura within the GUI is performed every second according to the computed *percentage* value.

Another factor that is taken into account is the intensity of the light. When large values in light intensity take place the aura is enabled. The aura is disabled when the surrounding light has a value under a threshold value of 50%. The surrounding light and aura are key elements along shooter's strength in order to provide the player with visual feedback of their current neurofeedback training status and progress.

5 Conclusions

In this paper, we described a design for a neurofeedback training game that allows the user to gain the benefit of neurofeedback while merely playing a

game. The design provides a framework and design principles on how any game could be turned into a potential neurofeedback training application. However, several questions remain that we aim to address in future work. First of all, we need to validate that the free-form game neurofeedback training works as well as “dedicated” neurofeedback setup without the potential distractor of the game. Also, we aim to study whether there exist interpersonal differences in the usefulness of different neurofeedback protocols (such as the alpha/theta vs. peak alpha). There are also many other potential training possibilities such as beta and gamma training that has shown potential for enhancing cognitive control [12].

Another practical question is to determine the bare minimum hardware for the training to be feasible: is it necessary to have a high-quality laboratory grade EEG device with 64 or even more sensors, or might a commercial grade device with only one or two sensors be capable of providing the same benefits. This has huge economic ramifications as most people are not able to afford a laboratory grade equipment yet there exist several lower quality commercial devices in the market. It is also important to determine how long and how many times the training has to be repeated to gain maximum benefit. Similarly, it is necessary to determine how long the effects last after the feedback training.

References

1. Angelakis, E., Stathopoulou, S., Frymiare, J.L., Green, D.L., Lubar, J.F., Kounios, J.: EEG neurofeedback: a brief overview and an example of peak alpha frequency training for cognitive enhancement in the elderly. *Clin. Neuropsychol.* **21**(1), 110–129 (2007)
2. Arns, M., Heinrich, H., Ros, T., Rothenberger, A., Strehl, U.: Neurofeedback in ADHD. *Front. Hum. Neurosci.* **9**, 602 (2015)
3. Bazanova, O.M., Auer, T., Sapina, E.: On the efficiency of individualized theta/beta ratio neurofeedback combined with forehead emg training in ADHD children. *Front. Hum. Neurosci.* **12**, 3 (2018)
4. Davelaar, E.J.: Mechanisms of neurofeedback: a computation-theoretic approach. *Neuroscience* **378**, 175–188 (2017)
5. Egner, T., Strawson, E., Gruzelier, J.H.: EEG signature and phenomenology of alpha/theta neurofeedback training versus mock feedback. *Appl. Psychophysiol. Biofeedback* **27**(4), 261 (2002)
6. Escolano, C., Navarro-Gil, M., Garcia-Campayo, J., Congedo, M., De Ridder, D., Minguez, J.: A controlled study on the cognitive effect of alpha neurofeedback training in patients with major depressive disorder. *Front. Behav. Neurosci.* **8**, 296 (2014)
7. Fernández, T., et al.: EEG and behavioral changes following neurofeedback treatment in learning disabled children. *Clin. EEG Neurosci.* **34**(3), 145–152 (2003). <https://doi.org/10.1177/155005940303400308>
8. Ghasemian, M., Taheri, H., Kakhki, A.S., Ghoshuni, M.: The effect of alpha neurofeedback training on motor skill acquisition. *Biosci. Biotechnol. Res. Asia* **13**(3), 1651–1656 (2016)

9. Gruzelier, J.H.: EEG-neurofeedback for optimising performance. I: a review of cognitive and affective outcome in healthy participants. *Neurosci. Biobehav. Rev.* **44**, 124–141 (2014)
10. Hsueh, J.J., Chen, T.S., Chen, J.J., Shaw, F.Z.: Neurofeedback training of EEG alpha rhythm enhances episodic and working memory. *Hum. Brain Mapp.* **37**(7), 2662–2675 (2016)
11. Jurewicz, K., Paluch, K., Kublik, E., Rogala, J., Mikicin, M., Wróbel, A.: EEG-neurofeedback training of beta band (12–22 Hz) affects alpha and beta frequencies—a controlled study of a healthy population. *Neuropsychologia* **108**, 13–24 (2018)
12. Keizer, A.W., Verment, R.S., Hommel, B.: Enhancing cognitive control through neurofeedback: a role of gamma-band activity in managing episodic retrieval. *Neuroimage* **49**(4), 3404–3413 (2010)
13. Kober, S.E., et al.: Specific effects of EEG based neurofeedback training on memory functions in post-stroke victims. *J. Neuroeng. Rehabil.* **12**(1), 107 (2015)
14. Kosunen, I., Salminen, M., Järvelä, S., Ruonala, A., Ravaja, N., Jacucci, G.: Relaworld: neuroadaptive and immersive virtual reality meditation system. In: *Proceedings of the 21st International Conference on Intelligent User Interfaces*, pp. 208–217. ACM (2016)
15. Kouijzer, M.E., de Moor, J.M., Gerrits, B.J., Buitelaar, J.K., van Schie, H.T.: Long-term effects of neurofeedback treatment in autism. *Res. Autism Spectr. Disord.* **3**(2), 496–501 (2009). <https://doi.org/10.1016/j.rasd.2008.10.003>
16. Kuikkaniemi, K., Laitinen, T., Turpeinen, M., Saari, T., Kosunen, I., Ravaja, N.: The influence of implicit and explicit biofeedback in first-person shooter games. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 859–868. ACM (2010)
17. Lubar, J., Swartwood, M., Swartwood, J., O'Donnell, P.: Evaluation of the effectiveness of EEG neurofeedback training for ADHD in a clinical setting as measured by changes in T.O.V.A. scores, behavioral ratings, and WISC-R performance. *Biofeedback Self-regul.* **20**(1), 83–99 (1995). <https://doi.org/10.1007/BF01712768>
18. Marzbani, H., Marateb, H.R., Mansourian, M.: Neurofeedback: a comprehensive review on system design, methodology and clinical applications. *Basic Clin. Neurosci.* **7**(2), 143 (2016)
19. Mierau, A., et al.: The interrelation between sensorimotor abilities, cognitive performance and individual EEG alpha peak frequency in young children. *Clin. Neurophysiol.* **127**(1), 270–276 (2016)
20. Pineda, J.A., Silverman, D.S., Vankov, A., Hestenes, J.: Learning to control brain rhythms: making a brain-computer interface possible. *IEEE Trans. Neural Syst. Rehabil. Eng.* **11**(2), 181–184 (2003)
21. Stein, A., Yotam, Y., Puzis, R., Shani, G., Taieb-Maimon, M.: EEG-triggered dynamic difficulty adjustment for multiplayer games. *Entertain. Comput.* **25**, 14–25 (2018)
22. Serman, M., Egner, T.: Foundation and practice of neurofeedback for the treatment of epilepsy. *Appl. Psychophysiol. Biofeedback* **31**(1), 21–35 (2006). <https://doi.org/10.1007/s10484-006-9002-x>



Taking e-Assessment Quizzes - A Case Study with an SVD Based Recommender System

Oana Maria Teodorescu^{1,2(✉)}, Paul Stefan Popescu¹,
and Marian Cristian Mihaescu¹

¹ University of Craiova, Craiova, Romania
`teodorescu_oanamarina@yahoo.com`

² NetRom Software, Craiova, Romania
`{spopescu,mihaescu}@software.ucv.ro`

Abstract. Recommending learning assets in e-Learning systems represents a key feature. Among many available assets there are quizzes that validate and also evaluate learner's knowledge level. This paper presents a recommender system based on SVD algorithm that is able to properly recommend quizzes such that learner's knowledge level is evaluated and displayed in real time by means of a custom designed concept map for *graphs* algorithms within the *Data Structures* course. A preliminary case study presents a comparative analysis between a group of learners that received random quizzes and a group of learners that received recommended questions. The visual analytics and interpretation of two representative cases show a clear advantage of the students who received recommended questions over the other ones.

Keywords: SVD · Recommender systems · Quizzes · e-Learning

1 Introduction

E-assessments are an important part of any e-learning platform, as they test the learner's accomplishment of meeting the learning objectives. They are useful in two directions, for both the learner and the professor. One of them is for strengthening one's knowledge (involving primarily the learner's objective when studying a course) and the other is evaluating the learner's comprehension of the course (involving both the learner and the professor, as a means of self-evaluation and progress tracking versus progress evaluation and defectiveness of learning materials).

Therefore, development of knowledge evaluation and tracking methods is an issue that gets continuous attention in educational data mining. One of the paper's purpose is to investigate a visualization approach of the student's knowledge by means of concept maps and question coverage. Thus, several visualization methods accompanied by a SVD based recommendation algorithm were

proposed. Firstly, for each student, we design a student's concept map which presents concepts and their coverage in percentages. Based on available weighted concept maps, a student's time-line is constructed for one or more tests (i.e., pool of quizzes), such that the evolution in time of the concept's coverage in percentages may be easily visualized.

The scope of this paper is the evaluation of the effectiveness of the test question recommender system implemented in the Tesys platform for multiple types of questions. This will be done by studying and comparing the results and knowledge gained by learners taking tests using the random vs recommender strategy. Furthermore, an objective function will be defined for each learner to visualize the unique path taken by each individual when learning new concepts. This function will also be used to validate the recommender's effectiveness against the random-pick strategy. The function will result from creating a concept map for the concepts of a particular subject and by mapping them, in a many-to-many relationship, to the questions designed for the self-testing purpose of the learner for each chapter of that subject. This also allows for the tracking and evaluation of a student's progress by both learner and professor.

2 Related Work

Tesys [2] is a web e-learning platform running for the students of the University of Craiova enrolled in distance education. It provides the resources needed by students, professors and secretaries to perform their activity.

D3.js (or just D3 for Data-Driven Documents) is a JavaScript library for producing dynamic, interactive data visualizations in web browsers. In contrast to many other libraries, D3.js allows great control over the final visual result [14].

A concept map or conceptual diagram is a diagram that depicts suggested relationships between concepts, which are defined as "perceived regularities or patterns in events or objects, or records of events or objects, designated by a label" and are depicted as shapes in the diagram [12]. It is a graphical tool that instructional designers, engineers, technical writers, and others use to organize and structure knowledge [8].

Content-based recommendation systems analyze item descriptions to identify items that are of particular interest to the user [7]. In a content-based recommender system, keywords are used to describe the items and a user profile is built to indicate the type of item this user likes. This approach has its roots in information retrieval and information filtering research [5].

Collaborative Filtering (CF) is a subset of algorithms that exploit other users and items along with their ratings (selection, purchase information could be also used) and target user history to recommend an item that target user does not have ratings for. CF differs itself from content-based methods in the sense that user or the item itself does not play a role in recommendation but rather how(rating) and which users(user) rated a particular item. (Preference of users is shared through a group of users who selected, purchased or rate items similarly) [3].

An example of use of hybrid filtering is the American global provider of streaming films and television series Netflix, which make recommendations by comparing the watching and searching habits of similar users (i.e. collaborative filtering) as well as by offering movies that share characteristics with films that a user has rated highly (content-based filtering) [10]. Basic techniques for recommender systems (collaborative, content-based, knowledge-based, and demographic techniques) have known shortcomings such as the well known cold-start problem for collaborative and content-based systems (what to do with new users with few ratings) and the knowledge engineering bottleneck [9] in knowledge-based approaches, as Wikipedia states in [10].

