

Denis Helic · Gerhard Leitner ·
Martin Stettinger · Alexander Felfernig ·
Zbigniew W. Raś (Eds.)

LNAI 12117

Foundations of Intelligent Systems

25th International Symposium, ISMIS 2020
Graz, Austria, September 23–25, 2020
Proceedings

 Springer

Lecture Notes in Artificial Intelligence

12117

Subseries of Lecture Notes in Computer Science

Series Editors

Randy Goebel

University of Alberta, Edmonton, Canada

Yuzuru Tanaka

Hokkaido University, Sapporo, Japan

Wolfgang Wahlster

DFKI and Saarland University, Saarbrücken, Germany

Founding Editor

Jörg Siekmann

DFKI and Saarland University, Saarbrücken, Germany

More information about this series at <http://www.springer.com/series/1244>

Denis Helic · Gerhard Leitner ·
Martin Stettinger · Alexander Felfernig ·
Zbigniew W. Raś (Eds.)

Foundations of Intelligent Systems

25th International Symposium, ISMIS 2020
Graz, Austria, September 23–25, 2020
Proceedings

Editors

Denis Helic
Graz University of Technology
Graz, Austria

Martin Stettinger
Graz University of Technology
Graz, Austria

Zbigniew W. Raś
University of North Carolina at Charlotte
Charlotte, NC, USA

Gerhard Leitner
University of Klagenfurt
Klagenfurt, Austria

Alexander Felfernig
Graz University of Technology
Graz, Austria

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Artificial Intelligence
ISBN 978-3-030-59490-9 ISBN 978-3-030-59491-6 (eBook)
<https://doi.org/10.1007/978-3-030-59491-6>

LNCS Sublibrary: SL7 – Artificial Intelligence

© Springer Nature Switzerland AG 2020

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, expressed 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 volume contains the papers selected for presentation at the 25th International Symposium on Methodologies for Intelligent Systems (ISMIS 2020), which was held in Graz, Austria, 2020. The symposium was organized by the Department of Software Technology at the Technical University of Graz, Austria. ISMIS is a conference series that started in 1986. Held twice every three years, it provides an international forum for exchanging scientific, research, and technological achievements in building intelligent systems. In particular, major areas selected for ISMIS 2020 included: Explainable AI (XAI), machine learning, deep learning, data mining, recommender systems, constraint based systems, autonomous systems, applications (configuration, Internet of Things, financial services, e-health, etc.), intelligent user interfaces, user modeling, human computation, socially-aware systems, digital libraries, intelligent agents, information retrieval, natural language processing, knowledge integration and visualization, knowledge representation, soft computing, and web and text mining. This year, the following sessions were organized: special sessions for invited talks, for best paper nominees and best student paper nominees, thematic sessions on natural language processing, deep learning and embeddings, digital signal processing, modeling and reasoning, machine learning applications, and finally a section containing short paper presentations on diverse related topics.

We received 79 submissions that were carefully reviewed by three or more Program Committee members or external reviewers. After a rigorous reviewing process, 35 regular papers, 8 short papers, and 3 invited papers were accepted for presentation at the conference and publication in the ISMIS 2020 proceedings volume. An additional selection of papers was assigned to a specific industrial track. It is truly a pleasure to thank all the people who helped this volume come into being and made ISMIS 2020 a successful and exciting event. In particular, we would like to express our appreciation for the work of the ISMIS 2020 Program Committee members and external reviewers who helped assure the high standard of accepted papers. We would like to thank all authors of ISMIS 2020, without whose high-quality contributions it would not have been possible to organize the conference. We are grateful to all the organizers of and contributors to a successful preparation and implementation of ISMIS 2020.

The invited talks for ISMIS 2020 were: “Complementing Behavioural Modeling with Cognitive Modeling for Better Recommendations,” given by Marko Tkalcić, University of Primorska in Koper, Slovenia; “Fairness is not a number: Methodological implications of the politics of fairness-aware systems,” given by Robin Burke, University of Colorado Boulder, USA; and “Brevity*,” given by Robert West, EPFL Lausanne, Switzerland. We wish to express our thanks to all the invited speakers for accepting our invitation to give plenary talks. We are thankful to the people at Springer (Alfred Hofmann, Anna Kramer, and Aliaksandr Birukou) for supporting the ISMIS

2020. We believe that the proceedings of ISMIS 2020 will become a valuable source of reference for your ongoing and future research activities.

July 2020

Denis Helic
Gerhard Leitner
Martin Stettinger
Alexander Felfernig

Aijun An	University of York, UK
Troels Andreasen	Roskilde University, Denmark
Annalisa Appice	Università degli Studi di Bari, Italy
Martin Atzmueller	Tilburg University, The Netherlands
Arunkumar Bagavathi	Oklahoma State University, USA
Ladjel Bellatreche	University of Poitiers, France
Robert Bembenik	Warsaw University of Technology, Poland
Petr Berka	University of Economics, Prague, Czech Republic
Maria Bielikova	Slovak University of Technology in Bratislava, Slovakia
Gloria Bordogna	CNR, Italy
Jose Borges	University of Porto, Portugal
François Bry	Ludwig Maximilian University of Munich, Germany
Jerzy Błaszczyński	Poznań University of Technology, Poland
Michelangelo Ceci	Università degli Studi di Bari, Italy
Jianhua Chen	Louisiana State University, USA
Silvia Chiusano	Politecnico di Torino, Italy
Roberto Corizzo	Università degli Studi di Bari, Italy
Alfredo Cuzzocrea	ICAR-CNR, University of Calabria, Italy
Marcilio De Souto	LIFO, University of Orleans, France
Luigi Di Caro	University of Torino, Italy
Stephan Doerfel	Micromata, Germany
Peter Dolog	Aalborg University, Denmark
Dejing Dou	University of Oregon, USA
Saso Dzeroski	Jožef Stefan Institute, Slovenia
Christoph F. Eick	University of Houston, USA
Tapio Elomaa	Tampere University of Technology, Finland
Andreas Falkner	Siemens AG, Austria
Nicola Fanizzi	Università degli studi di Bari “Aldo Moro”, Italy
Stefano Ferilli	Università degli Studi di Bari, Italy
Gerhard Friedrich	University of Klagenfurt, Austria
Naoki Fukuta	Shizuoka University, Japan
Maria Ganzha	Warsaw University of Technology, Poland
Paolo Garza	Politecnico di Torino, Italy
Martin Gebser	University of Klagenfurt, Austria
Bernhard Geiger	Know-Center GmbH, Austria
Michael Granitzer	University of Passau, Germany
Jacek Grekow	Bialystok University of Technology, Poland
Mohand-Said Hacid	Université Claude Bernard Lyon 1, France
Hakim Hacid	Zayed University, UAE
Allel Hadjali	LIAS, ENSMA, France
Mirsad Hadzikadic	UNC Charlotte, USA
Ayman Hajja	College of Charleston, USA
Alois Haselboeck	Siemens AG, Austria
Shoji Hirano	Shimane University, Japan
Jaakko Hollmén	Aalto University, Finland

Andreas Holzinger	Medical University and Graz University of Technology, Austria
Andreas Hotho	University of Würzburg, Germany
Lothar Hotz	University of Hamburg, Germany
Dietmar Jannach	University of Klagenfurt, Austria
Adam Jatowt	Kyoto University, Japan
Roman Kern	Know-Center GmbH, Austria
Matthias Klusch	DFKI, Germany
Dragi Kocev	Jožef Stefan Institute, Slovenia
Roxane Koitz	Graz University of Technology, Austria
Bozena Kostek	Gdańsk University of Technology, Poland
Mieczysław Kłopotek	Polish Academy of Sciences, Poland
Dominique Laurent	Université de Cergy-Pontoise, France
Marie-Jeanne Lesot	LIP6, UPMC, France
Rory Lewis	University of Colorado at Colorado Springs, USA
Elisabeth Lex	Graz University of Technology, Austria
Antoni Ligeza	AGH University of Science and Technology, Poland
Yang Liu	Hong Kong Baptist University, Hong Kong
Jiming Liu	Hong Kong Baptist University, Hong Kong
Corrado Loglisci	Università degli Studi di Bari, Italy
Henrique Lopes Cardoso	University of Porto, Portugal
Donato Malerba	Università degli Studi di Bari, Italy
Giuseppe Manco	ICAR-CNR, Italy
Yannis Manolopoulos	Open University of Cyprus, Cyprus
Małgorzata Marciniak	Polish Academy of Science, Poland
Mamoun Mardini	University of Florida, USA
Elio Masciari	University of Naples, Italy
Paola Mello	University of Bologna, Italy
João Mendes-Moreira	University of Porto, Portugal
Luis Moreira-Matias	NEC Laboratories Europe, Germany
Mikolaj Morzy	Poznań University of Technology, Poland
Agnieszka Mykowiecka	IPI PAN, Poland
Tommi Männistö	University of Helsinki, Finland
Mirco Nanni	ISTI-CNR, Italy
Amedeo Napoli	LORIA, CNRS, Inria, Université de Lorraine, France
Pance Panov	Jožef Stefan Institute, Slovenia
Jan Paralic	Technical University Kosice, Slovakia
Ruggero G. Pensa	University of Torino, Italy
Jean-Marc Petit	Université de Lyon, INSA Lyon, France
Ingo Pill	Graz University of Technology, Austria
Luca Piovesan	DISIT, Università del Piemonte Orientale, Italy
Olivier Pivert	IRISA-ENSSAT, France
Lubos Popelinsky	Masaryk University, Czech Republic
Jan Rauch	University of Economics, Prague, Czech Republic
Marek Reformat	University of Alberta, Canada
Henryk Rybiński	Warsaw University of Technology, Poland

Hiroshi Sakai	Kyushu Institute of Technology, Japan
Tiago Santos	Graz University of Technology, Austria
Christoph Schommer	University of Luxembourg, Luxembourg
Marian Scuturici	LIRIS, INSA Lyon, France
Nazha Selmaoui-Folcher	University of New Caledonia, France
Giovanni Semeraro	Università degli Studi di Bari, Italy
Samira Shaikh	UNC Charlotte, USA
Dominik Slezak	University of Warsaw, Poland
Urszula Stanczyk	Silesian University of Technology, Poland
Jerzy Stefanowski	Poznań University of Technology, Poland
Marcin Sydow	PJIT and ICS PAS, Poland
Katarzyna Tarnowska	San Jose State University, USA
Herna Viktor	University of Ottawa, Canada
Simon Walk	Graz University of Technology, Austria
Alicja Wieczorkowska	Polish-Japanese Academy of Information Technology, Poland
David Wilson	UNC Charlotte, USA
Yiyu Yao	University of Regina, Canada
Jure Zabkar	University of Ljubljana, Slovenia
Slawomir Zadrozny	Polish Academy of Sciences, Poland
Wlodek Zadrozny	UNC Charlotte, USA
Bernard Zenko	Jožef Stefan Institute, Slovenia
Beata Zielosko	University of Silesia
Arkaitz Zubiaga	Queen Mary University of London, UK

Additional Reviewers

Max Toller	Michelangelo Ceci
Henryk Rybiński	Giovanni Semeraro
Allel Hadjali	Michael Granitzer
Giuseppe Manco	Simon Walk
Aijun An	

Publicity Chair

Müslüm Atas	Graz University of Technology, Austria
-------------	--

Publication Chair

Trang Tran	Graz University of Technology, Austria
------------	--

Web and Local Committee

Jörg Baumann
Elisabeth Orthofer
Petra Schindler

Graz University of Technology, Austria
Graz University of Technology, Austria
Graz University of Technology, Austria

Invited Talks

Complementing Behavioural Modeling with Cognitive Modeling for Better Recommendations

Marko Tkalčič 

University of Primorska, Faculty of Mathematics, Natural Sciences and Information Technologies, Glagoljaška 8, SI-6000 Koper, Slovenia
marko.tkalcic@famnit.upr.si

Abstract. Recommender systems are systems that help users in decision-making situations where there is an abundance of choices. We can find them in our everyday lives, for example in online shops. State-of-the-art research in recommender systems has shown the benefits of behavioural modeling. Behavioural modeling means that we use past ratings, purchases, clicks etc. to model the user preferences. However, behavioural modeling is not able to capture certain aspects of the user preferences. In this talk I will show how the usage of complementary research in cognitive models, such as personality and emotions, can benefit recommender systems.

Keywords: Recommender systems · Behavioural modeling · Cognitive modeling

Fairness Is Not a Number: Methodological Implications of the Politics of Fairness-Aware Systems

Robin Burke 

Department of Information Science, University of Colorado, Boulder
robin.burke@colorado.edu

Abstract. This invited talk will explore the methodological challenges for intelligent systems development posed by questions of algorithmic fairness. In particular, we will discuss the ambiguity of fairness as a concept and the gap between simple formalizations of fairness questions and the challenges of real applications, where there are a multiplicity of stakeholders who may perceive fairness in different ways.

Keywords: Fairness · Machine learning

1 Fairness in Machine Learning and Recommendation

The problem of bias and fairness in algorithmic systems generally and in machine learning systems in particular is a critical issue for our increasingly data-centric world. However, the emergence of this topic has thrust computer scientists into unfamiliar territory, especially with regard to system development. A substantial body of research on fairness in machine learning, especially in classification settings, has emerged in the past ten years, including formalizing definitions of fairness [1, 5, 7, 10] and offering algorithmic techniques to mitigate unfairness ~ [8, 11–13]. However, as noted in [4], there is little published research that investigates the complexity of real-world practices around fairness and reports findings from practical implementations. As a consequence, results from fairness-aware machine learning research often lack relevance for developers of real systems, and methodologies are lacking.

Fairness may be thought of as a property of particular system outcomes, like other system characteristics, such as reliability, and like these properties has to be designed into systems, accounted for in testing, and monitored in deployment. Fairness is also a product of a complex set of social norms and interactions. Fair outcomes will be achieved and sustained throughout a system’s lifecycle, if and only if, the process or methodology by which the system is developed and maintained itself incorporates the

multiple viewpoints of different stakeholders and provides a mechanism for arbitrating between them.

Economists recognize four rubrics under which fairness can be defined [9]:

1. Fairness as exogenous right: The definition of and necessity for fairness is imposed from outside, by legal requirement, for example.
2. Fairness as reward: Fairness is served by providing extra benefits from a system for those who contribute more to it. For example, a salary bonus in recognition of excellent work.
3. Fairness as compensation: Providing a benefit to those otherwise disadvantaged. A handicap in golf, for example, helps players of different abilities compete on a level basis.
4. Fairness as fitness. A subtle efficiency-based condition that states that a fair distribution is one in which resources go to those best able to utilize them. Thus, in dividing an estate, the inheritance of a piano might be fairly given to the most musically-inclined family member.

It is worth noting that the definitions do not all agree with each other or even point in the same direction in particular cases, particularly #2 and #3.

Fairness in machine learning is most often framed in terms of rubric #1. If government regulations require non-discriminatory treatment, organizations must comply. Such a stance has the promotes a legalistic approach in which organizations treat algorithmic fairness as just another compliance concern along with environmental regulations or financial disclosures. Note that such regulations themselves are inevitably the product of societal embrace of one or more of the other fairness rubrics, as applied to a particular issue.

A legalistic approach to fairness sometimes results in a “head-in-the-sand” approach. An organization may avoid evaluating its systems for discriminatory outcomes, because such a finding would incur liability for remedying the problem. Costs for building and maintaining fairness properties in systems can thus be deferred or avoided by neglecting to test for them, leaving it to external parties (usually those facing discrimination) to identify unfair impacts and seek redress under the law. Such an approach may be practically appealing, but it is not ethical for system designers who can reasonably anticipate such harms not to seek to mitigate them in advance: note the assertion in the ACM’s code of ethics that computing professionals should strive to “minimize negative consequences of computing.”