According to an MIT tutorial for SVD (Singular Value Decomposition) [4], calculating the SVD for a matrix M consists of finding the eigenvalues and eigenvectors of MM^T at the power of T and $M^T M$ at the power of T multiplied by M . The eigenvectors of MM^T at the power of T multiplied by M make up the columns of V , the eigenvectors of $M^T M$ at the power of T make up the columns of U . Also, the singular values in Σ are square roots of eigenvalues from MM^T at the power of T or $M^T M$ at the power of T multiplied by M . The singular values are the diagonal entries of the Σ matrix and are arranged in descending order. The singular values are always real numbers. If the matrix M is a real matrix, then U and V are also real.

Recommender systems for e-Learning platforms are based on many approaches like web mining and information retrieval [6], recommender systems based on the context [13] or even using intelligent agents [15]. One interesting approach of using collaborative filtering in e-Learning systems [1] was to assign greater weights for users with higher knowledge than the users with lower knowledge and to obtain that the authors propose some new equations in the nucleus of the memory-based collaborative filtering. Another interesting paper, presenting clear results regarding recommender systems in smart e-Learning environments presents their approach [11] along with their encouraging results and their aim to extend the system for more faculties.

3 Proposed Approach

One of the goals of this project is to offer the users of Tesys a personalized experience by the ability of the Tesys e-learning platform to adapt to the students' individual needs in order to enhance the effectiveness of the learning process. For achieving this, a recommender system for choosing questions to be included in a test for a specific subject and chapter has been used. The algorithm used for the recommender system is SVD. The unique path taken by each student in the learning process of a subject is displayed using an evolution concept map based on the concepts covered in questions answered in a self-testing environment.

The following question types were defined: *Matching question* - given two columns, A and B, with options on both sides, match the corresponding item from column A with the one on column B given a certain criteria stated in the question, *Short-answer question* - given a phrase, fill in the gaps with a short

answer (word or group of words), **Numerical question** - given a phrase, fill in the gaps with a numerical value (may be either an integral value or a real number), **True/False question** - given a statement, assert its truth value. Therefore, we have designed an SVD-based recommender system that performs in the following way. The procedure is presented below:

Algorithm 1. Recommendation algorithm using SVD

Apply SVD algorithm on matrix $M = (m_{(i,j)})$ of size $n \times m$ which generates matrices $U = (u_{(i,j)})$ of size $n \times k$ and $V = (v_{(i,j)})$ of size $k \times m$.

if the student is new (has taken no tests for the subject and chapter so far) **then**

 Construct user feature row as a row vector L of size $1 \times k$ filled with values of 1.

else

 Extract user feature row as a row vector L of size $1 \times k$ from the user-features matrix U .

end if

for each question feature column vector C **in** the question-features matrix V (index i) **do**

 Perform the dot product p between L and C .

 Save dot product p value to question recommendation vector along with its index i as key.

end for

Sort question recommendation vector ascending with respect to values, then keys.

Select the first recommended questions that were not previously answered by the student.

if the number of required questions has not been reached **then**

 Select from a list of unanswered questions by any student.

end if

if the number of required questions has not been reached **then**

 Select the last recommended questions (they were previously answered by student, thus selecting the ones the current student or others answered mostly incorrectly).

end if

Input. The logical input of our algorithm consists of students' results to tests, or more specifically, to questions in tests. We collect this information based on subject and chapter (by computing the mean grade, value normalized on a scale of 0 to 1, for each answered question for each student) and process it to a form which is understandable to the algorithm we chose for recommendation (singular value decomposition). Therefore, the actual (processed) input of our algorithm consists of: **A list of n students** taken into account for the recommendation process (represented, in our implementation, by the list of student identifiers). **A list of m questions** taken into account for the recommendation process, that have been answered by students (represented, in our implementation, by the list of question identifiers). **A matrix $M = (m_{(i,j)})$** of size $n \times m$, having values in the range $[0, 1]$ which represent the accuracy of the student i when answering the question j (a measure computed by normalizing the average grade of a student for a question by the maximum grade for that question).

Output. The output of the algorithm consists of a list of questions recommended to be included in a test taken by a student for a specific subject. The aim is to provide personalized results that help the student to discover areas which need to be improved in the learning process of a particular subject.

The Recommendation Mechanism. The algorithm relies on the SVD technique which decomposes a matrix into two matrices U and V in a lower dimensional feature space. The decomposition is used to achieve a separation of the user and question models into features that identify individual attributes of the students and questions whose associations give their uniqueness.

4 Experimental Results

4.1 Usage of SVD Method for Recommending Questions

If M is the initial matrix of $m \times n$ dimension, the algorithm finds U of $m \times m$, Σ of $m \times n$ and V of $n \times n$ dimensions that multiplied give an approximation of the initial M matrix. The U matrix represents the strength of the associations between a user and the features in the hidden feature space while the V matrix represents the strength of the associations between an item and the features in the hidden feature space. Σ is a diagonal matrix.

In our case, the users are students and items are questions for a specific subject and chapter.

$$U = \begin{pmatrix} 0.44 & -0.22 & -0.90 \\ 0.66 & 0.69 & 0.30 \\ 0 & 0 & 0 \\ 0.61 & -0.73 & 0.31 \end{pmatrix} V = \begin{pmatrix} 0.75 & -0.07 & -0.66 \\ 0.55 & -0.50 & 0.67 \\ 0.38 & 0.86 & 0.33 \end{pmatrix}$$

With these two matrices, we will select a number of q questions for a student at the request of a test on a specific chapter. We identify two cases:

1. student has already answered some questions
2. student is new and the system knows nothing about him (no row data in the matrix).

These two cases will be explained separately in the following sub-chapters.

Selecting Questions for an Existing Student

In the case of an existing student, matrix U contains user-features information and the row corresponding to the user can be used to compute the recommendation vector. If the user row in the U matrix is L , then compute the final vector by multiplying it with the V vector containing question-features information through the features space (Σ).

Let's say we want to recommend 2 questions for a test to student. In this case and the conditions of the example above, the computed L and R matrix are:

$$L = (0.61 \ -0.73 \ 0.31)$$

$$R = L * \Sigma * V = (0.61 \ -0.73 \ 0.31) * \Sigma * \begin{pmatrix} 0.75 & -0.07 & -0.66 \\ 0.55 & -0.50 & 0.67 \\ 0.38 & 0.86 & 0.33 \end{pmatrix} = (0.30 \ 0.91 \ -0.29)$$

By sorting the values ascending in the R vector and recalling the previous indices, the system can take the first questions for our student as the recommendation. The result will select more similar questions with respect to the features identified which have not been answered by the student or, in case the number of questions needed is higher than the number of questions the student didn't answer, it will take questions from a list of questions unanswered by any student (new questions), and if the number of questions needed for the test still hasn't been reached, it takes the last values in the ascending ordering. The reason to do this is we want to select the questions that the student mostly failed to answer.

In the above example, the top 2 recommended questions are the third and first questions.

Selecting Questions for a New Student

In the case of a new student, nothing is known about his/her skills (he/she has taken no tests). This is called cold-start. But from the V matrix, the features of each question were extracted from the experience with the other students who already answered some questions.

To deal with this problem, the user row L has been filled only with values of 1. This indicates that the features have equal probability for the user and the system will rely solely on how these features impact the questions in the recommendation. By multiplying this line with each question's column from V through the features space (Σ), for each question a value is obtained, thus leading to a one dimensional matrix R.

In our example, the computed L and R matrices are:

$$L = (1 \ 1 \ 1)$$

$$R = L * \Sigma * V = (1 \ 1 \ 1) * \Sigma * \begin{pmatrix} 0.75 & -0.07 & -0.66 \\ 0.55 & -0.50 & 0.67 \\ 0.38 & 0.86 & 0.33 \end{pmatrix} = (0.01 \ 0.72 \ 1.58)$$

By sorting the values ascending in the R vector and recalling the previous indexes, the first questions are taken for our student as the recommendation.

The result of this approach is the questions that most students answered correctly with a higher probability have priority, so the questions selected to be used to test our new student will test his/her basic skills first and then adjust the difficulty after more tests have been taken. In the above example, the top 2 recommended questions are the third and second questions.

4.2 Case Study Using Recommender System and Concept Map

A case study has been conducted on bachelor students in 3-rd semester at Computers to assess the recommender system's efficiency. A set of 98 questions were added for the self-testing of students in the Tesys platform for a chapter module

about Graphs at Data Structures course. Furthermore, each question has been associated a concept within a custom built concept map whose structure can be seen in Fig. 1.

Students were divided in two groups: some were using the platform with the random approach while others were using the recommender system for picking the questions for the tests. A number of 50 students participated at trial, 21 were randomly selected to receive random questions and 29 received questions from the recommender system. A total number of 140 tests were taken in total.

Figure 1 presents the current status of concepts in five tests with recommended quizzes. As it may be observed, some nodes are becoming orange or even yellow, as the number of correctly answered question covers a higher percentage from the entire number of questions assigned to a specific concept. For example, concept *GSearch* is covered half after the 4th test and becomes green after the 5th. Visual analytics of the concept coverage reveals a smooth coloring of concepts from red to orange to yellow and finally to green without major red areas between internal nodes displaying progress (i.e., orange, yellow or green).

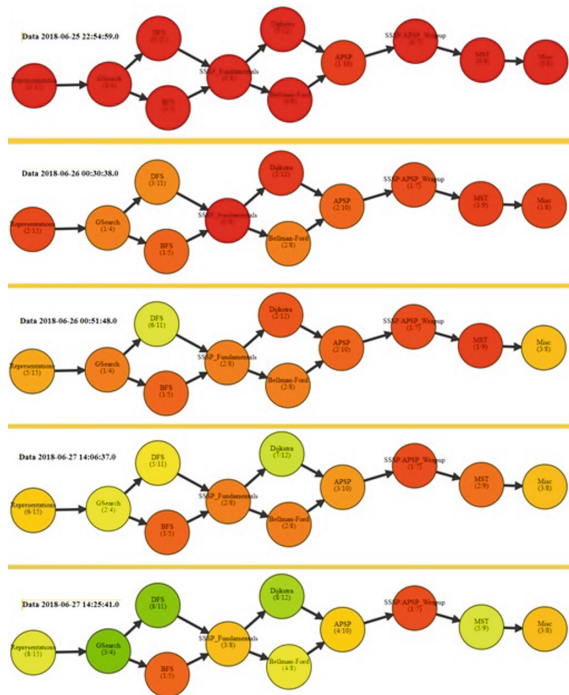


Fig. 1. Five tests evolution with recommended quizzes (Color figure online)

5 Conclusions

The paper presents an SVD-based recommender system that offers students the possibility to visualize their knowledge level in a weighted graph of concepts. We defined a concept map as a graph where each node represents a concept which has an associated weight representing the knowledge as a percentage of correctly answered questions from the number of available questions at that concept. The concept map was designed for *Graphs* chapter within the *Data Structures* discipline. The main goal of the recommender system is to provide a personalized set of questions depending on their current status (i.e., correctly and incorrectly questions answered so far) and the status of all other colleagues that have previously taken tests. Visual analytics of the experimental results in terms of impact of recommendation procedure in knowledge coverage of the concept map show promising initial results. Further work needs to be performed in terms of defining and integrating appropriate quantitative and qualitative metrics for measuring accumulated knowledge with and without the usage of the recommender system.




References

1. Bobadilla, J., Serradilla, F., Hernando, A.: Collaborative filtering adapted to recommender systems of e-learning. *Knowl.-Based Syst.* **22**(4), 261–265 (2009)
2. Burdescu, D.D., Mihaescu, M.C.: Tesys: e-learning application built on a web platform. In: ICE-B, pp. 315–318 (2006)
3. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst. (TOIS)* **22**(1), 5–53 (2004)
4. Hoekstra, R.: The knowledge reengineering bottleneck. *Semant. Web* **1**(1, 2), 111–115 (2010)
5. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: *Recommender Systems: An Introduction*. Cambridge University Press, Cambridge (2010)
6. Khribi, M.K., Jemni, M., Nasraoui, O.: Automatic recommendations for e-learning personalization based on web usage mining techniques and information retrieval. In: Eighth IEEE International Conference on Advanced Learning Technologies, ICALT 2008, pp. 241–245. IEEE (2008)
7. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. *Computer* **8**, 30–37 (2009)
8. Novak, J.D., Cañas, A.J.: The theory underlying concept maps and how to construct and use them (2008)
9. Ricci, F., Cavada, D., Mirzadeh, N., Venturini, A., et al.: Case-based travel recommendations. In: *Destination Recommendation Systems: Behavioural Foundations and Applications*, pp. 67–93 (2006)
10. Shani, G., Heckerman, D., Brafman, R.I.: An MDP-based recommender system. *J. Mach. Learn. Res.* **6**(Sep), 1265–1295 (2005)
11. Soonthornphisaj, N., Rojsattarat, E., Yim-ngam, S.: Smart e-learning using recommender system. In: Huang, D.S., Li, K., Irwin, G.W., et al. (eds.) *Computational Intelligence*. LNCS (LNAI), vol. 4114, pp. 518–523. Springer, Heidelberg (2006). https://doi.org/10.1007/978-3-540-37275-2_63

12. Sowa, J.F.: Conceptual structures: information processing in mind and machine (1983)
13. Verbert, K., et al.: Context-aware recommender systems for learning: a survey and future challenges. *IEEE Trans. Learn. Technol.* **5**(4), 318–335 (2012)
14. Viau, C.: What’s behind our business infographics designer? d3.js of course (2012)
15. Zaïfane, O.R.: Building a recommender agent for e-learning systems. In: *Proceedings of the International Conference on Computers in Education*, pp. 55–59. IEEE (2002)



Towards Complex Features: Competitive Receptive Fields in Unsupervised Deep Networks

Richard Hankins^(✉) , Yao Peng , and Hujun Yin 

School of Electrical and Electronic Engineering, The University of Manchester,
Manchester, UK

{richard.hankins,yao.peng,hujun.yin}@manchester.ac.uk

Abstract. In this paper we propose a simple unsupervised approach to learning higher order features. This model is based on the recent success of lightweight approaches such as SOMNet and PCANet to the challenging task of image classification. Contrary to the more complex deep learning models such as convolutional neural networks (CNNs), these methods use naive algorithms to model the input distribution. Our endeavour focuses on the self-organizing map (SOM) based method and extends it by incorporating a competitive connection layer between filter learning stages. This simple addition encourages the second filter learning stage to learn complex combinations of first layer filters and simultaneously decreases channel depth. This approach to learning complex representations offers a competitive alternative to common deep learning models whilst maintaining an efficient framework. We test our proposed approach on the popular MNIST and challenging CIFAR-10 datasets.

Keywords: Self-organising maps · Unsupervised learning
Deep learning · Representation learning · Receptive fields · Pooling

1 Introduction

The classification of images is a very challenging task that has received considerable attention for the last few years due to many potential applications. Good feature representations must discriminate between independent classes whilst being robust to many intra-class variations. Approaches to this problem generally fall into two categories. Traditionally, hand-crafted methods were prevalent with models such as SIFT and HOG performing well. However, these methods struggled when applied to other tasks and therefore each new problem would involve the design of new methodologies. Recently, using deep models to learn representations from both labelled and unlabelled data has made much progress with models such as the supervised convolutional neural network (CNN) [21] and its many variants have achieved the current state-of-the-art on a number of tasks [4, 19]. Key to the success of deep models are their inherent ability to learn higher level data specific features in contrast to the low-level features of hand crafted

techniques. However, whilst successful, deep learning models can be difficult to train as they are burdened by many parameters which need expert fine-tuning. These parameters are in addition to the architecture parameters such as receptive field size, feature number and depth. Furthermore, supervised approaches require vast labelled datasets, which present problems with annotations.

Recently, PCANet [3] showed a simple approach to deep learning that used a limited filter bank and could perform as good as some deep learning models whilst requiring less configuration and parametrization. SOMNet [14] and CUNet [11] used the self-organizing map (SOM) and K-means algorithms respectively to train filter bank “dictionaries” and adopt similar efficient pipelines to that of PCANet. In addition, SOMNet introduced an alternative encoding strategy which enabled the use of additional filters for a more over-complete solution which aided performance. These methods demonstrated that unsupervised approaches could compete with more complex supervised methods. DCTNet [26] showed that generated features could also perform well. However, these methods learn similar features replicated over multiple levels which visually do not concur with higher level representations of the input. This is because the second layer of these methods do not learn from combinations of first layer features. In comparison, common deep learning approaches build feature hierarchies over many layers with features from previous layers combining to form more complex features in subsequent ones. Early CNNs used a parsimonious approach when establishing the connections between layers [22], however, in recent years it has become standard to use full connections [19, 29]. Yet, this requires that the features in a given layer have a depth equal to the number of features in the preceding layers. Some recent advances have worked on 1×1 convolutions [24] between layers for which only a linear weighting is learned which, dramatically reduces the dimensions of the features, reducing redundancy, and forces the network to learn combination of features instead of offsets.

Inspired by this we propose a novel approach which selects local receptive fields in order to learn data-dependent higher order features from more primitives features in lower layers. To this end we utilize a two layer SOMNet architecture and introduce a new layer, placed between the two layers of the SOMNet. This layer combines the activations of the previous layer which allows SOMNet to learn hierarchical features without increasing their dimensionality. Our approach demonstrates competitive results on the handwritten digit recognition task and only uses the simple competitive SOM algorithm throughout. In contrast to the often complex deep learning models our approach avoids tuning millions of parameters, for an elegant and time efficient solution capable of handling vast amounts of data without the need for annotations.

2 Related Work

Vector quantization (VQ) has been used extensively for computer vision tasks [9, 20]. Algorithms such as SOM and K-means are used to learn a visual dictionary of low level features (feature learning) which are then used to map or encode

the input into higher level image representations (feature extraction). K-means has proved to be an efficient and competitive algorithm in recent literature when combined with appropriate data pre-processing and encoding [6]. In [5] a single layer network demonstrated competitive performance on a number of datasets belying its simplicity. The self-organizing map [17,31] can be considered as a topological preserving alternative to K-means. It uses competitive learning to quantize an input distribution while maintaining a topographic structure. This neighbourhood preservation makes SOM more immune to initialization and outliers than K-means. Feature learning with SOM has been explored for face [1] and handwritten digit [1,8] recognition tasks.

Chan et al. [3] proposed a simple framework named PCANet that used a novel approach to deep learning characterized by minimal filters and less parametrization. Filter banks for each layer were trained by applying PCA to the input. Once learned, the filter banks were used in a traditional feed forward convolution architecture as is commonplace in deep learning structures. The authors used a binarization technique called Binarized Statistical Image Features (BSIF) as introduced by Kamala and Rahtu [16]. Once binarized, the activations were split into sub-regions and local histograms were formed, which were concatenated to form the final feature vector. SOMNet [14] used the self-organising map algorithm to learn the filters which alleviates the constraints imposed by the use of PCA, namely orthogonality and the size of filters which is limited by the covariance matrix. In addition, the size of the filters also governs their number and the mixture of low and high frequency components in the filter bank. Unburdened by these limitations, SOMNet also adapted the binarization process to enable the use of further filters. CUNet [11] used K-means to learn the filter banks and introduced a new weighted pooling method to further improve classification accuracy. DCTNet [26] used generated discrete cosine transform basis functions as filters.

Feature learning has always garnered a lot of attention. Yet, in recent years work has also focused on the importance of the connections between layers. In [6] it was shown that the choice of encoding scheme for a single layer network is actually much more important than the choice of dictionary learning method. Considering that the connections between layers are a form of encoding this presents a worthwhile line of inquiry. In addition, it has been demonstrated that features trained using K-means perform poorly in later layers when using a full connection [7]. Furthermore, connection schemes that provide less than a full connection can ultimately reduce dimensionality and redundancy in the feature space. There have been many recent attempts to group features in unsupervised learning. In [10] and [7] it was shown that grouping features randomly or by similarity can aid performance. Dundar et al. [12] suggested learning 1×1 convolutions across channels between unsupervised layers using gradient descent. However, they proposed grouping the features randomly first to allow the network to learn the best combinations.

3 Methodology

3.1 Self-organizing Map

Let each unit i in the SOM denote a reference vector $\mathbf{w}_i = [w_1, w_2, \dots, w_z]^T \in \mathbb{R}^{s^2 d}$ with equal dimension as the input vector. Prototype vectors or neurons are commonly arranged in a 2D grid.

In the case that the convolution is used as the similarity measure the winner or best matching unit at each time step is found by maximizing the activation between the input $\mathbf{x}(t)$ and all the neurons of the map:

$$bmu(t) = \arg \max_{i \in \Omega} [\mathbf{w}_i * \mathbf{x}(t)] \quad (1)$$

and weights of the winner and its neighbours are updated according to

$$\Delta \mathbf{w}_i(t) = \frac{\alpha(t) + \eta(bmu, i, t) [\mathbf{x}(t) - \mathbf{w}_i(t)]}{\|\alpha(t) + \eta(bmu, i, t) [\mathbf{x}(t) - \mathbf{w}_i(t)]\|} \quad (2)$$

where $*$ denotes convolution, Ω is the set of neuron indices, α ($0 < \alpha(t) < 1$) is the monotonically decreasing learning rate, $\eta(bmu, i, t) = \exp[-\frac{\|\mathbf{r}_{bmu} - \mathbf{r}_i\|^2}{2\sigma(t)^2}]$ is the neighbourhood function with \mathbf{r}_i being the location of neuron i on the map and σ the effective range of the neighbourhood, which decreases with time t . The neighbourhood is important to the function of the SOM and helps avoid sensitivity to initialization and outliers that other clustering algorithms such as K-means can suffer from [1]. In this work to avoid redundancy in the feature space we decrease the neighbourhood to a very small value towards the end of training and use a 1D map.

3.2 Self-organizing Map Network (SOMNet)

As shown in Fig. 1 the structure of SOMNet closely resembles PCANet however we alter the encoding process. What follows details the function of each layer. To learn SOM features randomly sampled $s \times s \times d$ patches are selected from an input image, where d is the number of channels, and then mean normalized by subtracting the patch mean from each pixel. When training subsequent layers a single $s \times s$ patch is sampled from a single activation map uniformly. Each SOM is trained until convergence. Whilst the first SOM is able to reach a global minimum, due to the bootstrap sampling process and with the increased input space of the activations the second layer SOM may only converge to local minima. Yet we note that this does not appear to be detrimental to the performance.