In addition to leading organizations to avoid proactive approaches, the focus on legalistic, exogenous aspects of fairness tends to reduce the issue to a matter of meeting some specific court-established legal test. This hides something crucial about fairness: namely, its contested and inevitably political nature. Regulations around discrimination or other fairness concerns do not arise from the sage wisdom of legal scholars; they arise from multi-vocal contestation in a political sphere where claims to various kinds of rights are asserted, argued and eventually codified (or not) through regulatory action.

Thus, as designers and developers of intelligent systems, it is incumbent on us to be proactive in the incorporation of fairness into system design. In doing so, we will further need to recognize that our view of fairness will necessarily shift from a focus on

fixed constraints imposed from *without*, to an understanding of fairness as a nuanced, multiply-interpreted, and sometimes-contested construct derived from *within* a real-world organizational context, where different stakeholders understand fairness in different ways. Such a point of view is consistent with scholarship in sociology [3], organizational justice [2], welfare economics [9], and public administration [6].

As examples of this perspective in action, we can consider cases of systems operating in organizational contexts where fairness derives from organizational mission. In these cases, the legalistic framing is less salient and the contested nature of fairness is more evident. In this talk, we will look particularly at recommendation in two such contexts: the non-profit microlending site Kiva.org and a (planned) open news recommender site. We will examine the different claims to fairness that arise and how the design and implementation of intelligent systems in these areas can be open to the on-going dialog between them.

References

1. Chouldechova, A.: Fair prediction with disparate impact: a study of bias in recidivism prediction instruments. *Big data* **5**(2), 153–163 (2017)
2. Colquitt, J.A.: On the dimensionality of organizational justice: a construct validation of a measure. *J. Appl. Psychol.* **86**(3), 386 (2001)
3. Cook, K.S., Hegtvedt, K.A.: Distributive justice, equity, and equality. *Ann. Rev. Sociol.* **9** (1), 217–241 (1983)
4. Cramer, H., Holstein, K., Vaughan, J.W., Daumé III, H., Dudík, M., Wallach, H., Reddy, S., Garcia-Gathright, J.: Challenges of incorporating algorithmic fairness into industry practice (2019)
5. Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.: Fairness through awareness. In: *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. pp. 214–226. ACM (2012)
6. Frederickson, H.G.: *Social Equity and Public Administration: Origins, Developments, and Applications: Origins, Developments, and Applications*. Routledge (2015)
7. Hardt, M., Price, E., Srebro, N., et al.: Equality of opportunity in supervised learning. In: *Advances in Neural Information Processing Systems*. pp. 3315–3323 (2016)
8. Kamiran, F., Calders, T., Pechenizkiy, M.: Discrimination aware decision tree learning. In: *2010 IEEE 10th International Conference on Data Mining (ICDM)*, pp. 869–874. IEEE (2010)
9. Moulin, H.: *Fair Division and Collective Welfare*. MIT press (2004)
10. Narayanan, A.: Translation tutorial: 21 fairness definitions and their politics. In: *Proceedings of the Conference on Fairness Accountability and Transparency*, New York, USA (2018)
11. Pedreshi, D., Ruggieri, S., Turini, F.: Discrimination-aware data mining. In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 560–568. ACM (2008)
12. Zemel, R., Wu, Y., Swersky, K., Pitassi, T., Dwork, C.: Learning fair representations. In: *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pp. 325–333 (2013)
13. Zhang, L., Wu, X. Anti-discrimination learning: a causal modeling-based framework. *Int. J. Data Sci. Anal.* **4**, 1–16 (2017). <https://doi.org/10.1007/s41060-017-0058-x>

Brevity

Kristina Gligorić¹ and Ashton Anderson² and Robert West¹

¹ EPFL, Lausanne, Switzerland

{robert.west, kristina.gligoric}@epfl.ch

² University of Toronto, Toronto, Ontario, Canada

ashton@cs.toronto.edu

Abstract. In online communities, where billions of people strive to propagate their messages, understanding how wording affects success is of primary importance. In this work, we are interested in one particularly salient aspect of wording: brevity. What is the causal effect of brevity on message success? What are the linguistic traits of brevity? When is brevity beneficial, and when is it not? Whereas most prior work has studied the effect of wording on style and success in observational setups, we conduct a controlled experiment, in which crowd workers shorten social media posts to prescribed target lengths and other crowd workers subsequently rate the original and shortened versions. This allows us to isolate the causal effect of brevity on the success of a message. We find that concise messages are on average more successful than the original messages up to a length reduction of 30–40%. The optimal reduction is on average between 10% and 20%. The observed effect is robust across different subpopulations of raters and is the strongest for raters who visit social media on a daily basis. Finally, we discover unique linguistic and content traits of brevity and correlate them with the measured probability of success in order to distinguish effective from ineffective shortening strategies. Overall, our findings are important for developing a better understanding of the effect of brevity on the success of messages in online social media.

Keywords: Causal effects · Experimental methods · Linguistic style · Twitter · Crowdsourcing

1 Introduction

How to convey a message most successfully is an age-old question. In online communities, where billions of people consume and strive to propagate their messages, understanding how wording affects success is of primary importance. In this work, we are interested in one particularly salient aspect of wording: brevity, or conciseness. What is the causal effect of brevity on message success? What are the linguistic traits of brevity? When is brevity beneficial? To establish a causal link between brevity and success, one would need to compare posts that convey the exact same semantic

This is an abridged version of a longer paper with a longer title [4].

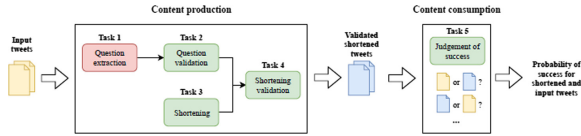


Fig. 1. Schematic diagram of the experimental design. The experiment consists of two parts, designed to replicate the *production* and *consumption* of textual content in online social media. The goal of the content production part (Tasks 1–4) is, for a given set of input tweets, to output shortened versions, having validated that the meaning is preserved. In the content consumption part of the experiment (Task 5), we show participants pairs of tweets, one treated short tweet, and the control long original tweet and ask which one will get more retweets. The outputs of the setup are binary votes (several per pair), based on which we compute the probabilities of success for each tweet version.

information and differ only in the number of characters used to express the fixed semantic content in a specific lexical and syntactic surface form.

Observational studies have striven to approximate this ideal goal by carefully controlling for confounding factors [2, 3, 6]. However, prior research arrived at contradictory conclusions and has not been able to guarantee that the semantic content of compared posts is identical, or that length is not confounded with other factors such as the inherent attractiveness of a message.

2 Experimental Setup

In order to overcome the aforementioned methodological hurdle inherent in observational designs and to more closely approximate the ideal of comparing two messages—one long, one short—expressing the exact same semantic content, we adopt an experimental approach instead, depicted schematically in Fig. 1.

Our experimental strategy consists of two steps: first, in the content production phase, we extract shortened versions of original tweets, and second, in the content consumption phase, we measure the quality of shortened versions compared with the unshortened version.

By ensuring (via additional checks) that length is the only difference between the unshortened tweets and the tweets shortened to prespecified lengths, we can attribute differences in quality to be causally related to brevity. Building on the fact that regular crowd workers can reduce the length of text by up to 70% without cutting any major content [1] and can accurately estimate which of two messages in a pair (for a fixed topic and user) will be shared more frequently [6], we deployed our experiment on Amazon Mechanical Turk.

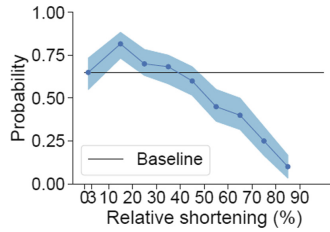


Fig. 2. The effect of conciseness as a function of the level of reduction, with 95% bootstrapped confidence interval.

3 Results

To the best of our knowledge, we are the first to study the effect of brevity in online social media in an experimental fashion. Applying our experimental framework, we collect short versions at 9 brevity levels for 60 original tweets, judged against the original tweets in a total of 27,000 binary votes. Based on this dataset, we address our research questions. We find that concise versions are on average more successful than the original messages up to a length reduction of 45%, while the optimal reduction is on average between 10% and 20% (resulting length 211–215 characters, down from the original 250) (Fig. 1). The observed effect is robust across different subpopulations of participants. Studying linguistic traits of brevity, we find that the shortening process disproportionately preserves verbs and negations—parts of speech that carry essential information—in contrast to, e.g., articles and adverbs. The shortening also preserves affect and subjective perceptions (quantified via the LIWC dictionary), and the effect is strongest for negative emotions. Addressing the question of when brevity is beneficial and when it is not, we find initial evidence that it is effective to omit certain function words and to insert commas and full stops, presumably as it increases readability by structuring or splitting long sentences. Ineffective editing strategies include deleting hashtags as well as question and exclamation marks, which have the potential to elicit discussion and reactions.

4 Discussion

In this work, our goals are threefold: to measure the effects of length constraints on tweet quality, to determine the linguistic traits of brevity, and to find when brevity is beneficial. To address these goals, we designed a large experiment that measured the quality of the shortened versions compared with the originals.

In brief, there are significant benefits of brevity. We observe that tweets can be successfully reduced up to 45% of their original length with no reduction in quality. The optimal range of shortening is consistently between 10% and 20% of the original length.

Practically, our findings are important for developing a better understanding of the effect of wording on the success of messages in online social media.

Methodologically, our experimental design highlights the power of a novel data collection paradigm, where crowd workers supply research data in the form of slightly edited text [1, 5, 7].

References

1. Bernstein, M.S., Little, G., Miller, R.C., Hartmann, B., Ackerman, M.S., Karger, D.R., Crowell, D., Panovich, K.: Soylent: a word processor with a crowd inside. *Commun. ACM*, **58**(8), 85–94 (2015)
2. Bramoulle, Y., Ductor, L.: Title length. *J. Econ. Behav. Organ.* **150**, 311–324 (2018)
3. Gligorić, K., Anderson, A., West, R.: How constraints affect content: the case of Twitter’s switch from 140 to 280 characters. In: *Proceedings of the International Conference on Web and Social Media (ICWSM)* (2018)
4. Gligorić, K., Anderson, A., West, R.: Causal effects of brevity on style and success in social media. In: *Proceedings of the ACM Conference on Computer-Supported Cooperative Work and Social (CSCW)* (2019)
5. Ribeiro, M.H., Gligorić, K., West, R.: Message distortion in information cascades. In: *Proceedings of the World Wide Web Conference (WWW)* (2019)
6. Tan, C., Lee, L., Pang, B.: The effect of wording on message propagation: Topic- and author-controlled natural experiments on Twitter. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)* (2014)
7. West, R., Horvitz, E.: Reverse-engineering satire, or paper on computational humor accepted despite making serious advances. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)* (2019)

Contents

Invited Talk

Complementing Behavioural Modeling with Cognitive Modeling for Better Recommendations.	3
<i>Marko Tkalčič</i>	

Nominees for Best Paper Award

The Construction of Action Rules to Raise Artwork Prices.	11
<i>Laurel Powell, Anna Gelich, and Zbigniew W. Ras</i>	
Metric-Guided Multi-task Learning	21
<i>Jinfu Ren, Yang Liu, and Jiming Liu</i>	
Sentiment Analysis with Contextual Embeddings and Self-attention.	32
<i>Katarzyna Biesialska, Magdalena Biesialska, and Henryk Rybinski</i>	

Nominees for Best Student Paper Award

Interpretable Segmentation of Medical Free-Text Records Based on Word Embeddings	45
<i>Adam Gabriel Dobrakowski, Agnieszka Mykowiecka, Małgorzata Marciniak, Wojciech Jaworski, and Przemysław Biecek</i>	
Decision-Making with Probabilistic Reasoning in Engineering Design	56
<i>Stefan Plappert, Paul Christoph Gembarski, and Roland Lachmayer</i>	
Hyperbolic Embeddings for Hierarchical Multi-label Classification	66
<i>Tomáš Stepišnik and Dragi Kocev</i>	

Natural Language Processing

Joint Multiclass Debiasing of Word Embeddings.	79
<i>Radomir Popović, Florian Lemmerich, and Markus Strohmaier</i>	
Recursive Neural Text Classification Using Discourse Tree Structure for Argumentation Mining and Sentiment Analysis Tasks.	90
<i>Alexander Chernyavskiy and Dmitry Ilvovsky</i>	
Named Entity Recommendations to Enhance Multilingual Retrieval in Europeana.eu	102
<i>Sergiu Gordea, Monica Lestari Paramita, and Antoine Isaac</i>	

A Deep Learning Approach to Fake News Detection. 113
*Elio Masciari, Vincenzo Moscato, Antonio Picariello,
and Giancarlo Sperli*

Satirical News Detection with Semantic Feature Extraction and Game-
Theoretic Rough Sets 123
Yue Zhou, Yan Zhang, and JingTao Yao

Deep Learning and Embeddings

Comparing State-of-the-Art Neural Network Ensemble Methods in Soccer
Predictions 139
Tiago Mendes-Neves and João Mendes-Moreira

Static Music Emotion Recognition Using Recurrent Neural Networks 150
Jacek Grekow

Saliency Detection in Hyperspectral Images Using Autoencoder-Based
Data Reconstruction 161
*Annalisa Appice, Francesco Lomuscio, Antonella Falini,
Cristiano Tamborrino, Francesca Mazzia, and Donato Malerba*

Mesoscale Anisotropically-Connected Learning. 171
Qi Tan, Yang Liu, and Jiming Liu

Empirical Comparison of Graph Embeddings for Trust-Based
Collaborative Filtering 181
*Tomislav Duricic, Hussain Hussain, Emanuel Lacic, Dominik Kowald,
Denis Helic, and Elisabeth Lex*

Neural Spike Sorting Using Unsupervised Adversarial Learning 192
Konrad A. Ciecierski

Digital Signal Processing

Poriferal Vision: Classifying Benthic Sponge Spicules to Assess Historical
Impacts of Marine Climate Change 205
Saketh Saxena, Philip Heller, Amanda S. Kahn, and Ivano Aiello

Experimental Evaluation of GAN-Based One-Class Anomaly Detection
on Office Monitoring. 214
*Ning Dong, Yusuke Hatae, Muhammad Fikko Fadjrimiratno,
Tetsu Matsukawa, and Einoshin Suzuki*

Ranking Speech Features for Their Usage in Singing
Emotion Classification. 225
Szymon Zaporowski and Bozena Kostek

Leveraging Machine Learning in IoT to Predict the Trustworthiness of Mobile Crowd Sensing Data. 235
Corrado Loglisci, Marco Zappatore, Antonella Longo, Mario A. Bochicchio, and Donato Malerba

A Hierarchical-Based Web-Platform for Crowdsourcing Distinguishable Image Patches. 245
Ayman Hajja and Justin Willis

Performing Arithmetic Using a Neural Network Trained on Digit Permutation Pairs 255
Marcus D. Bloice, Peter M. Roth, and Andreas Holzinger

Modelling and Reasoning

CatIO - A Framework for Model-Based Diagnosis of Cyber-Physical Systems. 267
Edi Muškardin, Ingo Pill, and Franz Wotawa

Data Publishing: Availability of Data Under Security Policies. 277
Juba Agoun and Mohand-Saïd Hacid

Matrix Factorization Based Heuristics Learning for Solving Constraint Satisfaction Problems. 287
Seda Polat Erdeniz, Ralph Samer, and Muesluem Atas

Explaining Object Motion Using Answer Set Programming 298
Franz Wotawa and Lorenz Klampfl

The GraphBRAIN System for Knowledge Graph Management and Advanced Fruition. 308
Stefano Ferilli and Domenico Redavid

Mining Exceptional Mediation Models. 318
Florian Lemmerich, Christoph Kiefer, Benedikt Langenberg, Jeffrey Cacho Aboukhalil, and Axel Mayer