Convolution Layer. Given an $m \times n \times d$ input image \mathbf{I}_d , where m and n are the height and width of the input respectively and d is the number of channels. We convolve the input with a filter bank of size $s \times s \times d$ which remains the same for each layer. Zero padding is applied with pad size $(s - 1)/2$ so that the output is the same size as the input. Convolution of the input \mathbf{I}_d with the filter

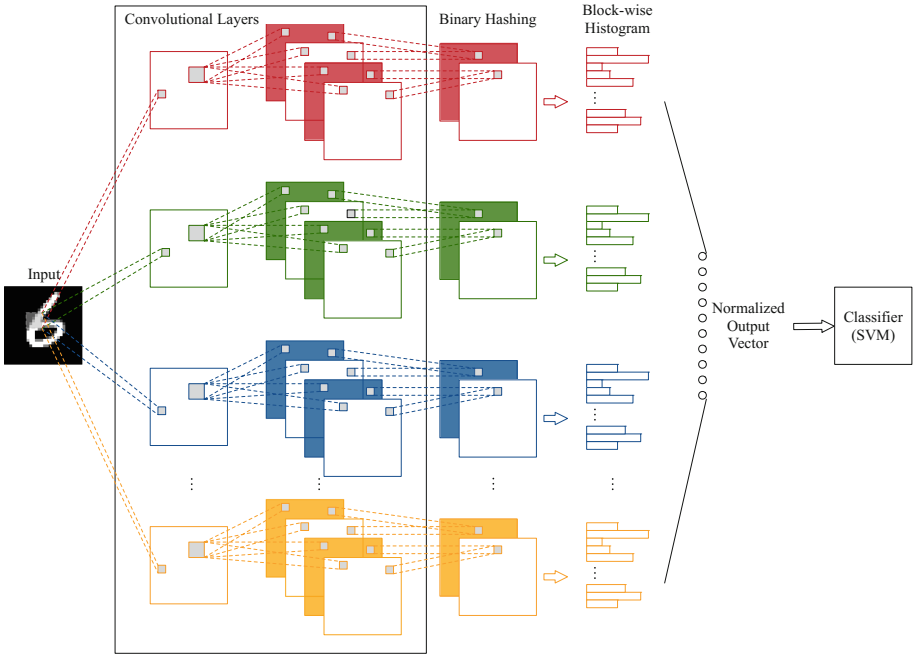


Fig. 1. Block diagram of the proposed method [14].

bank $\mathbf{W}_l^i \in \mathbb{R}^{s \times s \times d}$, $i = 1, 2, \dots, h_l$ where h_l is the number of filters in layer l , provides

$$\mathbf{O}^i = [\mathbf{I}_d * \mathbf{W}_l^i], i = 1, 2, \dots, h_l \tag{3}$$

We take each \mathbf{O}^i as input to subsequent layers.

Binarization and Histograming Layer. The final layer has input size $s \times s \times d \times h_l$. By performing the heaviside step function, $He(\cdot)$, each set of d real-valued outputs is binarized by thresholding the values at zero as $He(\mathbf{O}_j^i)$. Each resultant binary string is grouped into four bit nibbles (hexadecimal), G_ϵ^β , and encoded using

$$\sum_{\beta=1}^4 2^{\beta-1} G_\epsilon^\beta, \epsilon = 1, 2, \dots, \frac{h_l}{4} \tag{4}$$

which produces $\frac{h_l}{4} \times d$ encoded output images where each pixel has the range $[0, 2^4 - 1]$. Each of these images is split into $B_{size} \times B_{size}$ blocks with overlapping ratio τ . The final output feature vector $\theta \in \mathbb{R}^{2^4 \frac{h_l}{4} dB}$ is formed by concatenating each block's histogram together, where B is the number of local histograms. Splitting the images into local histograms encodes spatial information and also some degree of invariance to translations.

3.3 Proposed Method

Our proposed method consists of a two-layer SOMNet with additional layers as described below. We consider SOMNet to have a full utilization between the first and second layers since during the second layer feature learning, activations from the first are sampled uniformly. Both of the new layers described below can be placed between the two layers of SOMNet as shown in Fig. 2. We use these additional layers to encourage the learning of more complex features in the second SOMNet layer by promoting only the most competitive channels for a given spatial location. All layers reduce the dimensionality of the activations from the first layer of SOMNet whilst increasing the complexity. Training involves three steps: feature learning, feature extraction and classification using a linear SVM [13]. Prior to classification we normalize the output feature vectors to unit length. It is noted that the following layers are only used during the feature learning stage.

Fully Aggregated Connections (FAC) Layer. This is a naive approach which simply max pools the activations along the activation dimension to produce a single activation, from which a patch is sampled to learn the second layer filters. Specifically, for a set of activations from the first layer $\mathbf{A} = [\mathbf{O}^1, \dots, \mathbf{O}^{h_1}]$, where $\mathbf{O}^i \in \mathbb{R}^{m \times n}$, we perform pooling at each spatial location (x, y) with $x \in \{1, 2, \dots, m\}$ and $y \in \{1, 2, \dots, n\}$. Therefore only the highest activated channel at each location (x, y) is retained for second layer feature learning.

Grouped Aggregated Connections (GAC) Layer. This layer selects subsets of activations to be pooled by grouping them according to their respective filters position on the map. Since SOM clusters the filters whilst maintaining topology, neighbourhood structure is implicit in the groupings. Specifically, for a given input, the set of activations are grouped into non-overlapping subsets of twos and fours (named GAC_2 and GAC_4 respectively) and then pooled. Again we use max pooling. As with the original SOMNet, a patch is sampled from the resultant pooled activations uniformly.

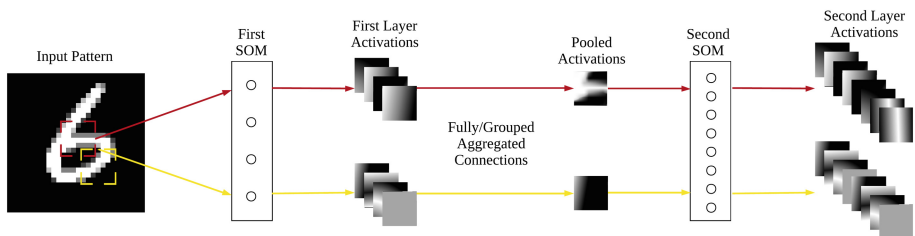


Fig. 2. Proposed feature learning method. Corresponds to the convolutional layers in Fig. 1

Table 1. Error rate on MNIST

Method	Error rate (%)	p-value
CDBN [23]	0.82	–
CSOM (linear SVM) [1]	0.82	–
PCANet-2 [3]	0.66	–
ConvNet [15]	0.53	–
ScatNet-2 (RBF SVM) [2]	0.43	–
MRF-CNN [28]	0.38	–
SOMNet ₈₋₃₂	0.69 ± 0.02	N/A
SOMNet ₈₋₃₂ + FAC	0.57 ± 0.01	0.0007
SOMNet ₈₋₃₂ + GAC ₂	0.66 ± 0.03	0.2230
SOMNet ₈₋₃₂ + GAC ₄	0.61 ± 0.03	0.0184

4 Experiment and Discussion

4.1 Digit Recognition on MNIST

The MNIST [22] dataset consists of 70000 28×28 gray-scale images of handwritten digits 0–9. The training and testing sets contain 60000 and 10000 examples respectively. Table 1 shows our results against methods employing similar pipelines and other state-of-the-art approaches. We used a SOMNet with 8 and 32 filters of size 7×7 in the first and second layers respectively. The remaining parameters were set as $B_{size} = 7$, $\tau = 0.6$. We also added spatial max pooling over 4×4 sub-regions to the output. We find this step improves the performance and reduces the size of the final output feature vector. We selected max pooling as it selects the most competitive features and has been shown to work effectively with linear SVM [30]. We applied whitening prior to training the SOMNet features. This de-correlates the input and enables the SOM to learn more discriminative features. We fine tuned the parameters of the SVM using five-fold cross-validation for all our results. Our method performed as well as or better than other more complex methodologies apart from ConvNet (trained supervisedly), ScatNet (which used an RBF SVM) and MRF-CNN (which used a deeper architecture with many more parameters). The proposed approach improved over the SOMNet baseline for all three configurations. The significance of the top two results, when FAC and GAC₄ are used, is demonstrated to be significant at 1% and 5% levels respectively.

4.2 Object Recognition on CIFAR-10

The CIFAR-10 [18] dataset consists of 60000 32×32 colour images of 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck. There are 5000 training and 1000 testing examples per class. Although each image contains an instance of its respective class there is much intra-class variation. Through

experimentation we find the optimal parameters of SOMNet to be $h_1 = 40$, $h_2 = 32$, $s_1 = s_2 = 5$, $B_{size} = 8$ and $\tau = 0.5$. We divide the output into 4×4 sub-regions and apply max pooling to give 16 pooled histograms. In order to simplify spatial pooling of the block histograms bins we zero pad the final activation maps to produce an even sized output. The results for SOMNet as well as PCANet, CUNet and other state-of-the-art methods are shown in Table 2. In order to speed up the experimentation process we only trained the SVM using a subset of data by uniformly sampling 1000 examples from each class to compare the application of the different layers and fine tune parameters. We then scale up the training to the full dataset. Results on the subset can be found in the second section of the table and the results on the full set in the final section. We fine tune the parameters of the SVM using five-fold cross-validation for all subset results. From the results, we only see improvement when the filters are grouped in twos on the subset. We do not see the same improvement when we replicate the experiment on the full set. This is possibly due to the input space being inherently more complex compared to MNIST, and therefore, the combination of too many filters diminishes the ability of the resultant filters learned in the second layer, to discern useful features. A qualitative analysis revealed that the filters do not appear more complex when the proposed method is used compared to the baseline. However, perhaps the improvement observed on the subset is due to reduced redundancy in the second layer induced by the competitive aggregation. Yet, statistically, no strong conclusions can be made.

Table 2. Accuracy on CIFAR-10

Method	Accuracy (%)
Tiled CNN [27]	73.10
PCANet-2 [3]	77.14
K-means (Triangle, 4000 features) [5]	79.60
CUNet + Weighted Pooling [11]	80.31
RF Learning [7]	82.0
NOMP-20 [25]	82.9
Stochastic Pooling ConvNet [32]	84.87
NIN + Dropout [24]	89.59
MRF-CNN [28]	91.25
SOMNet ₄₀₋₃₂	73.25 ± 0.19
SOMNet ₄₀₋₃₂ + FAC	72.19 ± 0.15
SOMNet ₄₀₋₃₂ + GAC ₂	73.33 ± 0.29
SOMNet ₄₀₋₃₂ + GAC ₄	72.98 ± 0.31
SOMNet ₄₀₋₃₂	78.51 ± 0.06
SOMNet ₄₀₋₃₂ + FAC	77.56 ± 0.12
SOMNet ₄₀₋₃₂ + GAC ₂	78.13 ± 0.35
SOMNet ₄₀₋₃₂ + GAC ₄	77.98 ± 0.35

5 Conclusion and Future Work

This paper presents a novel approach to representation learning. The proposed method can build complex representations based on combinations of previous layer features. This approach to selecting receptive fields requires no additional parameter learning over the previously proposed SOMNet but significantly increases the performance on the MNIST dataset, and also shows promising results on more complex datasets like CIFAR-10, where further research is needed to extract more spatial relationships among the features. In addition, we would also like to explore the possibility of this methodology being used alongside more traditional pooling layers to reduce redundancy and dimensionality in the feature space. Specifically, we would like to experiment with this channel aggregation layer in a supervised CNN to see if it compliments standard spatial pooling. In general, this study gives further evidence that simple unsupervised algorithms such as SOM have a place along side more complex deep models.