Machine Learning Applications

Multivariate Predictive Clustering Trees for Classification 331
Tomaž Stepišnik and Dragi Kocev

Comparison of Machine Learning Methods to Detect Anomalies in the Activity of Dairy Cows. 342
Nicolas Wagner, Violaine Antoine, Jonas Koko, Marie-Madeleine Mialon, Romain Lardy, and Isabelle Veissier

Clustering Algorithm Consistency in Fixed Dimensional Spaces	352
<i>Mieczysław Alojzy Kłopotek and Robert Albert Kłopotek</i>	
Estimating the Importance of Relational Features by Using Gradient Boosting	362
<i>Matej Petković, Michelangelo Ceci, Kristian Kersting, and Sašo Džeroski</i>	
Multi-objective Discrete Moth-Flame Optimization for Complex Network Clustering	372
<i>Xingjian Liu, Fan Zhang, Xianghua Li, Chao Gao, and Jiming Liu</i>	
Predicting Associations Between Proteins and Multiple Diseases	383
<i>Martin Breskvar and Sašo Džeroski</i>	
Short Papers	
Exploiting Answer Set Programming for Building explainable Recommendations	395
<i>Erich Teppan and Markus Zanker</i>	
Tailoring Random Forest for Requirements Classification.	405
<i>Andreas Falkner, Gottfried Schenner, and Alexander Schörghuber</i>	
On the Design of a Natural Logic System for Knowledge Bases	413
<i>Troels Andreasen, Henrik Bulskov, and Jørgen Fischer Nilsson</i>	
Evaluation of Post-hoc XAI Approaches Through Synthetic Tabular Data . . .	422
<i>Julian Tritscher, Markus Ring, Daniel Schlr, Lena Hettinger, and Andreas Hotho</i>	
SimLoss: Class Similarities in Cross Entropy	431
<i>Konstantin Kobs, Michael Steininger, Albin Zehe, Florian Lautenschlager, and Andreas Hotho</i>	
Efficient and Precise Classification of CT Scannings of Renal Tumors Using Convolutional Neural Networks.	440
<i>Mikkel Pedersen, Henning Christiansen, and Nessn H. Azawi</i>	
Deep Autoencoder Ensembles for Anomaly Detection on Blockchain	448
<i>Francesco Scicchitano, Angelica Liguori, Massimo Guarascio, Ettore Ritacco, and Giuseppe Manco</i>	

A Parallelized Variant of Junker’s QUICKXPLAIN Algorithm 457
*Cristian Vidal Silva, Alexander Felfernig, Jose Galindo, Müslüm Atas,
and David Benavides*

Author Index 469

Invited Talk



Complementing Behavioural Modeling with Cognitive Modeling for Better Recommendations

Marko Tkalčič^(✉) 

Faculty of Mathematics, Natural Sciences and Information Technologies,
University of Primorska, Glagoljaška 8, 6000 Koper, Slovenia
`marko.tkalcic@famnit.upr.si`

Abstract. Recommender systems are systems that help users in decision-making situations where there is an abundance of choices. We can find them in our everyday lives, for example in online shops. State-of-the-art research in recommender systems has shown the benefits of behavioural modeling. Behavioural modeling means that we use past ratings, purchases, clicks etc. to model the user preferences. However, behavioural modeling is not able to capture certain aspects of the user preferences. In this talk I will show how the usage of complementary research in cognitive models, such as personality and emotions, can benefit recommender systems.

Keywords: Recommender systems · Behavioural modeling · Cognitive modeling

1 Introduction

Recommender Systems (RS) help people make better choices when there is an abundance of choice. Early approaches were limited in finding, for each user, such items that would maximize the user's utility [1]. Examples of this kind of recommender systems can be found in online services, for example in Amazon or Netflix. Today, the RS research community is addressing more diverse topics than plain utility prediction, such as multiple stakeholders (company's profit vs. user satisfaction), trust and transparency, preference elicitation, diversity, group recommendations and new algorithms (e.g. Deep Learning).

In the vast majority of cases, the RS community is addressing the issues above by collecting user behaviour data (e.g. clicks, purchases, viewing time etc.) and using machine learning algorithms to perform the necessary predictions. I will refer to the usage of behavioural data to do recommendations as *behavioural modeling*.

I argue that it's important to understand not only the user behaviour, but also the reasoning that a user does before performing a certain action. I will refer to the modeling of the user's cognitive processes as *cognitive modeling*. I will demonstrate the need for complementing behavioural modeling with cognitive modeling with three stories.

1.1 The Netflix Story

Netflix is known for its heavy use of recommendations in their user interface. Neil Hunt from Netflix, in his keynote speech at the ACM RecSys 2014 conference, explained how they approached the recommendations¹ [10]. He stated that Netflix is optimizing for viewing hours. However, not all viewing hours are equal and their system did not distinguish between customer value and addiction.

1.2 The Nature and Nurture Story

This is an old dilemma, which debates how much of the human mind is built-in, and how much of it is constructed by experience. Throughout history there have been several manifestations of this dilemma. A prominent one has been the debate between Skinner and Chomsky on language acquisition. Skinner believed that children learned language starting as a tabula rasa through reinforcement, while Chomsky claimed that children have innate capabilities to learn language, the so-called Language Acquisition Device (LAD) [2]. Today, a similar debate is going on in machine learning between connectionists, who model mental phenomena with neural nets trained with behavioural data, and symbolists, who model them with ingrained concepts. The debate has been documented heavily online². In this debate, Gary Marcus is arguing to merge both behaviour-based modeling and cognitive modeling: *To get computers to think like humans, we need a new A.I. paradigm, one that places top down and bottom up knowledge on equal footing. Bottom-up knowledge is the kind of raw information we get directly from our senses, like patterns of light falling on our retina. Top-down knowledge comprises cognitive models of the world and how it works*³.

1.3 The “La vita e’ bella” Story

The third story revolves around the film *Life is Beautiful* (La vita e’ bella). The film narrates a sad story that takes place during the holocaust. It is also full of hilarious scenes, such as the one in which the main character, played by Roberto Benigni, translates the instructions of a German officer. In a movie recommender system, if a user gives this film a high rating, the system doesn’t really know if the user liked it because of the excellent drama or because of the hilarious jokes. Again, we see that observing only the behaviour (i.e. the high rating) we may miss important information about the user, which can lead to bad recommendations.

Based on these three stories we can conclude that understanding the cognitive processes of the user is important to interpret correctly the behaviour. It is hard, especially because the data is not readily available.

¹ <https://youtu.be/IYcDR8z-rRY?t=4727>.

² <https://twitter.com/ylecun/status/1201233472989356032>.

³ Marcus, Gary, Artificial Intelligence Is Stuck. Here’s How to Move It Forward. New York Times, July 29, 2017.

2 Cognitive Modeling in Recommender Systems

There are many models that describe various cognitive processes. In one such model of decision making, two important factors that influence decision-making are emotions and personality [13]. Other models of the human mind also include emotions and personality, for example the model of artificial general intelligence [8].

2.1 Personality and Emotions

Personality is a set of traits that are supposed to be relatively stable across long periods of time. The most widely used model of personality is the five-factor model (FFM), which is composed of the following factors: openness, conscientiousness, extraversion, agreeableness, and neuroticism [14]. Personality can be acquired either using extensive questionnaires, such as the Big-Five Inventory [14] or using algorithms that predict personality from social media usage [7, 11, 16, 18].

Emotions, on the other hand, change very rapidly with time. We usually model them either with discrete labels (e.g. happy, sad, angry etc. [3] or with dimensions, such as valence, arousal and dominance [17]. There are off-the-shelf solutions that use various sensors (e.g. cameras, physiological sensors etc.) to infer the current emotion of a person⁴.

2.2 Personality in Recommender Systems

Personality has been used in a variety of ways to improve recommendations. For example, personality has been shown to be important for mood regulation. In an experiment by Ferwerda et al., the authors induced various moods in people using film music. Among other things, they observed that when people are sad they don't always regulate their mood towards being happy. People with different personalities had different needs in such situations, which has clear implications for personality-mood-based music recommendations [6]. Personality has also shown to be useful in calculating user similarities for neighbourhood-based RS. Both, Tkalcic et al. and Hu et al., managed to solve the new user problem by introducing personality in the calculation of user similarity [9, 20]. Matrix factorization-based RS were also augmented successfully with personality information [4, 5]. Finally, personality has been shown to be useful also for adjusting the diversity of recommended items [25].

2.3 Emotions in Recommender Systems

Emotions have also been used in different ways to improve recommendations. Odic et al., as well as Zheng et al., have shown in a series of experiments that

⁴ <https://www.affectiva.com/>.

emotions are useful as contextual variables [15,26]. In an early adoption of emotions, Tkalčić et al. showed that the induced emotions of images are effective features for doing content-based recommendations [19,23]. Emotions are also useful as feedback, which can be used without disrupting the user interaction with the system. Vodlan et al. used the emotion of hesitation to infer whether the list of recommended items is good [24]. Emotions were also used in a pairwise preference acquisition system to replace the user’s explicit rating [21,22]. Finally, emotions should be taken in account also in group recommendation settings. In group dynamics, user preferences are influenced by other group members through emotional contagion. The impact of emotional contagion has been demonstrated also in virtual groups, such as the experiment conducted on Facebook users [12].

3 Conclusion

Recommender systems algorithms take advantage of past behaviour of users to make recommendations. However, this behaviouristic approach does not provide explanations for the user behaviour, which can lead to bad recommendations. In this contribution I surveyed how cognitive modeling, in particular the usage of personality and emotions, can be used to make better recommendations.

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.* **17**(6), 734–749 (2005). <https://doi.org/10.1109/TKDE.2005.99>, <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1423975>
2. Chomsky, N.: A review of BF skinner’s verbal behavior. *Readings Philos. Psychol.* **1**, 48–63 (1980)
3. Ekman, P.: Basic emotions. In: Dalglish, T., Power, M.J. (eds.) *Handbook of Cognition and Emotion*, vol. 1992, pp. 45–60. John Wiley & Sons Ltd., Chichester (2005). <https://doi.org/10.1002/0470013494.ch3>
4. Elahi, M., Braunhofer, M., Ricci, F., Tkalčić, M.: Personality-based active learning for collaborative filtering recommender systems. In: Baldoni, M., Baroglio, C., Boella, G., Micalizio, R. (eds.) *AI*IA 2013. LNCS (LNAI)*, vol. 8249, pp. 360–371. Springer, Cham (2013). https://doi.org/10.1007/978-3-319-03524-6_31
5. Fernández-Tobías, I., Braunhofer, M., Elahi, M., Ricci, F., Cantador, I.: Alleviating the new user problem in collaborative filtering by exploiting personality information. *User Model. User-Adapt. Inter.* **26**(2), 1–35 (2016). <https://doi.org/10.1007/s11257-016-9172-z>
6. Ferwerda, B., Schedl, M., Tkalčić, M.: Personality & emotional states : understanding users music listening needs. In: Cristea, A., Masthoff, J., Said, A., Tintarev, N. (eds.) *UMAP 2015 Extended Proceedings* (2015). <http://ceur-ws.org/Vol-1388/>
7. Ferwerda, B., Tkalčić, M.: Predicting users’ personality from instagram pictures. In: *Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization - UMAP 2018*, pp. 157–161. ACM Press, New York (2018). <https://doi.org/10.1145/3209219.3209248>, <http://dl.acm.org/citation.cfm?doid=3209219.3209248>

8. Goertzel, B., Iklé, M., Wigmore, J.: The Architecture of Human-Like General Intelligence, pp. 123–144. Atlantis Press, Paris (2012). https://doi.org/10.2991/978-94-91216-62-6_8
9. Hu, R., Pu, P.: Using personality information in collaborative filtering for new users. In: Proceedings of the 2nd ACM RecSys 2010 Workshop on Recommender Systems and the Social Web, pp. 17–24 (2010). http://www.dcs.warwick.ac.uk/~ssanand/RSWeb_files/Proceedings_RSWEB-10.pdf#page=23
10. Hunt, N.: Quantifying the value of better recommendations (2014)
11. Kosinski, M., Stillwell, D., Graepel, T.: Private traits and attributes are predictable from digital records of human behavior. *Proc. Natl. Acad. Sci. United States Am.* **110**(15), 5802–5805 (2013). <https://doi.org/10.1073/pnas.1218772110>
12. Kramer, A.D.I., Guillory, J.E., Hancock, J.T.: Experimental evidence of massive-scale emotional contagion through social networks. *Proc. Natl. Acad. Sci. United States Am.* **111**(29), 8788–8790 (2014). <https://doi.org/10.1073/pnas.1320040111>
13. Lerner, J.S., Li, Y., Valdesolo, P., Kassam, K.S.: Emotion and decision making. *Ann. Rev. Psychol.* **66**(1), 799–823 (2015). <https://doi.org/10.1146/annurev-psych-010213-115043>
14. McCrae, R.R., John, O.P.: An introduction to the five-factor model and its applications. *J. Pers.* **60**(2), 175–215 (1992)
15. Odić, A., Tkalčič, M., Tasič, J.F., Košir, A.: Predicting and detecting the relevant contextual information in a movie-recommender system. *Inter. Comput.* **25**(1), 74–90 (2013). <https://doi.org/10.1093/iwc/iws003>, <http://iwc.oxfordjournals.org/content/25/1/74.short>, <https://academic.oup.com/iwc/article/768060/Predicting>
16. Quercia, D., Kosinski, M., Stillwell, D., Crowcroft, J.: Our twitter profiles, our selves: predicting personality with twitter. In: Proceedings - 2011 IEEE International Conference on Privacy, Security, Risk and Trust and IEEE International Conference on Social Computing, PASSAT/SocialCom 2011, pp. 180–185. IEEE (2011). <https://doi.org/10.1109/PASSAT/SocialCom.2011.26>, <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=611311> http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=611311
17. Scherer, K.R.: What are emotions? and how can they be measured? *Soc. Sci. Inf.* **44**(4), 695–729 (2005). <https://doi.org/10.1177/0539018405058216>
18. Skowron, M., Tkalčič, M., Ferwerda, B., Schedl, M.: Fusing social media cues. In: Proceedings of the 25th International Conference Companion on World Wide Web - WWW 2016 Companion, pp. 107–108. ACM Press, New York (2016). <https://doi.org/10.1145/2872518.2889368>, <http://dl.acm.org/citation.cfm?doid=2872518.2889368>
19. Tkalčič, M., Burnik, U., Košir, A.: Using affective parameters in a content-based recommender system for images. *User Model. User-Adapt. Inter.* **20**(4), 279–311 (2010). <https://doi.org/10.1007/s11257-010-9079-z>
20. Tkalčič, M., Kunaver, M., Košir, A., Tasič, J.: Addressing the new user problem with a personality based user similarity measure. In: Ricci, F., et al. (eds.) Joint Proceedings of the Workshop on Decision Making and Recommendation Acceptance Issues in Recommender Systems (DEMRA 2011) and the 2nd Workshop on User Models for Motivational Systems: The Affective and the Rational Routes to Persuasion (UMMS 2011) (2011). http://ceur-ws.org/Vol-740/DEMRA_UMMS_2011_proceedings.pdf#page=106

21. Tkalčič, M., Maleki, N., Pesek, M., Elahi, M., Ricci, F., Marolt, M.: A research tool for user preferences elicitation with facial expressions. In: Proceedings of the Eleventh ACM Conference on Recommender Systems - RecSys 2017, no. i, pp. 353–354. ACM Press, New York (2017). <https://doi.org/10.1145/3109859.3109978>, <http://dl.acm.org/citation.cfm?doid=3109859.3109978>
22. Tkalčič, M., Maleki, N., Pesek, M., Elahi, M., Ricci, F., Marolt, M.: Prediction of music pairwise preferences from facial expressions. In: Proceedings of the 24th International Conference on Intelligent User Interfaces - IUI 2019, pp. 150–159. ACM Press, New York (2019). <https://doi.org/10.1145/3301275.3302266>, <http://dl.acm.org/citation.cfm?doid=3301275.3302266>
23. Tkalčič, M., Odic, A., Kosir, A., Tasic, J.: Affective labeling in a content-based recommender system for images. *IEEE Trans. Multimedia* **15**(2), 391–400 (2013). <https://doi.org/10.1109/TMM.2012.2229970>, <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6362231>
24. Vodlan, T., Tkalčič, M., Košir, A.: The impact of hesitation, a social signal, on a user's quality of experience in multimedia content retrieval. *Multimedia Tools Appl.* **74**(17), 6871–6896 (2015). <https://doi.org/10.1007/s11042-014-1933-2>
25. Wu, W., Chen, L., He, L.: Using personality to adjust diversity in recommender systems. In: Proceedings of the 24th ACM Conference on Hypertext and Social Media - HT 2013, pp. 225–229 (2013). <https://doi.org/10.1145/2481492.2481521>, <http://dl.acm.org/citation.cfm?doid=2481492.2481521>
26. Zheng, Y., Mobasher, B., Burke, R.: Emotions in Context-Aware Recommender Systems, pp. 311–326 (2016). https://doi.org/10.1007/978-3-319-31413-6_15

Nominees for Best Paper Award



The Construction of Action Rules to Raise Artwork Prices

Laurel Powell¹  , Anna Gelich^{2,3}, and Zbigniew W. Ras^{1,3} 

¹ College of Computing and Informatics, University of North Carolina at Charlotte,
Charlotte, USA

{lpowel28,ras}@uncc.edu

² College of Arts + Architecture, University of North Carolina at Charlotte,
Charlotte, USA

agelich@uncc.edu

³ Polish-Japanese Academy of Information Technology, Warsaw, Poland

Abstract. This work explores the development of action rules for changing the prices of works of contemporary fine art. It focuses on the generation of action rules using LISp-Miner related to artwork profiles and artist descriptions. Additionally, this work explores the use of the dominant color of an artwork as a feature in the generation of action rules for adjusting its prices.

Keywords: Art analytics · Data mining · Action rules

1 Introduction

Artists in the contemporary market for fine art must make many decisions on how to present their work for commercial sale. While these choices are crucial to the artist's long term commercial success, artists are given limited guidance on how to present their work online to its best advantage. Can the price that an artist can ask for regarding a work be altered by changing minor aspects that their potential customers see? What drives the price for a work of fine art?

The challenge of objective valuation is worsened by the complex nature of pricing in the art market. There are a number of theories on how pricing in the art market works. As said in [35],

In a market that has to reconcile a fierce opposition between commercial and artistic values, and that has to commodify goods whose essence is considered to be non-commodifiable, dealers, artists, and collectors find ways to express and share non-economic values through the economic medium of pricing.

This work proposes a method for developing a set of action rules that can be used to suggest courses of action to artists that will improve the sales potential of their work. It begins by discussing the development of a dataset and feature

construction for the generation of action rules, moves on to contrast the results of different sets of features for the creation of these rules, and it concludes by discussing the future potential of this research avenue. These rules reclassify artworks from one price level to another higher price level. This work discusses possible sets of flexible attributes and proposes a basic set of stable attributes [24]. Stable attributes are generally aspects of the relevant item that are unlikely or difficult to change. In the context of this work, these represent the physical aspects of the artwork and fundamental qualities such as its medium. Flexible attributes are features that can be altered by an interested party to change its classification. In the context of this work, these represent aspects of how the artist presents themselves to potential customers.

2 Related Work

Applying computing techniques to artworks has received considerable attention from the academic community. Some researchers have addressed the challenge of creating a recommender system that picks out artworks similar to a given one for a user [29], while others are focused on automatically tagging a painting with emotions based on the colors represented [15].

Art price valuation has received limited attention. One interesting project in the realm of price prediction was Hosny’s work [13]. As discussed in [7], one notable trend explored in Hosny’s work was that certain colors, specifically blacks, whites and grays, are more likely to have a higher sales valuation. Different approaches have been used for price prediction, such as the approach used by [18] to value Rothko paintings. In [11], hedonic regression, a statistical modelling method based on characteristics of the object in question, combined with past sales was used to predict art value.

Action rules were first proposed in [24] as a method of reclassifying objects from one group to another group by changing values of their flexible attributes. They have been used for medical data, such as in [12] and for business purposes, as in [24]. Some have explored expansions on the original methodology, such as considering the cost and feasibility of the implementation of action rules as discussed in [33,34]. The cost of an action rule, which was initially proposed in [34], represents the average cost of changing an attribute from its initial value to another specific value.

This work uses LISp-Miner’s Ac4ft-Miner tool, introduced in [25], discussed in [19], and developed by [26]. This is an implementation of the GUHA method for action rule generation [25]. In this method, objects resembling traditional association rules are extracted from the dataset and used to form contingency tables [25]. Two of these objects, with matching stable attributes, are used to construct the action rules. One rule, termed here as the ‘before rule’, has the stable and flexible attributes of the object before any actions are taken and the other rule, called the ‘after rule’ here, has the desired set of flexible attributes and matching stable ones. Rules generated in this manner consist of an antecedent and succedent, which are association rules in the form $\varphi \approx \psi$ where the symbol \approx

represents the association between φ and ψ [25]. These are then used to create action rules, called G-Action Rules, through the examination of dependencies and similarities across the rules generated in the previous step [25].

3 Dataset

The dataset used in this work is an expanded version of the one used in [21–23]. Our work expands on the ideas discussed in those works by adding action rules to the original dataset and proposed feature sets.

We use a dataset of artworks and artist information collected from the online art sales site [Artfinder.com](https://www.artfinder.com) [5]. Artfinder represents artists from all over the world, and has diverse styles and subjects represented. It functions as a platform to allow artists to sell directly to consumers. Using a single source for artworks allows for more consistent definitions of tags used for artworks. The information was scraped using Beautiful Soup [2] to parse the pages, Apache Selenium [4] to work with dynamic webpages and Javascript, and Python to fetch pages and all other major steps. It contains approximately 200,000 artworks from approximately 3,300 artists.

All artworks posted on Artfinder have a dedicated page describing the work. This page consistently has at least one photograph of the work, and has its descriptive information, as well as the tags that the artist selected. These tags are used by potential customers to search for works. These tags are simple and specify the features such as the artistic style, subject or medium of the artwork. Some of the analysis methods discussed below use the primary image of the artwork from the product page. A small number of artwork images could not be retrieved, so they are omitted from the following analysis.

Artists on Artfinder create profiles of themselves discussing their achievements and backgrounds. The profiles have places for artists to provide their biographies, links to their social media and descriptions of achievements in the artist’s life. A number of artists have reviews posted from customers, and are rated on a star system with five stars being the best. The majority of artists with one or more reviews have ratings of four out of five stars or higher, and very few have any low reviews. The average of all artists with star ratings score is 4.904. This makes the star ratings received by artists of very low value as predictive features.

4 Methodology

The decision feature considered here, which represents the succedent of the G-Action Rule, is the listed asking price of the artwork. Artists would attempt to make changes to how their artworks are presented to the public in order to increase their sales price, and should be cautious about making changes that could potentially lower their sales price. The prices were discretized into ten groups using LISp-Miner [26]. The cuts were placed to automatically to create partitions containing an approximately equal number of tuples. The price

attribute was partitioned into 10 levels, referred to in later sections as levels one through ten. In this work, price is treated as a flexible attribute.

This work defines a set of stable attributes, which are used in the antecedent portion of the G-Action Rule, that can be used to describe a work of art. The set of features used takes inspiration from the sets used in [20]. The chosen medium of the artwork is used as a stable feature. When the artwork was posted the artist tagged it with a medium for search purposes. In order to keep the number of options within reasonable bounds, the mediums were discretized into seven categories and one “NA” category. The artistic style of the work as well as the artist’s subject is similarly used as a tag and as a stable attribute. As discussed in Pawlowski [20], the size of a work is very relevant to determining its price. So the length and width of the works were discretized into five bins of roughly equal size using the LISP-Miner [26] discretization tool and used as stable attributes. Additionally, the presence or absence of visible reviews was used as a stable binary feature. The review scores of the artists are consistently very high if they are present at all. This limits the utility of using the score as a quality metric. Instead, the presence or absence of reviews was used as a stable attribute. Lastly, the percentage of edges in the work was utilized as a stable attribute after being discretized. To calculate this, Canny edge detection, which was developed by John Canny in [10], was used from the OpenCV set of tools [32] on the primary artwork image. A small percentage of artworks had errors when an attempt was made to retrieve the photograph, so those tuples were removed from consideration. This algorithm determines if an individual pixel represents an edge or not based on the amount of color variation surrounding it. The number of edge pixels and non-edge pixels are then counted, and the number of edge pixels was used as a stable feature. This can be considered as giving a rough idea of the amount of detail or amount of color variation in the work. The full list of stable attributes is given below.

- Artistic Style
- Artistic Subject
- Medium
- Height
- Width
- Artist Has Visible Reviews
- Percentage of the Artwork Representing Edges

Lastly, the use of color in the artwork was explored as an additional stable feature in one of the rule sets. The pixels in the artwork were clustered using Open-CV [32] across the RGB dimensions using K-Means to determine the 10 most frequently appearing colors. Out of this set of ten, the centroid of the largest cluster was tested against 11 reference colors using the CIEDE2000 color difference measure [30] which was implemented using [31]. The reference colors are based on the idea of universal basic colors which was first proposed in [8]. While this work has been challenged and the colors proposed have been refined such as in [17, 28], the concept of basic colors is useful for classification. The 11 reference colors are white (R 255, G 255, B 255), gray (R 128, G 128, B 128),

black (R 0, G 0, B 0), red (R 255, G 0, B 0), orange (R 255, G 128, B), yellow (R 255, G 255, B 0), green (R 0, G 255, B 0), blue (R 0, G 0, B 255), purple (R 128, G 0, B 128), pink (R 255, G 192.0, B 203), brown (R 63.8, G 47.9, B 31.9). The RGB values, other than brown, are taken from [3] and the brown comes from [16]. The selected color, referred to as main color, is used as a stable attribute representing the single color that takes up the largest portion of the work.

The set of flexible attributes, part of the antecedent of the rules, explored center around how a work will be perceived by a consumer. How long is the artist’s biography? Do they have a presence on social media? What is the tone of their writing? These are easily changeable for an artist hoping to improve their sales. This gives them a very low cost, so stakeholders may be more willing to try recommended rules. The full list of flexible features used is given below.

- Word Count of the Artist Biography (Bio. WC)
- Word Count of the Artwork Description (Desc. WC)
- Social Media (Considered Together as SM)
 - Artist Listed a Facebook Profile
 - Artist Listed a Twitter Profile
 - Artist Listed an Instagram Profile
- Positive Sentiment Level of Biography (Bio. Ps.)
- Negative Sentiment Level of Biography (Bio. Ng.)
- Positive Sentiment Level of Artwork Description (Desc. Ps.)
- Negative Sentiment Level of Artwork Description (Desc. Ng.)

The number of words in the biography of the artist and in the artwork description are used as predictive features. As was discussed in [21–23], the word count does have some utility as a price predicting feature. In other arenas of online sales, the length and wording of a description can have a bearing on the sale ability of an item. In [27], the authors analyzed sales on Ebay, and determined that the length of the description could have an impact on the price.

Similarly, many collectors find artists through social media [1]. It has become an increasingly important tool for helping artists be discovered by collectors. Can adding or removing a link to a profile change the opinion of a collector? Determining the sentiment of the text was done using VADER, the ‘Valence Aware Dictionary for sEntiment Reasoning’, which is part of the Python Natural Language Toolkit [9,14]. The objective was to determine the polarity of the sentiment of the text. This strategy uses a set of words and phrases to determine the text’s sentiment, called an ‘opinion lexicon’ [6]. VADER uses a combination of an opinion lexicon and a set of rules [14]. The lexicon includes thousands of candidate terms rated by humans on scale, and the rule set considers the impact of capitalization and negation [14]. Each piece of text was given scores as positive, negative or neutral. These scores were discretized and used as flexible attributes. The sentiments found were largely neutral, with only a few pieces of text containing strongly emotional language.

Using the LISp-Miner Ac4ft-Miner tool, sets of rules were generated for combinations of prices and flexible attributes with a single set of stable attributes

used throughout for consistency. As the goal is to move prices from lower to higher price points, rules were generated to transition items from lower price levels to higher price levels. Rules generated using this method have support and confidence scores for both the before state association rule and after state association rule that are used together to construct the final action rule.

5 Results

The first set of rules generated had no minimum confidence for a rule and had a requirement that the rule must apply to 50 or more tuples out of the dataset. Rules were generated for each flexible attribute alone, and the social media features alone and in combinations. This work compares the results of different flexible attributes alone to explore which have the greatest impact on the confidence and support of the resulting rules. Each set of rules contains an average of 1700 action rules. At least one rule applies to approximately 97% of the elements at that price level. This is termed the coverage of the rule set and can be used as a measure of its applicability. However, the confidence in these rules is quite low. As all the features discussed here have an extremely low cost, exploring low confidence rules is not a concern. To make the changes these rules suggest, an artist would only need to rewrite their artwork descriptions or change their profile. While some individual rules do have high confidence, the average confidence for the before attribute is approximately 13%.

In response to the low confidence of the initial sets of rules tested, additional rule sets were generated with greater constraints on the generation process. Rules were only generated to move each artwork up to the next higher price level. The social media flexible features (abbreviated in tables as SM) were used to generate rules, and the word counts (abbreviated in tables as WC) of both the artist's biography (abbreviated as Bio.) and description (abbreviated as Desc.) and text polarity features (abbreviated as Ps. for positive sentiment and Ng. for negative sentiment) were all used separately. The minimum support level for a rule was lowered from 50 to 2, but the minimum confidence for a rule to be considered was 60% for the before and after rules. The minimum number of attributes for the stable portion of the rule is one and the maximum number is five. The rule set generated here had a considerably lower average coverage at 9.375%, but a considerably higher average confidence for the before rules at 79.16% and the after rules at 79.2%. Considerably fewer rules were generated, and the average number of rules per group was 1,155. Table 1 displays the coverage of each rule set for the selected price levels and flexible attribute. This represents the percentage of objects at the lower price level that had at least one applicable rule to raise that object to the selected higher price level.

The level of support varies dramatically depending on the price level being addressed. The exact values of the prices are as follows: (<12.97–58.28), (58.28–95.90), (95.90–130.04), (130.04–188.13), (188.13–250), (250–350), (350–490), (490–742), (742–1351.24), (1351.24–>1,000,000). For example, the coverage of the rules sets that move artworks from the lowest prices, level one, to the next

Table 1. Coverage of rules generated using base stable features

	SM	Bio. WC	Desc. WC	Bio. Ps.	Desc. Ps.	Bio. Ng.	Desc. Ng.
Prices 1 -> 2	19.24	38.89	30.26	37.05	26.56	35.86	19.45
Prices 2 -> 3	4.31	16.35	13.03	17.04	11.97	9.70	5.04
Prices 3 -> 4	3.60	11.91	9.52	11.42	9.40	7.78	3.52
Prices 4 -> 5	2.51	7.87	5.80	8.02	5.67	4.41	2.35
Prices 5 -> 6	2.21	7.31	5.91	7.82	5.54	5.18	1.64
Prices 6 -> 7	1.30	8.20	6.91	7.96	5.77	3.41	3.54
Prices 7 -> 8	1.34	7.28	4.80	6.34	4.41	3.47	1.54
Prices 8 -> 9	1.94	7.79	6.59	8.91	6.03	5.14	2.05
Prices 9 -> 10	4.77	14.25	12.18	13.17	10.87	11.35	5.21

lowest price level, level two, ranges between 19.24% and 38.89% across all the different sets of flexible attributes. Notably, the social media attributes had markedly worse performance than the others. The attributes that changed an artwork’s description had less consistently high coverage than the attributes that addressed an artist’s biography. At higher values, coverage decreases. This may be due to the size of the shifts necessary to move prices from one tier to another at the higher levels.