References

1. Aly, S.: Learning invariant local image descriptor using convolutional mahalanobis self-organising map. *Neurocomputing* **142**, 239–247 (2014)
2. Bruna, J., Mallat, S.: Invariant scattering convolution networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(8), 1872–1886 (2013)
3. Chan, T.H., Jia, K., Gao, S., Lu, J., Zeng, Z., Ma, Y.: PCANet: a simple deep learning baseline for image classification? *IEEE Trans. Image Process.* **24**(12), 5017–5032 (2015)
4. Cireřan, D., Meier, U., Masci, J., Schmidhuber, J.: A committee of neural networks for traffic sign classification. In: *International Joint Conference on Neural Networks (IJCNN)*, pp. 1918–1921. IEEE (2011)
5. Coates, A., Ng, A., Lee, H.: An analysis of single-layer networks in unsupervised feature learning. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 215–223. JMLR (2011)
6. Coates, A., Ng, A.Y.: The importance of encoding versus training with sparse coding and vector quantization. In: *International Conference on Machine Learning (ICML)*, pp. 921–928. ACM (2011)
7. Coates, A., Ng, A.Y.: Selecting receptive fields in deep networks. In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 2528–2536. Curran Associates, Inc. (2011)
8. Cordel, M.O., Antioquia, A.M.C., Azcarraga, A.P.: Self-organizing maps as feature detectors for supervised neural network pattern recognition. In: Hirose, A., Ozawa, S., Doya, K., Ikeda, K., Lee, M., Liu, D. (eds.) *ICONIP 2016*. LNCS, vol. 9950, pp. 618–625. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46681-1_73
9. Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: *Workshop on Statistical Learning in Computer Vision, European Conference on Computer Vision (ECCV)*, pp. 1–22. Springer, Berlin (2004)
10. Culurciello, E., Jin, J., Dundar, A., Bates, J.: An analysis of the connections between layers of deep neural networks. *arXiv preprint arXiv:1306.0152* (2013)

11. Dong, L., He, L., Kong, G., Zhang, Q., Cao, X., Izquierdo, E.: CUNet: a compact unsupervised network for image classification. arXiv preprint [arXiv:1607.01577](https://arxiv.org/abs/1607.01577) (2016)
12. Dundar, A., Jin, J., Culurciello, E.: Convolutional clustering for unsupervised learning. arXiv preprint [arXiv:1511.06241](https://arxiv.org/abs/1511.06241) (2015)
13. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: a library for large linear classification. *J. Mach. Learn. Res.* **9**(Aug), 1871–1874 (2008)
14. Hankins, R., Peng, Y., Yin, H.: SOMNet: unsupervised feature learning networks for image classification. In: International Joint Conference on Neural Networks (IJCNN), pp. 1221–1228. IEEE (2018)
15. Jarrett, K., Kavukcuoglu, K., LeCun, Y., et al.: What is the best multi-stage architecture for object recognition? In: International Conference on Computer Vision (ICCV), pp. 2146–2153. IEEE (2009)
16. Kannala, J., Rahtu, E.: BSIF: Binarized statistical image features. In: International Conference on Pattern Recognition (ICPR), pp. 1363–1366. IEEE (2012)
17. Kohonen, T.: Self-organized formation of topologically correct feature maps. *Biol. cybern.* **43**(1), 59–69 (1982)
18. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Technical report, Citeseer (2009)
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (NIPS), pp. 1097–1105. Curran Associates, Inc. (2012)
20. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2169–2178. IEEE (2006). <https://doi.org/10.1109/CVPR.2006.68>
21. LeCun, Y., et al.: Handwritten digit recognition with a back-propagation network. In: Advances in Neural Information Processing Systems (NIPS), pp. 396–404. Morgan-Kaufmann (1990)
22. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
23. Lee, H., Grosse, R., Ranganath, R., Ng, A.Y.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: International Conference on Machine Learning (ICML), pp. 609–616. ACM (2009)
24. Lin, M., Chen, Q., Yan, S.: Network in network. arXiv preprint [arXiv:1312.4400](https://arxiv.org/abs/1312.4400) (2013)
25. Lin, T.H., Kung, H.: Stable and efficient representation learning with nonnegativity constraints. In: International Conference on Machine Learning (ICML), pp. 1323–1331. JMLR (2014)
26. Ng, C.J., Teoh, A.B.J.: DCTNet: a simple learning-free approach for face recognition. In: Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), pp. 761–768. IEEE (2015)
27. Ngiam, J., Chen, Z., Chia, D., Koh, P.W., Le, Q.V., Ng, A.Y.: Tiled convolutional neural networks. In: Advances in Neural Information Processing Systems (NIPS), pp. 1279–1287. Curran Associates, Inc. (2010)
28. Peng, Y., Yin, H.: Markov random field based convolutional neural networks for image classification. In: Yin, H., et al. (eds.) IDEAL 2017. LNCS, vol. 10585, pp. 387–396. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68935-7_42
29. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint [arXiv:1409.1556](https://arxiv.org/abs/1409.1556) (2014)

30. Yang, J., Yu, K., Gong, Y., Huang, T.: Linear spatial pyramid matching using sparse coding for image classification. In: Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1794–1801. IEEE (2009)
31. Yin, H.: The self-organizing maps: background, theories, extensions and applications. In: Fulcher, J., Jain, L.C. (eds.) Computational Intelligence: A Compendium, vol. 115, pp. 715–762. Springer, Berlin (2008). https://doi.org/10.1007/978-3-540-78293-3_17
32. Zeiler, M.D., Fergus, R.: Stochastic pooling for regularization of deep convolutional neural networks. arXiv preprint [arXiv:1301.3557](https://arxiv.org/abs/1301.3557) (2013)



Deep Neural Networks with Markov Random Field Models for Image Classification

Yao Peng^(✉), Menyü Liu, and Hujun Yin

School of Electrical and Electronic Engineering, The University of Manchester,
Manchester M13 9PL, UK

{yao.peng,hujun.yin}@manchester.ac.uk,
mengyu.liu-3@postgrad.manchester.ac.uk

Abstract. As one of the most intensively researched topics, image classification has attracted significant attention in recent years. Numerous approaches have been proposed to derive robust and effective image representations and to counter the intra-class variability. Conventional feature extraction and the recent deep neural networks are two common methods to find a good set of features for image description and recognition. Apart from these features-based approach, Markov random fields (MRFs) are generative, probabilistic image texture models, in which global model can be obtained by means of local relations and neighbourhood dependencies. This kind of property shares compatibility with convolutional neural networks (CNNs) and enables combination of CNNs and model-based MRF features. In this work, we propose an MRF loss function in CNNs to minimise modelling errors and estimate parameters. Incorporated with CNNs, these estimated parameters are utilised as the initialised weights in the first convolutional layer. Then the networks are trained. Comprehensive experiments conducted on the MNIST, CIFAR-10 and CIFAR-100 datasets are reported to verify the proposed approach.

Keywords: Image classification · Image representations
Markov random field · Convolutional neural networks

1 Introduction

Image classification is a process of identifying and classifying objects in images or videos through deriving robust image representations and analysing their contextual constraints. Over the last few decades, image classification has received substantial attention in computer vision due to its various potential applications such as surveillance [1], texture analysis [2], remote sensing [3] and medical image analysis [4]. Approaches to image classification have fallen into two major categories: feature-based and model-based. The feature-based methods attempt to find a set of good features which can be used to describe and analyse image properties. Representative methods include conventional feature extraction methods

(e.g. local binary patterns [5], binary gradient patterns [6], histogram of oriented gradient [7] and scale invariant feature transform [8]) and recent deep neural networks (AlexNet [9], GoogleNet [10], ResNet [11] and VGG [12]). In the model-based approaches, however, images are described by a mathematical model, in which a set of parameters are extracted as image features for description and discrimination [13]. Markov random fields (MRFs) and autoregressive models are typical examples.

As one key ingredient of the success of feature-based methods, recently deep learning, in particular convolutional neural networks (CNNs), provides a plausible way of automatically learning hierarchical features with multiple levels of abstraction [14]. Recent advances have witnessed increasing number of applications of deep networks to a wide range of image classification tasks.

Unlike these feature-based approaches, MRFs are generative, flexible and stochastic image texture models, in which contextual dependencies can be established by specifying local conditional probabilities. MRFs take pixel intensity levels in an image as random variables and model image properties which are significant in visual interpretation and image understanding [15–17]. These image models derive global description by specifying local properties of images, which shares compatibility with weight sharing and local connectivity properties in CNNs.

In this paper, we introduce an image classification framework by combining the model-based MRFs and the feature-based CNNs. By minimising the MRF loss function, various models can be derived as the initialisation of the first convolutional layer in CNNs. Normal training process are then employed to learn robust representations for image classification. The effectiveness of the proposed method has been verified on the MNIST [18], CIFAR-10 [19] and CIFAR-100 [19] image datasets.

The remainder of this paper is structured as follows. Section 2 provides a brief review of related work, and Sect. 3 reviews the definition of MRFs, Gaussian MRF models and the parameter estimation approach. The proposed method is demonstrated in Sect. 4. Section 5 presents experimental results, followed by conclusions in Sect. 6.

2 Related Work

2.1 Markov Random Fields

MRFs are region-based models that regard image pixels as random variables and model contextual dependencies among each pixel and its neighbours. Since the Hammersley-Clifford theorem [20,21], MRFs have been widely used in many different computer vision tasks, such as denoising [22], deblurring [23], image segmentation [24] and synthesis [25]. There are many methods for estimating MRF parameters, among which the pseudo-likelihood [20], mean field approximations [26], and coding [20] are based on the maximum a posterior, in addition to the Markov chain Monte Carlo methods [27] which provide a tractable way of sampling and estimation.

Quattoni *et al.* [28] proposed a part-based image recognition method for which the hidden labels of parts in images were considered as hidden variables. The dependencies among the local features of parts, the hidden labels and image labels were modelled by a conditional random field, and belief propagation [29] was used for inference and parameter estimation. Wang and Greg [30] presented a discriminative model for human action recognition which combined local patch features and large-scale features to assign the part labels according to probabilities. The combined method achieved better results than using either of these features alone.

2.2 Convolutional Neural Networks

CNNs were first introduced in 1980s for recognising handwritten digits [31]. In the last few years, with the rapid development in modern computing units, CNNs have been widely used in various fields such as object detection [32], image segmentation [33] and robotics [34]. In the area of image classification, CNNs have been proposed for more complicated tasks [9, 11, 12].

Recently, some researchers explored the properties of early layers in CNNs and replaced these filters with pre-defined kernels. Oyallon *et al.* [35] trained a CNN on top of a scattering network and obtained comparable performance on various datasets with less convolutional layers. Perronnin and Larlus [36] proposed a hybrid network which incorporated fisher vectors with CNNs to produce three preceding layers for feature extraction, encoding and dimensionality reduction. In [37], Sarwar *et al.* replaced parts of the filters in different layers in CNNs with pre-defined Gabor filters to balance the computational efficiency and network performance.

3 Markov Random Field Model and Parameter Estimation

An MRF X defined on the set \mathcal{S} with respect to the neighbourhood system $\mathcal{N} = \{N_s, s \in \mathcal{S}\}$ is a random field that satisfies the following properties:

- Positivity: Any configuration is positive.

$$P(X = x) > 0, \forall x \in \Omega, \quad (1)$$

where x is a possible realisation of the field X and Ω denotes the set of all possible configurations of X .

- Markovianity: The conditional probability of any site given the others only depends upon the configuration of its neighbours.

$$P(X_s = x_s \mid X_r = x_r, r \neq s) = P(X_s = x_s \mid X_r = x_r, r \in N_s), \quad (2)$$

where N_s is the neighbourhood of site s .

- Homogeneity: The conditional probability depends only on its neighbouring sites and is translation invariant.

An MRF is an undirected graph model that explicitly represents random variables and their conditional dependencies. With its conditional probability distribution, it is still difficult to specify an MRF due to several reasons [20]. Fortunately, the Hammersley-Clifford theorem [20,21] provides a mathematically means of specifying MRFs by demonstrating the equivalence between MRFs and Gibbs distribution with regard to the same graph. A random field X is an MRF on \mathcal{S} with respect to a neighbourhood system \mathcal{N} if and only if the corresponding $P(X = x)$ is a Gibbs distribution, which indicates that each MRF can be measured by the Gibbs distribution, and every Gibbs distribution defines an MRF.

As a typical category of MRF models, a Gaussian MRF models is a stationary noncausal two-dimensional autoregressive process that can be expressed by a set of difference equations [38,39] as

$$x_s = \sum_{s+r \in N_s} \beta_r x_{s+r} + e_s, \quad (3)$$

where r is the relative position with respect to central pixel s , and $\{e_s\}$ is a stationary Gaussian noise sequence with zero mean and standard deviation σ^2 characterised by

$$E(e_s e_{s+r}) \begin{cases} \sigma^2 & \text{if } r = (0,0) \\ -\sigma^2 \beta_r & \text{if } r \neq (0,0) \\ 0 & \text{otherwise} \end{cases}, \quad (4)$$

where β_r is the parameter describing relationship between pixels x_s and x_{s+r} . All β_r in the neighbourhood system N_s forms the parameter vector β .

In model-based texture methods, model parameters are used as features for classification. Model parameter estimation plays an significant role in analysing image properties and least square estimation are commonly accepted to estimate Gaussian MRF models [39]. The quadratic difference Θ between the centre pixel and its neighbours can be defined as

$$\Theta = \sum_s \left(x_s - \sum_{s+r \in N_s} \beta_r x_{s+r} \right)^2. \quad (5)$$

which can be implemented and minimised by convolutional operation and gradient descent method in CNNs.

4 Proposed Method

The main advantage of MRF models is that they provide flexible and natural models for defining interaction between spatially related random variables in their neighbourhood systems via clique functions [40]. As a statistical model, MRF models pixels dependently together with their spatial constraints and thus

avoids losing information in feature extraction. In this work we model image texture as MRFs and take the set of parameters as a starting point for CNN training.

MRFs Loss Function: Let $\psi(\mathbf{I})$ denote all the local patches extracted from $\mathbf{I} \in \mathbb{R}^{M \times N \times D}$ - either the original input or activation from previous layer, where M, N denote the width and height of \mathbf{I} and D is the number of channels in the layer where the patch is extracted from. Each patch $\psi_i(\mathbf{I})$ indexed by i is of the size $k \times k \times D$, where k is the width and height of the patch. We define a set of Gaussian MRF models with parameters $\beta_j = [\beta_{j_1}, \beta_{j_2}, \dots, \beta_{j_D}]$ specified on a $k \times k$ neighbourhood system as

$$\beta_{j_h} = (\beta_{j_h}^{11}, \beta_{j_h}^{12}, \dots, \beta_{j_h}^{kk})$$

$$= \begin{bmatrix} \beta_{j_h}^{11} & \beta_{j_h}^{12} & \dots & \beta_{j_h}^{1v} & \dots & \beta_{j_h}^{1k} \\ \beta_{j_h}^{21} & \beta_{j_h}^{21} & \dots & \beta_{j_h}^{2v} & \dots & \beta_{j_h}^{2k} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \beta_{j_h}^{u1} & \beta_{j_h}^{u2} & \dots & \beta_{j_h}^{uv} & \dots & \beta_{j_h}^{uk} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \beta_{j_h}^{k1} & \beta_{j_h}^{k2} & \dots & \beta_{j_h}^{kv} & \dots & \beta_{j_h}^{kk} \end{bmatrix}, \quad (h = 1, 2, \dots, D), \tag{6}$$

where $\beta_{j_h}^{uv} = -1$, $u = v = (k + 1)/2$. Then we take the MRF modelling error as the loss function

$$\begin{aligned} J(\psi_i(\mathbf{I}), \beta_j, \mathbf{b}_j) &= \frac{1}{MNQ} \sum_{i=1}^{MN} \sum_{j=1}^Q \mathbf{a}_{ij} \\ &= \frac{1}{MNQ} \sum_{i=1}^{MN} \sum_{j=1}^Q \|\mathbf{z}_{ij}\| \\ &= \frac{1}{MNQ} \sum_{i=1}^{MN} \sum_{j=1}^Q \|\psi_i(\mathbf{I}) \odot \beta_j + \mathbf{b}_j\|, \end{aligned} \tag{7}$$

where $\mathbf{z}_{ij} = \psi_i(\mathbf{I}) \odot \beta_j + \mathbf{b}_j$, and $\mathbf{a}_{ij} = \|\mathbf{z}_{ij}\|$. The operation \odot denotes the dot product, Q is the total number of MRF models and \mathbf{b}_j denotes the bias of β_j .

Minimisation: To determine these MRF parameters that minimise the loss function, we employ the gradient descent algorithm and the chain rule to calculate partial derivatives with respect to parameters as

$$\frac{\partial}{\partial \beta_j} J(\psi_i(\mathbf{I}), \beta_j, \mathbf{b}_j) = \frac{\partial J}{\partial \mathbf{a}_{ij}} \frac{\partial \mathbf{a}_{ij}}{\partial \mathbf{z}_{ij}} \frac{\partial \mathbf{z}_{ij}}{\partial \beta_j}, \tag{8}$$

$$\frac{\partial}{\partial \mathbf{b}_j} J(\psi_i(\mathbf{I}), \beta_j, \mathbf{b}_j) = \frac{\partial J}{\partial \mathbf{a}_{ij}} \frac{\partial \mathbf{a}_{ij}}{\partial \mathbf{z}_{ij}} \frac{\partial \mathbf{z}_{ij}}{\partial \mathbf{b}_j}, \tag{9}$$

$$\frac{\partial J}{\partial \mathbf{a}_{ij}} = \frac{1}{MNQ}, \quad (10)$$

$$\frac{\partial \mathbf{a}_{ij}}{\partial \mathbf{z}_{ij}} = \begin{cases} 1 & \mathbf{z}_{ij} > 0 \\ -1 & \mathbf{z}_{ij} < 0 \end{cases}, \quad (11)$$

$$\frac{\partial \mathbf{z}_{ij}}{\partial \boldsymbol{\beta}_j} = \boldsymbol{\psi}_i(\mathbf{I}), \quad (12)$$

$$\frac{\partial \mathbf{z}_{ij}}{\partial \mathbf{b}_j} = 1. \quad (13)$$

These trained MRF parameters are used as the initialised weights of the first convolutional layer in CNNs. We then train the network normally using error backpropagation for image classification.

5 Experiments

5.1 Overview

We evaluated our method on three benchmark datasets: MNIST [18], CIFAR-10 [19] and CIFAR-100 [19]. A set of experiments were conducted to demonstrate potential of the proposed method. The networks for the three datasets all consisted of a normal deep convolutional network whose parameters were drawn from a zero centred Gaussian as the baseline, and a set of MRF models whose parameters were estimated by minimising the modelling error. Estimated MRF parameter were considered as the initialisation of the first convolutional layer in CNNs. We implemented the networks using the PyTorch framework [41] with CUDA backends. Training was performed using the stochastic gradient descent with mini-batch size of 128, momentum of 0.9 and weight decay of 0.0005. Furthermore, similar to [42], we employed a “poly” learning rate policy as

$$Lr \left(1 - \frac{Iter}{MaxIter}\right)^{0.9}, \quad (14)$$

where Lr is the initial learning rate, $Iter$ is the current iteration and $MaxIter$ denotes the maximum number of iterations.

5.2 MNIST Experiments

The MNIST handwritten digits dataset contains 60000 training and 10000 test images, each of which is a grayscale image of size 28×28 . For this dataset, a two-layer convolutional network was proposed, and the only pre-processing taken was the mean subtraction.

The baseline network (of architecture: $28 \times 28 \times 1$ -32C5-MP-64C5-MP-128FC-DP-10FC¹) was trained for 60 epochs and started with a learning rate of 0.02 and we obtained a test error of 0.63% as a baseline. To evaluate the proposed method, we first estimated MRF parameters by minimising the MRF loss function. Then these trained MRF filters were applied to the network as initialisation of the first layer filters. The proposed network was trained using the same architecture, parameters and learning strategy as in the baseline. The proposed approach obtained a 0.56% test error, as shown in Table 1, along with other methods. It can be shown that the proposed approach performs better than the baseline, though slightly worse than some state-of-the-art methods. Further statistical test revealed the improvement of the proposed method against the baseline was significant at a level of 1% (p -value = 0.0081).

Table 1. Classification results on the MNIST dataset.

Methods	Error Rate (%)
CNN [43]	0.53
Stochastic Pooling [44]	0.47
Maxout [45]	0.45
DropConnect [46]	0.57
Network in Network [47]	0.47
PCANet [48]	0.62
MRF-CNN [49]	0.38 \pm 0.01
SOMNet [50]	0.65 \pm 0.02
Baseline	0.63 \pm 0.04
Proposed	0.56 \pm 0.02

5.3 CIFAR-10 and CIFAR-100 Experiments

The CIFAR-10 and CIFAR-100 datasets both consist of 50000 training and 10000 test images in 10 and 100 classes respectively, and each RGB image is of size 32×32 . Mean subtraction was applied, and translation and horizontal flippings were used to augment the data.

The baseline and proposed networks all contain five convolutional layers and two fully connected layers (architecture: $32 \times 32 \times 3$ -96C5-MP-(192C3) \times 2-MP-(192C3) \times 2-192FC-DP-10FC). We trained these networks for 200 epochs and

¹ “C”: Convolution, “MP”: Max pooling, “FC”: Fully connected, “DP”: Dropout. The architecture represents a net with input images of size $28 \times 28 \times 1$, two convolutional layers with 32 and 64 feature maps and 5×5 filters respectively, two max pooling layers, a dropout layer and two fully connected output layers with 128 and 10 neurons respectively.

set the initial learning rate to 0.1, and achieved baseline test errors of 9.36% and 34.88% for CIFAR-10 and CIFAR-100 respectively. Then the networks were trained with the MRF initialisation and achieved 8.69% and 33.83% respectively. We also generalised the proposed method to ResNet-20, ResNet-32, and ResNet-110 [11]. Table 2 compares the results of the baseline, proposed methods and other state-of-the-art networks. The statistical significant tests against the baselines showed that the proposed method improved the performance at a significant of 1% (p -value = 0.0076 for CIFAR-10 and p -value = 0.0014 for CIFAR-100). The same trend can also be observed when generalised to ResNet-110 architecture (p -value = 0.0093). The proposed methods achieved better results on these two datasets across different network architectures indicating the usefulness in bringing MRFs in weight updating in the first layer.

Table 2. Classification results on the CIFAR-10 and CIFAR-100 datasets.

Methods	Error rate (%)	
	CIFAR-10	CIFRA-100
Maxout [45]	9.38	-
DropConnect [46]	9.32	-
Network in Network [47]	8.81	-
MRF-CNN [49]	8.75 ± 0.14	33.92 ± 0.17
SOMNet [50]	28.19 ± 0.06	-
Baseline	9.36 ± 0.23	34.88 ± 0.10
Proposed	8.69 ± 0.25	33.83 ± 0.21
ResNet-20 (Baseline) [11]	8.75	-
Proposed + ResNet-20	8.05	-
ResNet-32 (Baseline) [11]	7.51	-
Proposed + ResNet-32	7.27	-
ResNet-110 (Baseline) [11]	6.61 ± 0.16	-
Proposed + ResNet-110	6.27 ± 0.05	-

6 Conclusions

This paper presents an approach to image classification by combining MRF models, to minimise image texture modelling errors, with CNNs, to implicitly learn features with multiple levels of abstraction. We define an MRF loss function and estimate model parameters to efficiently and effectively describe image texture properties. These trained MRF parameters are used as the initialisation of the first convolutional layer in the CNNs, which are then trained normally for image classification.