To expand on the set of stable attributes, another set of rules was generated that added the dominant color of the work to the list of stable attributes. This set was generated using a randomly selected subset with approximately 100,000 tuples. Lisp-Miner’s discretization tools were used to create a new set of partitions similar to those used previously for this set. The same restrictions on rule generation were repeated. Rules must have a minimum confidence of 60% for the before and after rules and rules must have a minimum support of 10. Slightly more rules were generated per set than in the previous variation with an average number of rules per group of 1551. This set did have a slightly higher average coverage of 15.30%, and a similar average before confidence of 78.81% and an after confidence of 79.07%. As with the previous set of rules, the coverage shifts across different price levels. The coverage of each rule group changes depending on the selected flexible attributes and the selected price levels as demonstrated in Table 2.

Table 2. Coverage of rules generated using base stable features and main color

	SM	Bio. WC	Desc. WC	Bio. Ps.	Desc. Ps.	Bio. Ng.	Desc. Ng.
Prices 1 -> 2	25.79	52.14	34.97	52.43	31.83	47.50	24.31
Prices 2 -> 3	9.80	28.21	14.67	28.87	17.96	17.60	9.13
Prices 3 -> 4	7.51	21.34	15.15	20.55	13.29	13.12	7.48
Prices 4 -> 5	5.62	16.48	10.70	16.27	10.49	9.09	5.50
Prices 5 -> 6	3.73	14.87	9.22	14.77	8.31	9.61	3.96
Prices 6 -> 7	4.02	13.92	9.07	12.63	7.67	7.94	4.45
Prices 7 -> 8	3.22	13.47	7.89	13.64	7.26	6.97	3.79
Prices 8 -> 9	4.41	16.72	11.85	18.47	10.01	11.44	5.56
Prices 9 -> 10	12.52	28.39	18.84	27.51	18.43	20.73	10.74

6 Conclusions and Future Work

This work only begins to address the potential for the use of action rules to improve artist sales in the market for contemporary fine art. Many potential avenues for further research exist in the development of features for action rules.

In the initial set of results, the very high rate of coverage is promising. However, the low confidence level, while high for specific rules, is on average quite low. This issue may be attributable to allowing too many low confidence rules. It may also be due to the partitions of the feature sets being overly broad. However, low confidence for the set of G-Action rules is not a barrier to their being useful to an artist. These rules are extremely low cost and easily implemented. Stakeholder's may want to try them on even the small chance that the rules will provide an improvement. The second and third set of rules discussed demonstrates that is possible to significantly raise the average confidence in the rules, but at the expense of lowering the average coverage of the rule sets.

One strong avenue for potential future research is the development of more features for rule generation. The development of more flexible attributes, as well as exploring the attributes discussed here in combination has potential value for stakeholders in the art market. In addition to expanding the list of artwork features, artists have characteristics that may serve as additional stable attributes. A greater exploration of larger scale career changes an artist could make has great potential for research.

Acknowledgement. This research is supported by the National Science Foundation under grant IIP 1749105. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. The Hiscox Online Art Trade Report 2018. Technical report, ArtTactic (2018). <https://arttactic.com/product/hiscox-online-art-trade-report-2018/>
2. Beautiful Soup (2019). <https://www.crummy.com/software/BeautifulSoup/>
3. Color by name (2019). <http://colormine.org/colors-by-name>
4. Selenium (2019). <https://www.seleniumhq.org/>
5. Artfinder.com (2020). <https://www.artfinder.com/>
6. Aggarwal, C.C.: Machine Learning for Text. Springer, Cham (2018). <https://doi.org/10.1007/978-3-319-73531-3>
7. Bailey, J.: Machine Learning for Art Valuation. An Interview With Ahmed Hosny, December 2017. <https://www.artnome.com/news/2017/12/2/machine-learning-for-art-valuation>
8. Berlin, B., Kay, P.: Basic Color Terms: Their Universality and Evolution. University of California Press, Berkeley (1969)
9. Bird, S., Klein, E., Loper, E.: Natural Language Processing with Python, 1st edn. O'Reilly Media, Inc., Sebastopol (2009)
10. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **PAMI-8**(6), 679–698 (1986). <https://doi.org/10.1109/TPAMI.1986.4767851>. <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4767851>
11. Galbraith, J., Hodgson, D.: Econometric fine art valuation by combining hedonic and repeat-sales information. *Econometrics* **6**(3), 32 (2018). <https://doi.org/10.3390/econometrics6030032>. <http://www.mdpi.com/2225-1146/6/3/32>
12. Hajja, A.: Object-driven and temporal action rules mining (2013). <https://eric.ed.gov/?id=ED564978>
13. Hosny, A., Huang, J., Wang, Y.: The Green Canvas (2014). <http://ahmedhosny.github.io/theGreenCanvas/>
14. Hutto, C., Gilbert, E.: VADER: a parsimonious rule-based model for sentiment analysis of social media text. In: Proceedings of the 8th International Conference on Weblogs and Social Media, ICWSM 2014 (2015)
15. Kang, D., Shim, H., Yoon, K.: A method for extracting emotion using colors comprise the painting image. *Multimed. Tools Appl.* **77**(4), 4985–5002 (2017). <https://doi.org/10.1007/s11042-017-4667-0>
16. Labrecque, L.I., Milne, G.R.: Exciting red and competent blue: the importance of color in marketing. *J. Acad. Mark. Sci.* **40**(5), 711–727 (2012). <https://doi.org/10.1007/s11747-010-0245-y>
17. Lindsey, D.T., Brown, A.M.: Universality of color names. *Proc. Natl. Acad. Sci.* **103**(44), 16608–16613 (2006). <https://doi.org/10.1073/pnas.0607708103>. <http://www.pnas.org/cgi/doi/10.1073/pnas.0607708103>
18. Liu, D., Woodham, D.: Using AI to Predict Rothko Paintings' Auction Prices (2019). <https://www.artsy.net/article/artsy-editorial-ai-predict-mark-rothko-paintings-auction-prices>
19. Nekvapil, V.: Using the ac4ft-miner procedure in the medical domain [online] (2009). <https://theses.cz/id/0abafc/>. Accessed 03 Jan 2020
20. Pawlowski, C., Gelich, A., Raś, Z.W.: Can we build recommender system for artwork evaluation? In: Bembeník, R., Skonieczny, Ł., Protaziuk, G., Kryszkiewicz, M., Rybinski, H. (eds.) *Intelligent Methods and Big Data in Industrial Applications*. SBD, vol. 40, pp. 41–52. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-77604-0_4

21. Powell, L., Gelich, A., Ras, Z.W.: Developing artwork pricing models for online art sales using text analytics. In: Mihálydeák, T., et al. (eds.) IJCRS 2019. LNCS (LNAI), vol. 11499, pp. 480–494. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-22815-6_37
22. Powell, L., Gelich, A., Ras, Z.W.: Applying analytics to artist provided text to model prices of fine art. In: Appice, A., Ceci, M., Loglisci, C., Manco, G., Masciari, E., Ras, Z.W. (eds.) Complex Pattern Mining. SCI, vol. 880, pp. 189–211. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-36617-9_12
23. Powell, L., Gelich, A., Ras, Z.W.: Art innovation systems for value tagging. In: Encyclopedia of Organizational Knowledge, Administration, and Technologies. IGI Global (2020). <https://www.igi-global.com/book/encyclopedia-organizational-knowledge-administration-technology/242894>
24. Ras, Z.W., Wieczorkowska, A.: Action-rules: how to increase profit of a company. In: Zighed, D.A., Komorowski, J., Żytkow, J. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 587–592. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45372-5_70
25. Rauch, J., Šimůnek, M.: Action rules and the GUHA method: preliminary considerations and results. In: Rauch, J., Raś, Z.W., Berka, P., Elomaa, T. (eds.) ISMIS 2009. LNCS (LNAI), vol. 5722, pp. 76–87. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04125-9_11
26. Rauch, J., et al.: LISp-Miner, October 2019. <https://lispminer.vse.cz/index.html>
27. Rawlins, C., Johnson, P.: Selling on eBay: persuasive communication advice based on analysis of auction item descriptions. *J. Strateg. E-Commerce* **5**(1&2), 75–81 (2007)
28. Roberson, D., Hanley, J.: Color vision: color categories vary with language after all. *Curr. Biol.* **17**(15), R605–R607 (2007). <https://doi.org/10.1016/j.cub.2007.05.057>. <http://www.sciencedirect.com/science/article/pii/S0960982207014819>
29. Saleh, B., Elgammal, A.: Large-scale classification of fine-art paintings: learning the right metric on the right feature. [arXiv:1505.00855](https://arxiv.org/abs/1505.00855) [cs], May 2015. <http://arxiv.org/abs/1505.00855>
30. Sharma, G., Wu, W., Dalal, E.N.: The CIEDE2000 color-difference formula: implementation notes, supplementary test data, and mathematical observations. *Color Res. Appl.* **30**(1), 21–30 (2004). <https://doi.org/10.1002/col.20070>
31. Taylor, G.: Python-colormath (2014)
32. Team, O.: OpenCV, October 2017
33. Tzacheva, A.A., Bagavathi, A., Ayila, L.: Discovery of action rules at lowest cost in spark. In: 2017 IEEE International Conference on Data Mining Workshops (ICDMW), New Orleans, LA, pp. 87–94. IEEE, November 2017. <https://doi.org/10.1109/ICDMW.2017.173>. <http://ieeexplore.ieee.org/document/8215648/>
34. Tzacheva, A.A., Raś, Z.W.: Action rules mining. *Int. J. Intell. Syst.* **20**(7), 719–736 (2005). <https://doi.org/10.1002/int.20092>. <http://doi.wiley.com/10.1002/int.20092>
35. Velthuis, O.: Talking Prices: Symbolic Meanings of Prices on the Market for Contemporary Art. Princeton University Press (2005). <http://www.jstor.org/stable/j.ctt4cgd14>



Metric-Guided Multi-task Learning

Jinfu Ren, Yang Liu, and Jiming Liu^(✉)

Department of Computer Science, Hong Kong Baptist University, Kowloon Tong,
Hong Kong SAR, People's Republic of China
{jinfuren, csygliu, jiming}@comp.hkbu.edu.hk

Abstract. Multi-task learning (MTL) aims to solve multiple related learning tasks simultaneously so that the useful information in one specific task can be utilized by other tasks in order to improve the learning performance of all tasks. Many representative MTL methods have been proposed to characterize the relationship between different learning tasks. However, the existing methods have not explicitly quantified the distance or similarity of different tasks, which is actually of great importance in modeling the task relation for MTL. In this paper, we propose a novel method called Metric-guided MTL (M^2TL), which explicitly measures the task distance using a metric learning strategy. Specifically, we measure the distance between different tasks using their projection parameters and learn a distance metric accordingly, so that the similar tasks are close to each other while the uncorrelated tasks are faraway from each other, in terms of the learned distance metric. With a metric-guided regularizer incorporated in the proposed objective function, we open a new way to explore the related information among tasks. The proposed method can be efficiently solved via an alternative method. Experiments on both synthetic and real-world benchmark datasets demonstrate the superiority of the proposed method over existing MTL methods in terms of prediction accuracy.

Keywords: Multi-task learning · Task relation · Metric learning · Metric-guided multi-task learning

1 Introduction

Multi-task learning (MTL), inspired by human learning behavior and patterns of applying the knowledge and experience learned from some tasks to help learn others, solves multiple learning tasks simultaneously and improves the learning performance of all tasks by exploring the task similarities or commonalities [3, 20]. MTL has attracted extensive research interest and has been widely applied to various real-world problems such as human facial recognition and pose estimation [4], traffic flow forecasting [5], climate prediction [6], and disease modeling [14].

One of the key issues in MTL is to characterize the relationship between different learning tasks. Some representative methods have been proposed to

address this challenging issue. They can be categorized into two main categories [20]: feature-based MTL and parameter-based MTL. Feature-based MTL characterizes the relationships among different tasks by introducing constraints on the feature representations of given tasks to model the task similarity. According to the property of weight matrix, feature-based MTL can be further categorized into multi-task feature extraction [1, 22] and multi-task feature selection [13]. Parameter-based MTL models the relationships among tasks by manipulating the parameters in each task. It can be further divided into four sub-categories: low-rank methods that model the task-relation by constraining the rank of a parameter matrix [18]; task clustering methods that group similar tasks into subsets and share parameters within the cluster [8]; task-relation learning methods that describe the relationships among tasks using a specific criterion such as covariance or correlation [21]; and decomposition methods that characterize task relationships by decomposing the parameter matrix to a set of component matrices and introducing constraints on them [7].

Although the aforementioned methods have explored the task relationship from different aspects, they have not explicitly quantified the distance or similarity of different tasks, which is, of course, very important in modeling the intrinsic correlation of multiple learning tasks. To address this problem, in this paper, we introduce a new MTL method called Metric-guided MTL (M^2TL), aiming at explicitly measuring the task distance via a metric learning strategy. Specifically, we represent the distance between different learning tasks using their projection parameters. Accordingly, we propose to learn a distance metric, under which the similar learning tasks are close to each other while the uncorrelated ones are apart from each other. With the formulated distance metric, we introduce a metric-guided regularizer into the objective function of M^2TL . By jointly optimizing the loss function and the metric-guided regularizer, the learned task relationship is expected to well reflect the explicitly quantified similarity between different tasks.

The rest of the paper is organized as follows. Section 2 briefly reviews the related work on feature-based MTL. Section 3 introduces the proposed M^2TL , including the formulation of task distance, the objective function of M^2TL , the optimization procedure, and the computational complexity analysis. Section 4 validates the effectiveness of M^2TL on both synthetic and real-world benchmark datasets, demonstrating the superiority of the proposed method over existing MTL methods in terms of prediction accuracy. Section 5 draws the conclusion of the paper.

2 Related Work

The proposed M^2TL aims to learn the common feature transformation among different tasks. This section, therefore, briefly reviews some related work in feature-based MTL. Generally, the feature-based MTL approaches, including both feature extraction and feature selection methods, can be formulated under the following regularization framework:

$$\arg \min_{\mathbf{W}} Loss(\mathbf{W}) + \mu \mathcal{R}(\mathbf{W}), \quad (1)$$

where $\mathbf{W} \in \mathbb{R}^{d \times T}$ denotes the weight matrix, which is column stacked by each task’s weight vector \mathbf{w}_i ($i = 1, \dots, T$). Here d and T denote the dimension of features and the number of tasks, respectively. Moreover, $Loss(\mathbf{W})$ denotes the total loss on T learning tasks that we want to minimize, $\mathcal{R}(\mathbf{W})$ denotes the regularizer that characterize the relationship among different tasks, and $\mu \geq 0$ denotes the balancing parameter that adjusts the importance of $Loss(\mathbf{W})$ and that of $\mathcal{R}(\mathbf{W})$.

Different feature-based MTL approaches adopt their own ways to formulate the loss function $Loss(\mathbf{W})$ and the regularizer $\mathcal{R}(\mathbf{W})$, so that the relationship between different tasks can be modeled and the learning performance of all tasks can be optimized simultaneously. In [1], Argyriou et al. aimed to learn a square transformation matrix for features, which lies in the assumption that transformed feature space is more powerful than the original. They further proposed a convex formulation [2] to solve the optimization problem. In [11, 13], the $L_{2,1}$ -norm was used as the regularizer to select common features shared across different tasks. A more general form, the $L_{p,q}$ -norm, can also be utilized to select the common or shared features as it owns the property of sparsity as well [19]. However, the $L_{p,q}$ -norm may perform even worse when the value of shared features are highly uneven [9]. To address this issue, Jalali et al. proposed a method called dirty MTL in which the weight matrix is decomposed into two components, one to ensure block-structured row-sparsity and the other for element-wise sparsity with different regularizers imposed [9].

The regularizers adopted in the above methods have explored the relationship between different tasks from various perspectives. However, none of them has explicitly quantified the distance/similarity between different tasks, which is of vital importance in modeling the task correlation in MTL. This observation motivates us to propose a new MTL method, which can measure the distance between different tasks in an explicit way. Metric learning [16], which aims to learn a distance metric to reflect the intrinsic similarity/correlation between data samples, becomes a natural choice to model the task distance in our case. In recent years, metric learning has been widely used in various applications, such as healthcare [12], person re-identification [17], and instance segmentation [10]. Different from existing metric learning methods that work on data samples, the nature of MTL problem requires us to define the distance metric over tasks, so that the correlation between different tasks can be explicitly measured and integrated into the learning objectives.

3 Proposed Method

In this section, we introduce the proposed M²TL. First, we define the formulation of task distance metric. Based on that, we propose the objective function of M²TL and describe the optimization procedure. Finally, we analyze the computational complexity of the proposed method.

3.1 Task Distance Metric

For each task, the weight vector is a meaningful index to represent the information learned from the corresponding task. Therefore, we define a task distance metric based on weight vectors of different tasks:

$$D(t_i, t_j) = (\mathbf{w}_i - \mathbf{w}_j)^T \mathbf{M} (\mathbf{w}_i - \mathbf{w}_j), \quad i, j = 1, \dots, T, \quad (2)$$

where t_i and t_j denote the i -th task and the j -th task, respectively; \mathbf{w}_i and \mathbf{w}_j denote the weight vector learned for the i -th task and that for the j -th task, respectively; and $D(t_i, t_j)$ denotes the distance between task i and task j . Similar to the typical distance metric learning, we use the Mahalanobis matrix \mathbf{M} to flexibly adjust the importance of different dimensions. Note that if \mathbf{M} equals to the identity matrix, then the defined distance metric will reduce to the Euclidean distance.

3.2 Objective Function of M²TL

With the above definition on task distance metric, we can formulate the objective function of the proposed M²TL. Given T regression tasks and the corresponding datasets $\{(\mathbf{X}_1, \mathbf{y}_1), (\mathbf{X}_2, \mathbf{y}_2), \dots, (\mathbf{X}_T, \mathbf{y}_T)\}$, where $\mathbf{X}_i \in \mathbb{R}^{n_i \times d}$ and $\mathbf{y}_i \in \mathbb{R}^{n_i}$ ($i = 1, \dots, T$) denote the training samples and the corresponding labels in the i -th task respectively, and n_i denotes the number of samples in the i -th task. In this paper, we use regression tasks as an example to show the formulation of the proposed M²TL. In fact, the idea of the proposed method can be extended to other supervised/unsupervised learning tasks, such as classification tasks and clustering tasks, in a straightforward way. The proposed M²TL specifies the general formulation in Eq. (1) as follows:

$$\arg \min_{\mathbf{W}, \mathbf{M}} \text{Loss}(\mathbf{W}) + \frac{\mu}{T^2} \sum_{i=1}^T \sum_{j=1}^T D(t_i, t_j) + \|\mathbf{M} - \mathbf{Q}\|_F^2. \quad (3)$$

The first term in Eq. (3) represents the total loss of T tasks as mentioned previously. Without loss of generality, we utilize the least squares formulation as the loss function in our model: $\text{Loss}(\mathbf{W}) = \sum_{i=1}^T \frac{1}{2n_i} \|\mathbf{X}_i \mathbf{w}_i - \mathbf{y}_i\|_2^2$. Note that $\text{Loss}(\mathbf{W})$ in Eq. (3) can be any loss function according to different learning requirements. The second term in Eq. (3) is the task distance formulated in the previous subsection. By minimizing the summation of all task distances, we expect that the commonality/similarity among different tasks, which is generally hidden in the original feature space, can be extracted to the maximum extent. In addition to the loss function and the regularizer, we further introduce the

last term in Eq. (3) to avoid the trivial solution on \mathbf{M} by constraining it using a task correlation/covariance matrix \mathbf{Q} . Here \mathbf{Q} can be any correlation/covariance matrix that captures the relationship between different tasks. In our paper, we use the Pearson correlation coefficient matrix calculated by the initialization of \mathbf{W} because of its universality.

3.3 Optimization Procedure

To the best of our knowledge, there is no closed-form solution for the optimization problem in Eq. (3). Therefore, we use an alternating method to find the optimal \mathbf{M} and \mathbf{W} iteratively, which guarantees the optimality in each iteration as well as the local optimum of the solution. The detailed optimization procedure is described as follows:

Fix \mathbf{W} and update \mathbf{M} : With the fixed \mathbf{W} , the objective function in Eq. (3) can be rewritten as follows:

$$\begin{aligned}
 & \arg \min_{\mathbf{W}, \mathbf{M}} Loss(\mathbf{W}) + \frac{\mu}{T^2} \sum_{i=1}^T \sum_{j=1}^T D(t_i, t_j) + \|\mathbf{M} - \mathbf{Q}\|_F^2 \\
 = & \arg \min_{\mathbf{M}} \frac{\mu}{T^2} \sum_{i=1}^T \sum_{j=1}^T (\mathbf{w}_i - \mathbf{w}_j)^T \mathbf{M} (\mathbf{w}_i - \mathbf{w}_j) + \|\mathbf{M} - \mathbf{Q}\|_F^2 \\
 = & \arg \min_{\mathbf{M}} \frac{\mu}{T^2} tr(\mathbf{M} \sum_{i,j} (\mathbf{w}_i - \mathbf{w}_j)(\mathbf{w}_i - \mathbf{w}_j)^T) + \|\mathbf{M} - \mathbf{Q}\|_F^2 \tag{4} \\
 = & \arg \min_{\mathbf{M}} \frac{2\mu}{T^2} tr(\mathbf{M} (T \sum_{i=1}^T \mathbf{w}_i \mathbf{w}_i^T - \sum_{i,j} \mathbf{w}_i \mathbf{w}_j^T)) + \|\mathbf{M} - \mathbf{Q}\|_F^2 \\
 = & \arg \min_{\mathbf{M}} \frac{2\mu}{T^2} tr(\mathbf{M} \mathbf{W} \mathbf{L} \mathbf{W}^T) + \|\mathbf{M} - \mathbf{Q}\|_F^2,
 \end{aligned}$$

where \mathbf{L} is a $T \times T$ Lagrange matrix defined as: $\mathbf{L} = T\mathbf{I}_T - \mathbf{1}_T \mathbf{1}_T^T$, with \mathbf{I}_T and $\mathbf{1}_T$ being the T -dimensional identity matrix and all one column vector, respectively. Taking the derivative of the objective function in Eq. (4) with respect to \mathbf{M} and set it to zero, i.e.,

$$\frac{\partial \left[\frac{2\mu}{T^2} tr(\mathbf{M} \mathbf{W} \mathbf{L} \mathbf{W}^T) + \|\mathbf{M} - \mathbf{Q}\|_F^2 \right]}{\partial \mathbf{M}} = 0, \tag{5}$$

then we can obtain the update of \mathbf{M} :

$$\mathbf{M} = \mathbf{Q} - \frac{\mu}{T^2} \mathbf{W} \mathbf{L} \mathbf{W}^T. \tag{6}$$

Algorithm 1: Metric-guided Multi-Task Learning (M²TL)

1 Input: Training set for T learning tasks: $\{\mathbf{X}_i \in \mathbb{R}^{n_i \times d}, \mathbf{y}_i \in \mathbb{R}^{n_i}\}_{i=1}^T$
2 Output: Weight matrix $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_T] \in \mathbb{R}^{d \times T}$
1: **for** $i \leftarrow 1 : T$ **do**
2: Initialize \mathbf{w}_i : $\mathbf{w}_i \leftarrow \mathbf{X}_i^T \mathbf{y}_i$;
3: **end for**
4: **while** *not convergence* **do**
5: Update \mathbf{M} using Eq. (6);
6: **for** $i \leftarrow 1 : T$ **do**
7: Update \mathbf{w}_i using Eq. (9);
8: **end for**
9: **end while**

Fix \mathbf{M} and update \mathbf{W} : With the fixed \mathbf{M} , the objective function in Eq. (3) can be rewritten as follows:

$$\begin{aligned}
& \arg \min_{\mathbf{W}, \mathbf{M}} \text{Loss}(\mathbf{W}) + \frac{\mu}{T^2} \sum_{i=1}^T \sum_{j=1}^T D(t_i, t_j) + \|\mathbf{M} - \mathbf{Q}\|_F^2 \\
&= \arg \min_{\mathbf{W}} \text{Loss}(\mathbf{W}) + \frac{\mu}{T^2} \sum_{i=1}^T \sum_{j=1}^T (\mathbf{w}_i - \mathbf{w}_j)^T \mathbf{M} (\mathbf{w}_i - \mathbf{w}_j) \\
&= \arg \min_{\mathbf{W}} \sum_{i=1}^T \frac{1}{2n_i} \|\mathbf{X}_i \mathbf{w}_i - \mathbf{y}_i\|_2^2 + \frac{\mu}{T^2} \sum_{i=1}^T \sum_{j=1}^T (\mathbf{w}_i - \mathbf{w}_j)^T \mathbf{M} (\mathbf{w}_i - \mathbf{w}_j) \\
&= \arg \min_{\mathbf{W}} \sum_{i=1}^T \left(\frac{1}{2n_i} \|\mathbf{X}_i \mathbf{w}_i - \mathbf{y}_i\|_2^2 + \frac{\mu}{T^2} \sum_{j=1}^T (\mathbf{w}_i - \mathbf{w}_j)^T \mathbf{M} (\mathbf{w}_i - \mathbf{w}_j) \right).
\end{aligned} \tag{7}$$

Note that in the above formulation, each task can be updated individually. Therefore, for the i -th task, we fix the $1, \dots, i-1, i+1, \dots, T$ -th tasks, take the derivative with respect to \mathbf{w}_i , and set it to zero, then we have:

$$\frac{1}{n_i} \mathbf{X}_i^T (\mathbf{X}_i \mathbf{w}_i - \mathbf{y}_i) + \frac{4\mu}{T^2} \sum_{\substack{j=1 \\ j \neq i}}^T \mathbf{M} (\mathbf{w}_i - \mathbf{w}_j) = 0. \tag{8}$$

From Eq. (8), we can obtain the updating rule of \mathbf{w}_i :

$$\mathbf{w}_i = \left(\frac{1}{n_i} \mathbf{X}_i^T \mathbf{X}_i + \frac{4\mu(T-1)}{T} \mathbf{M} \right)^{-1} \left(\frac{1}{n_i} \mathbf{X}_i^T \mathbf{y}_i + \frac{4\mu}{T} \mathbf{M} \sum_{\substack{j=1 \\ j \neq i}}^T \mathbf{w}_j \right). \tag{9}$$

The details of the proposed M²TL are described in Algorithm 1.

3.4 Computational Complexity Analysis

In Algorithm 1, the most time-consuming steps are steps 4–9. The time complexity of step 5, i.e., updating \mathbf{M} , is $O(dT^2)$. The time complexity of step 7,

i.e., updating \mathbf{w}_i , is $O(d^3)$. Assume that t is the number of iterations in the outside *while* loop for convergence, then the total computational complexity of the proposed M²TL is $O(t(dT^2 + d^3T))$.

4 Experimental Results

In this section, we validate the performance of the proposed method on both synthetic and real-world datasets. We use two standard criteria in MTL for performance evaluation: the root mean squared error (RMSE) [23] and the normalized mean squared error (NMSE), which are defined as follows:

$$RMSE = \frac{\sum_{i=1}^m \|\mathbf{X}_i^T \mathbf{w}_i - \mathbf{y}_i\|_2 \times n_i}{\sum_{i=1}^m n_i}, \quad NMSE = \frac{\sum_{i=1}^m MSE_i / \text{var}(\mathbf{y}_i) \times n_i}{\sum_{i=1}^m n_i}, \quad (10)$$

where n_i is the number of samples in i -th task, and $MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \mathbf{x}_i \mathbf{w}_i)^2$ denotes the mean square error. We select the following baselines for performance comparison:

- **STL** [15]: the classical single-task learning method, which learns each task independently without modeling the task relationship. Here we employ Lasso as the STL model.
- **L21** [13]: a typical multi-task feature selection method, which uses $L_{2,1}$ -norm to achieve the row sparsity of weight matrix.
- **DirtyMTL** [9]: a representative dirty multi-task learning method, which decomposes the weight matrix into two components and regularizes these two components separately to overcome the shortage of $L_{q,p}$ -norm.
- **MTFSSR** [19]: a state-of-the-art multi-task feature selection method with sparse regularization, which extends the $L_{1,2}$ -norm regularization to the multi-task setting for capturing common features and extracting task-specific features simultaneously.

4.1 Experiments on Synthetic Dataset

In this subsection, we examine the convergence and the prediction accuracy of the proposed method on a synthetic dataset. We generate the data of T regression tasks. Each task includes N data samples. The data of the i -th task are generated from the normal distribution $\mathcal{N}(i/10, 1)$. The ground truth of the weight matrix, $\mathbf{W}^{(truth)}$, is generated from $\mathcal{N}(1, 1)$. The labels are then generated by: $\mathbf{y}_i = \mathbf{X}_i \mathbf{w}_i^{(truth)} + \epsilon$ accordingly, where ϵ is the Gaussian noise generated from $\mathcal{N}(0, 0.1)$. In the experiments, we assume that $\mathbf{W}^{(truth)}$ is unknown and aim to learn it from the training sets $\{\mathbf{X}_i, \mathbf{y}_i\}$ ($i = 1, \dots, T$). We set $T = 10$, $d = 10$, and $N = 30$ in our experiments.

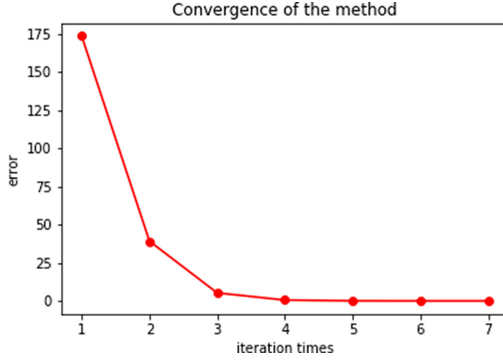


Fig. 1. Convergence speed of the proposed method on the synthetic dataset. The horizontal axis represents the number of iterations while the vertical axis represents the objective value of $\|\mathbf{W}^t - \mathbf{W}^{t-1}\|_F$. The red curve in the figure shows that M²TL can converge to a stable solution quickly. (Color figur online)

We first examine the convergence speed of the proposed method. We determine the stopping point by evaluating the difference between \mathbf{W}^t and \mathbf{W}^{t-1} . Specifically, we consider the algorithm as convergent and stop the iteration once the following inequality is satisfied: $\|\mathbf{W}^t - \mathbf{W}^{t-1}\|_F \leq \epsilon$. In this experiment, we set $\epsilon = 0.001$. Figure 1 shows the objective value of $\|\mathbf{W}^t - \mathbf{W}^{t-1}\|_F$ versus the number of iterations. The objective value decreases dramatically and satisfies the stopping criterion within only 7 iterations, demonstrating that the proposed method can converge to a stable solution quickly.

Table 1. The performance (in terms of RMSE) of STL [15], L21 [13], DirtyMTL [9], MTFSSR [19], and the proposed M²TL on the synthetic dataset, with three different training ratios. The best performances are highlighted in bold.