Results have validated the effectiveness of the proposed approach on the three benchmark datasets. The combination of MRFs and CNNs not only brings the expressive power of deep architectures to probability formulations, but also indicates that convolutional filters can be interpreted as MRF models. Meanwhile, the unsupervised way of MRF parameter estimation makes it possible to generalise across different datasets and applications. Future work will entail applying the approach to more comprehensive datasets, and further incorporating MRFs with CNN training in image segmentation and synthesis.

References

1. Hsieh, J.W., Yu, S.H., Chen, Y.S., Hu, W.F.: Automatic traffic surveillance system for vehicle tracking and classification. *IEEE Trans. Intell. Transp. Syst.* **7**(2), 175–187 (2006)
2. Haralick, R.M., Shanmugam, K., Dinstein, I.: Textural features for image classification. *IEEE Trans. Syst. Man Cybern.* **3**(6), 610–621 (1973)
3. Melgani, F., Bruzzone, L.: Classification of hyperspectral remote sensing images with support vector machines. *IEEE Trans. Geosci. Remote Sens.* **42**(8), 1778–1790 (2004)
4. Li, Q., Cai, W., Wang, X., Zhou, Y., Feng, D.D., Chen, M.: Medical image classification with convolutional neural network. In: *Proceedings of the International Conference on Control Automation Robotics & Vision (ICARCV)* (2014) 844–848
5. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(7), 971–987 (2002)
6. Huang, W., Yin, H.: Robust face recognition with structural binary gradient patterns. *Pattern Recognit.* **68**, 126–140 (2017)
7. Freeman, W.T., Roth, M.: Orientation histograms for hand gesture recognition. In: *Proc. IEEE International Conference on Automatic Face and Gesture Recognition (FG)*, vol. 12, pp. 296–301 (1995)
8. Lowe, D.G.: Object recognition from local scale-invariant features. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, vol. 2, pp. 1150–1157 (1999)
9. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: *Proceedings of the Conference on Advance Neural Information Processing Systems (NIPS)*, pp. 1097–1105 (2012)
10. Szegedy, C., et al.: Going deeper with convolutions. In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9 (2015)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778 (2016)
12. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014)
13. Kashyap, R.L., Khotanzad, A.: A model-based method for rotation invariant texture classification. *IEEE Trans. Pattern Anal. Mach. Intell.* **4**, 472–481 (1986)
14. LeCun, Y., Bengio, Y., Hinton, G.E.: Deep learning. *Nature* **521**(7553), 436–444 (2015)

15. Julesz, B.: Visual pattern discrimination. *IRE Trans. Inf. Theory* **8**(2), 84–92 (1962)
16. Julesz, B.: Textons, the elements of texture perception, and their interactions. *Nature* **290**(5802), 91–97 (1981)
17. Li, S.Z.: A Markov random field model for object matching under contextual constraints. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 866–866 (1994)
18. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
19. Krizhevsky, A., Hinton, G.E.: Learning multiple layers of features from tiny images (2009)
20. Besag, J.: Spatial interaction and the statistical analysis of lattice systems. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **36**(2), 192–236 (1974)
21. Hammersley, J.M., Clifford, P.E.: Markov random fields on finite graphs and lattices. Unpublished manuscript (1971)
22. Zoran, D., Weiss, Y.: From learning models of natural image patches to whole image restoration. In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 479–486 (2011)
23. Schmidt, U., Roth, S.: Shrinkage fields for effective image restoration. In: *Proceedings IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2774–2781 (2014)
24. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected CRFs with Gaussian edge potentials. In: *Proceedings of the Conference on Advances in Neural Information Processing Systems (NIPS)*, pp. 109–117 (2011)
25. Cross, G.R., Jain, A.K.: Markov random field texture models. *IEEE Trans. Pattern Anal. Mach. Intell.* **1**, 25–39 (1983)
26. Parisi, G.: *Statistical Field Theory*. Addison-Wesley, Boston (1988)
27. Smith, A.F., Roberts, G.O.: Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods. *J. R. Stat. Soc. Series B Stat. Methodol.* **55**(1), 3–23 (1993)
28. Quattoni, A., Collins, M., Darrell, T.: Conditional random fields for object recognition. In: *Proceedings of the Conference on Advances in Neural Information Processing Systems (NIPS)*, pp. 1097–1104 (2005)
29. Pearl, J.: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Elsevier, Amsterdam (2014)
30. Wang, Y., Mori, G.: Learning a discriminative hidden part model for human action recognition. In: *Proceedings of the Conference on Advances in Neural Information Processing Systems (NIPS)*, pp. 1721–1728 (2009)
31. LeCun, Y., et al.: Backpropagation applied to handwritten zip code recognition. *Neural Comput.* **1**(4), 541–551 (1989)
32. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 580–587 (2014)
33. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440 (2015)
34. Levine, S., Finn, C., Darrell, T., Abbeel, P.: End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.* **17**(1), 1334–1373 (2016)
35. Oyallon, E., Belilovsky, E., Zagoruyko, S.: Scaling the scattering transform: deep hybrid networks. In: *Proceedings IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (2017)

36. Perronnin, F., Larlus, D.: Fisher vectors meet neural networks: a hybrid classification architecture. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3743–3752 (2015)
37. Sarwar, S.S., Panda, P., Roy, K.: Gabor filter assisted energy efficient fast learning convolutional neural networks. In: Proceedings of the IEEE International Symposium on Low Power Electronics and Design (ISLPED), pp. 1–6 (2017)
38. Geman, S., Geman, D.: Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.* **6**, 721–741 (1984)
39. Kashyap, R., Chellappa, R.: Estimation and choice of neighbors in spatial-interaction models of images. *IEEE Trans. Inf. Theory* **29**(1), 60–72 (1983)
40. Dass, S.C.: Markov random field models for directional field and singularity extraction in fingerprint images. *IEEE Trans. Image Process.* **13**(10), 1358–1367 (2004)
41. Paszke, A., et al.: Automatic differentiation in PyTorch. In: Proceedings of the Conference on Advances in Neural Information Processing Systems (NIPS) (2017)
42. Liu, W., Rabinovich, A., Berg, A.C.: ParseNet: looking wider to see better. arXiv preprint [arXiv:1506.04579](https://arxiv.org/abs/1506.04579) (2015)
43. Jarrett, K., Kavukcuoglu, K., LeCun, Y., et al.: What is the best multi-stage architecture for object recognition? In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 2146–2153 (2009)
44. Zeiler, M.D., Fergus, R.: Stochastic pooling for regularization of deep convolutional neural networks. arXiv preprint [arXiv:1301.3557](https://arxiv.org/abs/1301.3557) (2013)
45. Goodfellow, I.J., Warde-Farley, D., Mirza, M., Courville, A., Bengio, Y.: Maxout networks. arXiv preprint [arXiv:1302.4389](https://arxiv.org/abs/1302.4389) (2013)
46. Wan, L., Zeiler, M., Zhang, S., Cun, Y.L., Fergus, R.: Regularization of neural networks using dropconnect. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 1058–1066 (2013)
47. Lin, M., Chen, Q., Yan, S.: Network in network. arXiv preprint [arXiv:1312.4400](https://arxiv.org/abs/1312.4400) (2013)
48. Chan, T.H., Jia, K., Gao, S., Lu, J., Zeng, Z., Ma, Y.: PCANet: a simple deep learning baseline for image classification? *IEEE Trans. Image Process.* **24**(12), 5017–5032 (2015)
49. Peng, Y., Yin, H.: Markov random field based convolutional neural networks for image classification. In: Proceedings of the International Conference on Intelligent Data Engineering and Automated Learning (IDEAL), pp. 387–396 (2017)
50. Hankins, R., Peng, Y., Yin, H.: SOMNet: unsupervised feature learning networks for image classification. In: Proceedings of the International Joint Conference on Neural Networks, pp. 1221–1228 (2017)

Author Index

- AbuRas, Radu I-821
Adla, Abdelkader II-271
Agafonov, Anton I-253
Agudelo-Castañeda, Dayana II-210
Aguiar, Guilherme I-773
Ahmadi, Roja I-556
Albarca-Molina, Alejandro I-277
Alcañiz, Mariano I-35
Aledo, Juan A. I-783
Aler, R. II-155
Alexandre, Luís A. I-711
Almatarneh, Sattam I-215
Almeida, José João I-374
Arroba, Patricia I-491
Atzmueller, Martin II-94
- Baghoussi, Yassine I-753
Baldominos, Alejandro I-364
Banković, Zorana II-241
Barbosa, Paulo I-623
Barhamgi, Mahmoud II-3
Basante, C. I-661
Becerra, M. A. I-410
Béjar, Javier I-89
Beko, Marko II-262
Bello-Orgaz, Gema II-316
Benslimane, Djamel II-3
Bertrand, Olivier J. N. I-508
Bielza, Concha I-117, I-354
Biswal, Pradyut Kumar I-269
Blachnik, Marcin I-631
Blanco Valencia, X. P. I-410
Blanco, Roberto I-531
Blanco-Valencia, X. P. I-661
Blundo, Carlo I-204
Bobek, Szymon II-64, II-75
Bobowska, Barbara I-184
Bootkrajang, Jakramate I-69
Borrison, Reuben I-611
Bovo, Riccardo II-84
Briongos, Samira I-531
Brisson, Laurent I-233
Brzywczy, Edyta II-54, II-75
Bu, Seok-Jun I-468
- Calikus, Ece II-41
Calise, Anthony I-328
Camacho, David II-3, II-307, II-316, II-325
Cardoso, Henrique Lopes I-730
Carnero, Mercedes II-316
Carrascosa, Carlos I-520
Casanova-Mateo, C. II-163
Castellanos-Domínguez, G. I-661
Catalina, Alejandro II-147
Chaijaruwanich, Jeerayut I-69
Charte, Francisco I-439
Chioua, Moncef I-611
Cho, Sung-Bae I-457, I-468, I-481, I-499
Cilla, Juan J. I-531
Cimato, Stelvio I-204
Ciurez, Marius Andrei I-99
Cobo, M. J. II-130, II-138
Cobo, Manuel Jesús II-121
Cobos, Carlos I-801
Colomer, Adrián I-164, I-174, I-224, I-642
Córdoba, Irene I-117
Cornejo-Bueno, L. II-163
Correia, F. I-548
Cortés, Ulises I-89
Cortés-Toro, Enrique I-125
Costa, Angelo I-520
Crawford, Broderick I-125
Cruz, José Aleixo I-730
Cruz-Cruz, L. I-661
Czepielik, Łukasz I-631
- da Rocha Neto, Ajalmar R. I-670
da Silva Costa, Fernando Henrique I-317
Dash, Bodhisattva I-1
de Carvalho, André C. P. L. F. I-603, I-623
de la Pompa, Víctor II-147
de Luis Reboredo, Ana I-722
del Jesus Díaz, María José I-448
del Jesus, María J. I-439
Del Ser, Javier I-79, II-201, II-325
Delgado-Trejos, Edilson I-410
Dias, Madson L. D. I-670
Diaz, Alexandra Katuska Ramos I-317
Diaz-Pinto, Andres I-224