Training No.	Methods				
	STL	L21	DirtyMTL	MTFSSR	M ² TL
2	60.49 ± 12.45	31.07 ± 9.24	29.56 ± 4.64	38.90 ± 8.95	25.40 ± 12.26
4	53.36 ± 12.02	25.22 ± 3.23	25.69 ± 5.60	25.55 ± 7.24	24.93 ± 13.75
6	46.63 ± 19.95	18.70 ± 2.15	20.33 ± 5.53	18.56 ± 4.64	18.44 ± 8.39

In the second experiment, we compare the performance of the proposed method with that of four aforementioned baselines. We select 2, 4, and 6 samples from the 30 samples for training and use the rest for testing. We repeat the experiment for 10 times on randomly selected training samples and report the average RMSE as well as the standard deviation of each method. Table 1 lists the results of all methods. Obviously, with the exploration on the relationship between different tasks, the MTL methods (including L21, DirtyMTL, MTFSSR, and the proposed M²TL) achieve the lower RMSE than the STL method.

By further modeling the task distance in an explicit way, the proposed method outperforms other three MTL methods in all scenarios.

4.2 Experiments on Real-World Multi-task Datasets

In this subsection, we conduct experiments on two real-world multi-task datasets: the School dataset (<https://github.com/jiayuzhou/MALSAR/tree/master/data>) and the Sarcos dataset (<http://www.gaussianprocess.org/gpml/data/>). The School dataset is commonly used in many MTL literature. It is provided by the Inner London Education Authority and contains 15,362 data samples from 139 schools where each data sample has 27 attributes. We treat each school as a task and the learning target is to predict the exam score. The Sarcos dataset is about the inverse dynamic problem. The learning target is to predict the 7 joint torques given the 7 joint positions, 7 joint velocities and 7 joint accelerations. Here we treat prediction of one joint torque as a task so we have 7 tasks in total. For all 7 tasks, the 21 features (7 joint positions, 7 joint velocities and 7 joint accelerations) are used as the input, so the training data for different tasks are the same. We select 200 samples from each task to conduct our experiment. For both School and Sarcos datasets, we 20%, 30%, 40% of data for training and use the rest for testing. Similar to the synthetic experiments, we repeat the experiment for 10 times on randomly selected training samples. We report the average NMSE as well as the standard deviation of each method.

Tables 2 and 3 report the performance of all five methods on the School dataset and the Sarcos dataset, respectively. With the increase of training ratio, the NMSE of all methods decreases (except the L21 method from 20% to 30%), which is consistent with the common observation that providing more training data is generally beneficial to the learning task. Moreover, similar to the observations in the synthetic experiments, the MTL methods perform better than the STL method while our method again achieves the lowest prediction error among all five methods, demonstrating the necessity of exploring the task relationship in MTL and the effectiveness of the proposed formulation.

Table 2. The performance (in terms of NMSE) of STL [15], L21 [13], DirtyMTL [9], MTFSSR [19], and the proposed M²TL on the School dataset, with three different training ratios. The best performances are highlighted in bold.

Training ratio	Methods				
	STL	L21	DirtyMTL	MTFSSR	M ² TL
20%	2.118 ± 0.104	0.963 ± 0.005	0.924 ± 0.001	0.929 ± 0.000	0.931 ± 0.001
30%	2.052 ± 0.008	1.061 ± 0.007	0.858 ± 0.001	0.862 ± 0.000	0.855 ± 0.007
40%	2.013 ± 0.001	1.016 ± 0.001	0.821 ± 0.003	0.820 ± 0.000	0.809 ± 0.003

Table 3. The performance (in terms of NMSE) of STL [15], L21 [13], DirtyMTL [9], MTFSSR [19], and the proposed M²TL on the Sarcos dataset, with three different training ratios. The best performances are highlighted in bold.

Training ratio	Methods				
	STL	L21	DirtyMTF	MTFSSR	M ² TL
20%	2.180 ± 0.037	0.309 ± 0.113	0.294 ± 0.001	0.300 ± 0.004	0.291 ± 0.004
30%	2.052 ± 0.026	0.216 ± 0.001	0.226 ± 0.001	0.216 ± 0.002	0.211 ± 0.002
40%	2.000 ± 0.091	0.208 ± 0.001	0.202 ± 0.000	0.194 ± 0.000	0.189 ± 0.001

5 Conclusions

In this paper, we proposed a novel multi-task learning method called Metric-guided Multi-Task Learning (M²TL), which learns a task distance metric to explicitly measure the distance between different tasks and uses the learned metric as a regularizer to model the multi-task correlation. In the future, we plan to extend our experimental evaluation on large-scale datasets with distributed strategy in order to overcome the issues of quadratic time complexity with respect to the task number and cubic time complexity with respect to the feature dimension. Moreover, we will investigate more sophisticated ways to model and learn the task distance metric.

Acknowledgement. The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. This work was supported by the Grants from the Research Grant Council of Hong Kong SAR under Projects RGC/HKBU12201318 and RGC/HKBU12201619.

References

1. Argyriou, A., Evgeniou, T., Pontil, M.: Multi-task feature learning. In: Proceedings 19th NIPS, pp. 41–48 (2007)
2. Argyriou, A., Evgeniou, T., Pontil, M.: Convex multi-task feature learning. *Mach. Learn.* **73**(3), 243–272 (2008)
3. Caruana, R.: Multitask learning. *Mach. Learn.* **28**(1), 41–75 (1997)
4. Chu, X., Ouyang, W., Yang, W., Wang, X.: Multi-task recurrent neural network for immediacy prediction. In: Proceedings 15th ICCV, pp. 3352–3360 (2015)
5. Deng, D., Shahabi, C., Demiryurek, U., Zhu, L.: Situation aware multi-task learning for traffic prediction. In: Proceedings 17th ICDM, pp. 81–90 (2017)
6. Goncalves, A., Banerjee, A., Zuben, F.V.: Spatial projection of multiple climate variables using hierarchical multitask learning. In: Proceedings 31th AAAI, pp. 4509–4515 (2017)
7. Gong, P., Ye, J., Zhang, C.: Robust multi-task feature learning. In: Proceedings 18th SIGKDD, pp. 895–903 (2012)
8. Jacob, L., Vert, J.P., Bach, F.R.: Clustered multi-task learning: a convex formulation. In: Proceedings 21th NIPS, pp. 745–752 (2009)
9. Jalali, A., Sanghavi, S., Ruan, C., Ravikumar, P.K.: A dirty model for multi-task learning. In: Proceedings 22th NIPS, pp. 964–972 (2010)

10. Lahoud, J., Ghanem, B., Pollefeys, M., Oswald, M.R.: 3d instance segmentation via multi-task metric learning (2019). arXiv preprint [arXiv:1906.08650](https://arxiv.org/abs/1906.08650)
11. Liu, J., Ji, S., Ye, J.: Multi-task feature learning via efficient l_2, l_1 -norm minimization. In: Proceedings 25th UAI, pp. 339–348 (2009)
12. Ni, J., Liu, J., Zhang, C., Ye, D., Ma, Z.: Fine-grained patient similarity measuring using deep metric learning. In: Proceedings 26th CIKM, pp. 1189–1198. ACM (2017)
13. Obozinski, G., Taskar, B., Jordan, M.: Multi-task feature selection. University of California, Berkeley, Technical report (2006)
14. Pei, H., B. Yang, Liu, J., Dong, L.: Group sparse Bayesian learning for active surveillance on epidemic dynamics. In: Proceedings 32th AAAI, pp. 800–807 (2018)
15. Tibshirani, R.: Regression shrinkage and selection via the lasso. *J. Roy. Stat. Soc. Ser. B (Methodological)* **58**(1), 267–288 (1996)
16. Xing, E.P., Jordan, M.I., Russell, S.J., Ng, A.Y.: Distance metric learning with application to clustering with side-information. In: Proceedings 15th NIPS, pp. 521–528 (2003)
17. Yu, H.X., Wu, A., Zheng, W.S.: Cross-view asymmetric metric learning for unsupervised person re-identification. In: Proceedings 16th ICCV, pp. 994–1002 (2017)
18. Zhang, J., Ghahramani, Z., Yang, Y.: Learning multiple related tasks using latent independent component analysis. In: Proceedings 18th NIPS, pp. 1585–1592 (2006)
19. Zhang, J., Miao, J., Zhao, K., Tian, Y.: Multi-task feature selection with sparse regularization to extract common and task-specific features. *Neurocomputing* **340**, 76–89 (2019)
20. Zhang, Y., Yang, Q.: A survey on multi-task learning (2018). [arXiv:1707.08114v2](https://arxiv.org/abs/1707.08114v2)
21. Zhang, Y., Yeung, D.Y.: Multi-task boosting by exploiting task relationships. In: Flach, P.A., De Bie, T., Cristianini, N. (eds.) Proceedings 22th ECML PKDD, pp. 697–710 (2012)
22. Zhang, Z., Luo, P., Loy, C.C., Tang, X.: Facial landmark detection by deep multi-task learning. In: Proceedings 13th ECCV, pp. 94–108 (2014)
23. Zhou, J., Chen, J., Ye, J.: Malsar: multi-task learning via structural regularization. Arizona State University, Technical report (2011)



Sentiment Analysis with Contextual Embeddings and Self-attention

Katarzyna Biesialska¹(✉) , Magdalena Biesialska¹(✉) ,
and Henryk Rybinski²(✉) 

¹ Universitat Politècnica de Catalunya, Barcelona, Spain
{katarzyna.biesialska,magdalena.biesialska}@upc.edu

² Warsaw University of Technology, Warsaw, Poland
h.rybinski@ii.pw.edu.pl

Abstract. In natural language the intended meaning of a word or phrase is often implicit and depends on the context. In this work, we propose a simple yet effective method for sentiment analysis using contextual embeddings and a self-attention mechanism. The experimental results for three languages, including morphologically rich Polish and German, show that our model is comparable to or even outperforms state-of-the-art models. In all cases the superiority of models leveraging contextual embeddings is demonstrated. Finally, this work is intended as a step towards introducing a universal, multilingual sentiment classifier.

Keywords: Sentiment classification · Deep learning · Word embeddings

1 Introduction

All areas of human life are affected by people's views. With the sheer amount of reviews and other opinions over the Internet, there is a need for automating the process of extracting relevant information. For machines, however, measuring sentiment is not an easy task, because natural language is highly ambiguous at all levels, and thus difficult to process. For instance, a single word can hardly convey the whole meaning of a statement. Moreover, computers often do not distinguish literal from figurative meaning or incorrectly handle complex linguistic phenomena, such as: sarcasm, humor, negation etc.

In this paper, we take a closer look at two factors that make automatic opinion mining difficult – the problem of representing text information, and sentiment analysis (SA). In particular, we leverage contextual embeddings, which enable to convey a word meaning depending on the context it occurs in. Furthermore, we build a hierarchical multi-layer classifier model, based on an architecture of the Transformer encoder [32], primarily relying on a self-attention mechanism

K. Biesialska and M. Biesialska—Contributed equally to this work, which was mostly done at the Warsaw University of Technology.

and bi-attention. The proposed sentiment classification model is language independent, which is especially useful for low-resource languages (e.g. Polish).

We evaluate our methods on various standard datasets, which allows us to compare our approach against current state-of-the-art models for three languages: English, Polish and German. We show that our approach is comparable to the best performing sentiment classification models; and, importantly, in two cases yields significant improvements over the state of the art.

The paper is organized as follows: Sect. 2 presents the background and related work. Section 3 describes our proposed method. Section 4 discusses datasets, experimental setup, and results. Section 5 concludes this paper and outlines the future work.

2 Related Work

Sentiment classification has been one of the most active research areas in natural language processing (NLP) and has become one of the most popular downstream tasks to evaluate performance of neural network (NN) based models. The task itself encompasses several different opinion related tasks, hence it tackles many challenging NLP problems, see e.g. [16, 20].

2.1 Sentiment Analysis Approaches

The first fully-formed techniques for SA emerged around two decades ago, and continued to be prevalent for several years, until deep learning methods entered the stage. The most straight-forward method, developed in [30], is based on the number of positive and negative words in a piece of text. Concretely, the text is assumed to have positive polarity if it contains more positive than negative terms, and vice versa. Of course, the term-counting method is often insufficient; therefore, an improved method was proposed in [10], which combines counting positive and negative terms with a machine learning (ML) approach (i.e. Support Vector Machine).

Various studies (e.g. [31]) have shown that one can determine the polarity of an unknown word by calculating co-occurrence statistics of it. Moreover, classical solutions to the SA problem are often based on lexicons. Traditional lexicon-based SA leverages word-lists, that are pre-annotated with positive and negative sentiment. Therefore, for many years lexicon-based approaches have been utilized when there was insufficient amount of labeled data to train a classifier in a fully supervised way.

In general, ML algorithms are popular methods for determining sentiment polarity. A first ML model applied to SA has been implemented in [21]. Moreover, throughout the years, different variants of NN architectures have been introduced in the field of SA. Especially recursive neural networks [22], such as recurrent neural networks (RNN) [13, 28, 29], or convolutional neural networks (CNN) [9, 11] have become the most prevalent choices.

2.2 Vector Representations of Words

One of the principal concepts in linguistics states that related words can be used in similar ways [6]. Importantly, words may have different meaning in different contexts. Nevertheless, until recently it has been a dominant approach (e.g. word2vec [19], GloVe [23]) to learn representations such that each and every word has to capture all its possible meanings.

However, lately a new set of methods to learn dynamic representations of words has emerged [5, 7, 18, 24, 25]. These approaches allow each word representation to capture what a word means in a particular context. While every word token has its own vector, the vector can depend on a variable-length sequence of nearby words (i.e. context). Consequently, a context vector is obtained by feeding a neural network with these context word vectors and subsequently encoding them into a single fixed-length vector.

ULMFiT [7] was the very first method to induce contextual word representations by harnessing the power of language modeling. The authors proposed to learn contextual embeddings by pre-training a language model (LM), and then performing task-specific fine-tuning. ULMFiT architecture is based on a vanilla 3-layer Long Short-Term Memory (LSTM) NN without any attention mechanism.

The other contextual embedding model introduced recently is called ELMo (Embeddings from Language Models) [24]. Similarly to ULMFiT, this model uses tokens at the word-level. ELMo contextual embeddings are “deep” as they are a function of all hidden states. Concretely, context-sensitive features are extracted from a left-to-right and a right-to-left 2-layer bidirectional LSTM language models. Thus, the contextual representation of each word is the concatenation of the left-to-right and right-to-left representations as well as the initial embedding (see Fig. 1).

The most recent model – BERT [5] – is more sophisticated architecturally-wise, as it is a multi-layer masked LM based on the Transformer NN utilizing sub-word tokens. However, as we are bound to use word-level tokens in our sentiment classifier, we leverage the ELMo model for obtaining contextual embeddings. More specifically, by means of ELMo we are able to feed our classifier model with context-aware embeddings of an input sequence. Hence, in this setting we do not perform any fine-tuning of ELMo on a downstream task.

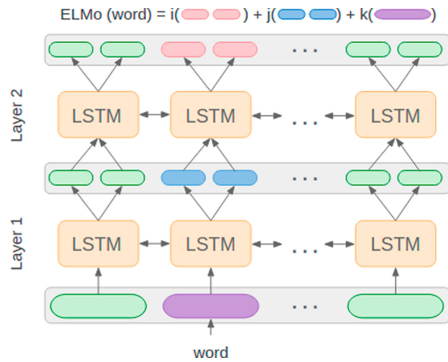


Fig. 1. The architecture of ELMo.

2.3 Self-attention Deep Neural Networks

The attention mechanism was introduced in [3] in 2014 and since then it has been applied successfully to different computer vision (e.g. visual explanation) and NLP (e.g. machine translation) tasks. The mechanism is often used as an extra source of information added on top of the CNN or LSTM model to enhance the extraction of sentence embedding [15, 26]. However, this scenario is not applicable to sentiment classification, since the model only receives a single sentence on input, hence there is no such extra information [15].