- Diego-Mas, Jose Antonio I-35
 Dix, Marcel I-611
 Dobrenko, Natalia I-195
 Doderó, Juan Manuel II-105
 Dora, Chinmayee I-269
 Dorado-Moreno, M. II-180
 Dorronsoro, José R. II-147
 Drotár, Peter I-344
 Duarte, Abraham II-336
 Duarte-Martínez, V. II-130
 Duque, Diogo I-730
 Duque-Mejía, C. I-410
 Durán-Domínguez, Arturo I-125
 Durán-Rosal, Antonio M. II-171
- Egelhaaf, Martin I-508
 EL-Manzalawy, Yasser I-740
 Elyan, Eyad I-689
- Fan, Yuantao II-41
 Faroughian, F. II-21
 Fazzolari, Michela I-698
 Fedulov, Vadim I-27
 Feld, Sebastian I-156
 Fister Jr., Iztok I-79, II-201, II-325
 Fister, Iztok I-79, II-201, II-325
 Frangi, Alejandro F. I-224
 Fuentes-Hurtado, Félix I-27, I-35
- Galván, I. M. II-155
 Gálvez, Akemi I-79, II-201, II-325
 Gamallo, Pablo I-215
 Gamboa-Rosales, H. II-113
 Gamboa-Rosales, N. K. II-113
 Gamez, Jose A. I-783
 García, José Gabriel I-642
 Garcia, Kemilly Dearo I-603, I-623
 García-Rodríguez, Jose I-245, I-297, II-288
 García-Sánchez, P. II-138
 Garlatti, Serge I-233
 Garrido-Merchán, Eduardo C. I-277
 Gazda, Vladimír I-344
 Gil González, Ana B. I-722
 Gil-Begue, Santiago I-354
 Gnip, Peter I-344
 Gomes, João P. P. I-670
 Gómez-Orellana, Antonio M. II-171
 Gómez-Pulido, Juan A. I-125
 Gonçalves, Teresa I-69
- Gonzalez-Pardo, Antonio II-307
 González-Toro, Ángel Rafael II-105
 Gorohov, Oleg II-221
 Grigull, Lorenz I-419
 Grzenda, Maciej II-233
 Guijo-Rubio, David II-171
 Gusarova, Natalia I-195
 Gutiérrez, P. A. II-180
 Gutiérrez, Pedro A. II-171
 Guziur, Jakub I-586
- Hamam, Y. I-9
 Hammer, Barbara I-508
 Hankins, Richard I-838
 Hasani, Zirije I-385
 Hashimoto, Takako II-280
 Hernández, José II-316
 Hernandez, Martha Mendoza II-210
 Herrera, Francisco I-801
 Herrera-Viedma, Enrique I-801, II-121
 Hervás-Martínez, C. II-180
 Hervás-Martínez, César II-171
 Honavar, Vasant I-740
 Hsieh, Tsung-Yu I-740
 Huertas-Tato, J. II-155
- Iglesias, Andrés I-79, II-201, II-325
 Inkeaw, Papangkorn I-69
 Ionascu, Costel I-261
 Isasi, Pedro I-364
 Islam, Manjurul I-147
 Itabashi, Taiki I-17
 Itani, Alya I-233
- Jakimovski, Boro I-385
 Jankowski, Dariusz I-184
 Jayne, Chrisina I-689
 Jebari, Chaker II-121
 Ji, Li I-679
 Jia, Caiyan I-134
 Jing, Yanguo II-31
 Jonas, Konstantin I-285
 Jordaan, J. A. I-9
 Jovanovic, Raka II-262
 Julian, Vicente I-520
 Julier, Simon II-84
 Jung, Jason J. I-792
 Jurek, Kamil II-64

- Kammoun, Wafa I-46
 Kaothanthong, Natsuda I-539
 Kazachuk, Maria II-221
 Kaznacheeva, Anna I-195
 Kelleher, John D. I-107
 Khan, Gul Muhammad I-594
 Khan, Nadia Masood I-594
 Kiermeier, Marie I-156
 Kim, Jin-Young I-499
 Kim, Jong-Myon I-147
 Kim, Tae-Young I-481
 Klawonn, Frank I-285, I-419
 Klemp, Kristian I-27
 Kloeppe, Benjamin II-94
 Klöpfer, Benjamin I-611
 Koehler, Wolfgang II-31
 Koga, Hisashi I-17
 Kon-Popovska, Margita I-385
 Kordos, Mirosław I-631
 Kortum, Xiaowei I-419
 Kostakos, Panos II-21
 Kosunen, Ilkka I-821
 Ksieniewicz, Paweł I-761, II-296
- Lafuente, Javier Teira I-722
 Laimek, Roongtawan I-539
 Lara-Cabrera, Raúl II-3
 Larrañaga, Pedro I-117, I-354
 Larsen, Michael I-27
 Lechner, Werner I-419
 Lin, Guangyan I-679
 Linnhoff-Popien, Claudia I-156
 Liu, Menyü I-849
 Lobantsev, Artem I-195
 Lóderer, Marek I-651
 Logofătu, Doina I-429
 López-Herrera, A. G. II-130
 López-Robles, J. R. II-113
- Macredie, Robert D. I-556
 Maddar, Hela I-46
 Maia, Átilla N. I-670
 Malagón, Pedro I-531
 Manero, Jaume I-89
 Mannhardt, Felix II-84
 Marcondes, Francisco S. I-374
 Martín-Vázquez, R. II-155
 Mashechkin, Igor II-221
 Mendes, Gonçalo Sousa II-250
- Mendes-Moreira, João I-603, I-623, I-753
 Mendoza, Heidy Posso II-210
 Mendoza, Martha I-801
 Meyer, Stefan I-508
 Michalak, Krzysztof I-58, I-305
 Mihaescu, Marian Cristian I-99, I-261,
 I-821, I-829
 Mohanty, Figlu I-1
 Moloi, K. I-9
 Morales, Sandra I-27, I-224
 Morgado Gamero, W. B. II-210
 Mossi, Jose M. I-27
 Mota, José Miguel II-105
 Moya, José M. I-491, I-531
 Muecke, Urs I-419
 Muñoz-Ordóñez, Julián I-801
- Nalepa, Grzegorz J. II-75
 Naranjo, Rodrigo I-812
 Naranjo, Valery I-27, I-35, I-164, I-174,
 I-224, I-642
 Nascimento, Susana II-250
 Nguyen, Hoang Long I-792
 Nguyen, Minh I-429
 Noering, Fabian Kai-Dietrich I-285
 Novais, Paulo I-374, I-520
 Nowaczyk, Sławomir II-41
- Oğul, Hasan I-576
 Ojeda, Francisco Chartre I-448
 Oliveira, Eugénio I-730
 Oliveira, Manuel Fradinho II-84
 Orsenigo, Carlotta I-567
 Ortega, C. I-661
 Osaba, Eneko I-79, II-201, II-325
 Otegi-Olaso, J. R. II-113
 Oussalah, Mourad II-21
- Palesi, Ileana I-328
 Panizo, Ángel II-9, II-15, II-316
 Parfenov, Vladimir I-195
 Parody, Alexander II-210
 Pavlík, Peter I-651
 Pawlak, Michał I-586
 Peluffo-Ordóñez, D. H. I-410, I-661
 Peña, D. F. I-661
 Peñaranda, Francisco I-642
 Peng, Yao I-838, I-849
 Pereira, Joana I-174

- Peres, Sarajane Marques I-317
 Pérez, Jaime I-491
 Pérez, Sergio I-491
 Pérez-Peló, Sergio II-336
 Person, Tatiana II-105
 Pertierra, Álvaro Pardo I-722
 Petrocchi, Marinella I-698
 Petrovic, Ivan I-328
 Petrovic, Smiljana I-328
 Petrovski, Andrei I-689
 Petrovskiy, Mikhail II-221
 Phan, Thomy I-156
 Pineda, Gibran Fuentes I-17
 Pongpech, Worapol Alex I-336
 Poniszewska-Maraña, Aneta I-586
 Popescu, Paul Stefan I-829
 Popescu, Paul-Stefan I-261
 Porto Gomez, I. II-113
 Prieto, L. II-180
 Prosvirin, Alexander I-147
 Pulgar, Francisco J. I-439
- Raja, Muhammad Adil II-191
 Ramirez, Margarita Castillo II-210
 Revelo-Fuelagán, E. J. I-410
 Revelo-Fuelagán, J. I-661
 Rincon, Jaime A. I-520
 Rivas, Antonio Jesús Rivera I-448
 Rivera, Antonio J. I-439
 Rivolli, A. I-548
 Robles-Berumen, H. II-113
 Rocha, P. I-548
 Rodrigo, Enrique G. I-783
 Ross, Robert J. I-107
 Rozinajová, Viera I-651
 Ruiz-Chavez, Zoila I-245, I-297, II-288
 Ruiz-Rube, Iván II-105
 Rup, Suvendu I-1
 Ryan, Conor II-191
- Saez, Yago I-364
 Salazar-Castro, J. A. I-661
 Salcedo-Sanz, S. II-163, II-180
 Sales, M. Á. I-642
 Salvador-Meneses, Jaime I-245, I-297,
 II-288
- Sánchez, Mabel II-316
 Sánchez-Oro, Jesús II-336
 Sant'Anna, Anita II-41
 Santos, Matilde I-812
 Sanz-Justo, J. II-163
 Serna-Guarín, L. I-410
 Shalyto, Anatoly I-195
 Shin, Wonsup I-457
 Silva, Andrei Martins I-317
 Silva, Cristiana I-164
 Silva, P. I-548
 Simić, Dragan II-241
 Simić, Svetislav D. II-241
 Simić, Svetlana II-241
 Siniscalchi, Luisa I-204
 Soares, C. I-548
 Soto, Ricardo I-125
 Spognardi, Angelo I-698
 Sprick, Barbara I-611
 Stankevich, Andrei I-195
 Suárez, Patricia II-201
 Sun, Yiwei I-740
 Supnithi, Thepchai I-539
 Suzuki, Satoshi I-17
- Tabik, Siham I-801
 Tallón-Ballesteros, Antonio J. II-262, II-280,
 II-288
 Tan, Huobin I-679
 Teodorescu, Oana Maria I-829
 Thorburn, Joshua II-9
 Toda, Takahisa I-17
 Torm, Marie I-27
 Torregrosa, Javier II-9, II-15
 Trzcionkowska, Agnieszka II-54
 Tuba, Eva II-262
 Tuba, Milan II-262, II-280
 Tucker, Allan I-556
 Turcu, Gabriel I-821
- Varando, Gherardo I-117
 Vatian, Aleksandra I-195
 Velinov, Goran I-385
 Vercellis, Carlo I-567
 Viedma, Daniel Trujillo I-448
 Vilain, Patricia I-773
 Viloría, Amelec II-210

Volpetti, Claudia [I-567](#)
Vuttipittayamongkol, Pattaramon [I-689](#)

Wang, Fei [I-107](#)
Wang, Jie [I-398](#)
Woldbye, David [I-27](#)
Woźniak, Michał [II-296](#)

Xu, Yanwu [I-224](#)
Xue, Bing [II-280](#)

Yin, Hujun [I-838, I-849](#)
Youssef, Habib [I-46](#)
Yumaganov, Alexander [I-253](#)

Zacarias, Abel [I-711](#)
Zapata-Hernández, C. [I-410](#)
Zhang, Hao [I-398](#)
Zhu, Sinan [I-134](#)
Zoričák, Martin [I-344](#)
Zouggar, Souad Taleb [II-271](#)