Self-attention (or intra-attention) is an attention mechanism that computes a representation of a sequence by relating different positions of a single sequence. Previous work on sentiment classification has not covered extensively attention-based neural network models for SA (especially using the Transformer architecture [32]), although some papers have appeared recently [2, 14].

3 The Proposed Approach

Our proposed model, called Transformer-based Sentiment Analysis (TSA) (see Fig. 2), is based on the recently introduced Transformer architecture [32], which has provided significant improvements for the neural machine translation task. Unlike RNN or CNN based models, the Transformer is able to learn dependencies between distant positions. Therefore, in this paper we show that attention-based models are suitable for other NLP tasks, such as learning distributed representations and SA, and thus are able to improve the overall accuracy.

The architecture of the TSA model and steps to train it can be summarized as follows:

- (a) At the very beginning there is a simple text pre-processing method that performs text clean-up and splits text into tokens.
- (b) We use contextual word representations to represent text as real-valued vectors.
- (c) After embedding the text into real-valued vectors, the Transformer network maps the input sequence into hidden states using self-attention.
- (d) Next a bi-attention mechanism is utilized to estimate the interdependency between representations.
- (e) A single layer LSTM together with self-attentive pooling compute the pooled representations.
- (f) A joint representation for the inputs is later passed to a fully-connected neural network.
- (g) Finally, a softmax layer is used to determine sentiment of the text.

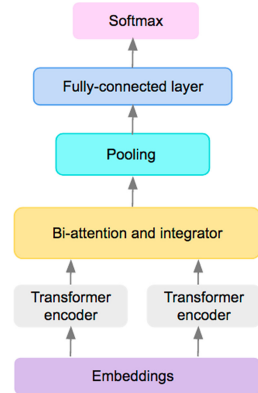


Fig. 2. An overview of the TSA model architecture.

3.1 Embeddings and Encoded Positional Information

Non-recurrent models, such as deep self-attention NN, do not necessarily process the input sequence in a sequential manner. Hence, there is no way they can record the position of each word in a sequence, which is an inherent limitation of every such model. Therefore, in the case of the Transformer, the need has been addressed in the following manner – the Transformer takes into account the order of the words in the input sequence by encoding their position information in extra vectors (so called positional encoding vectors) and adding them to input embeddings. There are many different approaches to embed position information, such as learned or fixed positional encodings (PE), or recently introduced relative position representations (RPR) [27]. The original Transformer used sine and cosine functions of different frequencies.

In this work, we explore the effectiveness of applying a modified attention function to what was proposed in [32]. In the vanilla Transformer a dot-product (multiplicative) attention is used with the scaling factor of $\sqrt{d_z}$. We also propose a different approach to incorporate positional information into the model, namely using RPR instead of PE. Furthermore, we use global average pooling in order to average the output of the last self-attention layer and prepare the model for the final classification layer.

3.2 The Transformer Encoder

The input sequence is combined with word and positional embeddings, which provide time signal, and together are fed into an encoder block. Matrices for a query Q , a key K and a value V are calculated and passed to a self-attention layer. Next, a normalization is applied and residual connections provide additional context. Further, a final dense layer with vocabulary size generates the output of the encoder. A fully-connected feed-forward network within the model is a single hidden layer network with a ReLU activation function in between:

$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2 \quad (1)$$

3.3 Self-attention Layer

The self-attention block in the encoder is called multi-head self-attention. A self-attention layer allows each position in the encoder to access all positions in the previous layer of the encoder immediately, and in the first layer all positions in the input sequence. The multi-head self-attention layer employs h parallel self-attention layers, called heads, with different Q , K , V matrices obtained for each head. In a nutshell, the attention mechanism in the Transformer architecture relies on a scaled dot-product attention, which is a function of Q and a set of K - V pairs. The computation of attention is performed in the following order. First, a multiplication of a query and transposed key is scaled through the scaling factor of $1/\sqrt{d_z}$ (Eq. 2)

$$m_{ij} = \frac{QK^T}{\sqrt{d_z}} \quad (2)$$

Next, the attention is produced using the softmax function over their scaled inner product:

$$\alpha_{ij} = \frac{e^{m_{ij}}}{\sum_{k=1}^n e^{m_{ik}}} \quad (3)$$

Finally, the weighted sum of each attention head and a value is calculated as follows:

$$z_i = \sum_{j=1}^n \alpha_{ij} V \quad (4)$$

3.4 Masking and Pooling

Similar to other sources of data, the datasets used for training and evaluation of our models contain sequences of different length. The most common approach in the literature involves finding a maximal sequence length existing in the dataset/batch and padding sentences that are shorter than the longest one with trailing zeroes. In the proposed TSA model, we deal with the problem of variable-length sequences by using masking and self-attentive pooling. The inspiration for our approach comes from the BCN model proposed in [18]. Thanks to this mechanism, we are able to fit sequences of different length into the final fixed-size vector, which is required for the computation of the sentiment score. The self-attentive pooling layer is applied just after the encoder block.

4 Experiments

4.1 Datasets

In this work, we compare sentiment analysis results considering four benchmark datasets in three languages. All datasets are originally split into training, dev and test sets. Below we describe these datasets in more detail (Table 1).

Table 1. Sentiment analysis datasets with number of classes and train/dev/test split.

Dataset	# Classes	Train	Dev	Test	Domain	Language
SST-2	2	6,920	872	1,821	Movies	English
SST-5	5	8,544	1,101	2,210	Movies	English
PolEmo 2.0-IN	5	5,783	723	722	Medical, hotels	Polish
GermEval	3	19,432	2,369	2,566	Travel, transport	German

Stanford Sentiment Treebank (SST). This collection of movie reviews [28] from the `rottentomatoes.com` is annotated for the binary (SST-2) and fine-grained (SST-5) sentiment classification. SST-2 divides reviews into two groups: *positive* and *negative*, while SST-5 distinguishes 5 different review types: *very*

positive, positive, neutral, negative, very negative. The dataset consists of 11,855 single sentences and is widely used in the NLP community.

PolEmo 2.0. The dataset [12] comprises online reviews from education, medicine and hotel domains. There are two separate test sets, to allow for in-domain (medicine and hotels) and out-of-domain (products and university) evaluation. The dataset comes with the following sentiment labels: *strong positive, weak positive, neutral, weak negative, strong negative, and ambiguous*.

GermEval. This dataset [33] contains customer reviews of the railway operator (Deutsche Bahn) published on social media and various web pages. Customers expressed their feedback regarding the service of the railway company (e.g. travel experience, timetables, etc.) by rating it as *positive, negative, or neutral*.

4.2 Experimental Setup

Pre-processing of input datasets is kept to a minimum as we perform only tokenization when required. Furthermore, even though some datasets, such as SST or GermEval, provide additional information (i.e. phrase, word or aspect-level annotations), for each review we only extract text of the review and its corresponding rating.

The model is implemented in the Python programming language, PyTorch¹ and AllenNLP². Moreover, we use pre-trained word-embeddings, such as ELMo [24], GloVe [23]. Specifically, we use the following ELMo models: Original³, Polish [8] and German [17]. In the ELMO+GloVe+BCN model we use the following 300-dimension GloVe embeddings: English⁴, Polish [4] and German⁵. In order to simplify our approach when training the sentiment classifier model, we establish a very similar setting to the vanilla Transformer. We use the same optimizer - Adam with $\beta_1 = 0.9, \beta_2 = 0.98$, and $\epsilon = 10^{-9}$. We incorporate four types of regularization during training: dropout probability $P_{drop} = 0.1$, embedding dropout probability $P_{emb} = 0.5$, residual dropout probability $P_{res} = 0.2$, and attention dropout probability $P_{attn} = 0.1$. We use 2 encoder layers. In addition, we employ label smoothing of value $\epsilon_{ls} = 0.1$. In terms of RPR parameters, we set clipping distance to $k = 10$.

4.3 Results and Discussion

In Table 2, we summarize experimental results achieved by our model and other state-of-the-art systems reported in the literature by their respective authors.

We observe that our models, baseline and ELMo+TSA, outperform state-of-the-art systems for all three languages. More importantly, the presented accuracy

¹ <https://pytorch.org>.

² <https://allennlp.org>.

³ <https://allennlp.org/elmo>.

⁴ <http://nlp.stanford.edu/data/glove.840B.300d.zip>.

⁵ <https://wikipedia2vec.github.io/wikipedia2vec/pretrained>.

Table 2. Results of our systems compared to baselines and state-of-the-art systems evaluated on English, Polish and German sentiment classification datasets.

	English		Polish	German
	SST-2	SST-5	PolEmo2.0-IN	GermEval
RNTN [28]	85.4	45.7	–	–
DCNN [9]	86.8	48.5	–	–
CNN [11]	88.1	48.0	–	–
DMN [13]	88.6	52.1	–	–
Constituency Tree-LSTM [29]	88.0	51.0	–	–
CoVe+BCN [18]	90.3	53.7	–	–
SSAN+RPR [2]	84.2	48.1	–	–
Polish BERT [1]	–	–	88.1	–
SWN2-RNN [33]	–	–	–	74.9
<i>Our baseline</i>				
ELMo+GloVe+BCN	91.4	53.5	88.9	78.2
<i>Our model</i>				
ELMo+TSA	89.3	50.6	89.8	78.9

scores indicate that the TSA model is competitive and for two languages (Polish and German) achieves the best results. Also noteworthy, in Table 2, there are two models that use some variant of the Transformer: SSAN+RPR [2] uses the Transformer encoder for the classifier, while Polish BERT [1] employs Transformer-based language model introduced in [5]. One of the reasons why we achieve higher score for the SST dataset might be that the authors of SSAN+RPR used word2vec embeddings [19], whereas we employ ELMo contextual embeddings [24]. Moreover, in our TSA model we use not only self-attention (as in SSAN+RPR) but also a bi-attention mechanism, hence this also should provide performance gains over standard architectures.

In conclusion, comparing the results of the models leveraging contextual embeddings (CoVe+BCN, Polish BERT, ELMo+GloVe+BCN and ELMo+TSA) with the rest of the reported models, which use traditional distributional word vectors, we note that the former category of sentiment classification systems demonstrates remarkably better results.

5 Conclusion and Future Work

We have presented a novel architecture, based on the Transformer encoder with relative position representations. Unlike existing models, this work proposes a model relying solely on a self-attention mechanism and bi-attention. We show that our sentiment classifier model achieves very good results, comparable to the state of the art, even though it is language-agnostic. Hence, this work is a step towards building a universal, multi-lingual sentiment classifier.

In the future, we plan to evaluate our model using benchmarks also for other languages. It is particularly interesting to analyze the behavior of our model with respect to low-resource languages. Finally, other promising research avenues worth exploring are related to unsupervised cross-lingual sentiment analysis.

References

1. Allegro: Klej benchmark. <https://klejbenchmark.com/>. Accessed 20 Jan 2020
2. Ambarsoumian, A., Popowich, F.: Self-attention: a better building block for sentiment analysis neural network classifiers. In: Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pp. 130–139 (2018)
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv (2014)
4. Dadas, S.: A repository of polish NLP resources. Github (2019). <https://github.com/sdadas/polish-nlp-resources/>. Accessed 20 Jan 2020
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 4171–4186 (2019)
6. Firth, J.R.: A synopsis of linguistic theory, 1930–1955. Studies in linguistic analysis (1957)
7. Howard, J., Ruder, S.: Universal language model fine-tuning for text classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, pp. 328–339 (2018)
8. Janz, A., Miłkowski, P.: ELMo embeddings for polish (2019). <http://hdl.handle.net/11321/690>. CLARIN-PL digital repository
9. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pp. 655–665 (2014)
10. Kennedy, A., Inkpen, D.: Sentiment classification of movie reviews using contextual valence shifters. *Comput. Intell.* **22**, 110–125 (2006)
11. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746–1751 (2014)
12. Kocoń, J., Miłkowski, P., Zaśko-Zielińska, M.: Multi-level sentiment analysis of PolEmo 2.0: extended corpus of multi-domain consumer reviews. In: Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL), pp. 980–991 (2019)
13. Kumar, A., et al.: Ask me anything: dynamic memory networks for natural language processing. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning, vol. 48, pp. 1378–1387 (2016)
14. Letarte, G., Paradis, F., Giguère, P., Laviolette, F.: Importance of self-attention for sentiment analysis. In: Proceedings of the 2018 EMNLP Workshop BlackboxNLP, pp. 267–275 (2018)
15. Lin, Z., et al.: A structured self-attentive sentence embedding. In: 5th International Conference on Learning Representations (2017)
16. Liu, B.: Sentiment analysis and opinion mining. *Synth. Lect. Hum. Lang. Technol.* **5**, 1–167 (2012). Morgan & Claypool Publishers

17. May, P.: German ELMo Model (2019). <https://github.com/t-systems-on-site-services-gmbh/german-elmo-model>. Accessed 20 Jan 2020
18. McCann, B., Bradbury, J., Xiong, C., Socher, R.: Learned in translation: contextualized word vectors. *Adv. Neural Inf. Process. Syst.* **30**, 6294–6305 (2017)
19. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. *Adv. Neural Inf. Process. Syst.* **26**, 3111–3119 (2013)
20. Mohammad, S.M.: Sentiment analysis: detecting valence, emotions, and other affectual states from text. In: *Emotion Measurement*, pp. 201–237. Elsevier (2016)
21. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? sentiment classification using machine learning techniques. In: *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 79–86 (2002)
22. Paulus, R., Socher, R., Manning, C.D.: Global belief recursive neural networks. *Adv. Neural Inf. Process. Syst.* **27**, 2888–2896 (2014)
23. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543 (2014)
24. Peters, M., et al.: Deep contextualized word representations. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2227–2237 (2018)
25. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018)
26. dos Santos, C.N., Tan, M., Xiang, B., Zhou, B.: Attentive pooling networks. *arXiv* (2016)
27. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 464–468 (2018)
28. Socher, R., et al.: Recursive deep models for semantic compositionality over a sentiment treebank. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642 (2013)
29. Tai, K.S., Socher, R., Manning, C.D.: Improved semantic representations from tree-structured long short-term memory networks. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pp. 1556–1566 (2015)
30. Turney, P.: Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 417–424 (2002)
31. Turney, P.D., Pantel, P.: From frequency to meaning: vector space models of semantics. *J. Artif. Intell. Res.* **37**, 141–188 (2010)
32. Vaswani, A., et al.: Attention is all you need. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pp. 5998–6008 (2017)
33. Wojatzki, M., Ruppert, E., Holschneider, S., Zesch, T., Biemann, C.: GermEval 2017: shared task on aspect-based sentiment in social media customer feedback. *Proc. GermEval 2017*, 1–12 (2017)

Nominees for Best Student Paper Award



Interpretable Segmentation of Medical Free-Text Records Based on Word Embeddings

Adam Gabriel Dobrakowski¹(✉), Agnieszka Mykowiecka²,
Małgorzata Marciniak², Wojciech Jaworski¹, and Przemysław Biecek¹

¹ University of Warsaw, Banacha 2, Warsaw, Poland
ad359226@students.mimuw.edu.pl,

{W.Jaworski,P.Biecek}@mimuw.edu.pl

² Institute of Computer Science Polish Academy of Sciences,
Jana Kazimierza 5, Warsaw, Poland
{agn,mm}@ipipan.waw.pl

Abstract. Is it true that patients with similar conditions get similar diagnoses? In this paper we present a natural language processing (NLP) method that can be used to validate this claim. We (1) introduce a method for representation of medical visits based on free-text descriptions recorded by doctors, (2) introduce a new method for segmentation of patients' visits, (3) present an application of the proposed method on a corpus of 100,000 medical visits and (4) show tools for interpretation and exploration of derived knowledge representation. With the proposed method we obtained stable and separated segments of visits which were positively validated against medical diagnoses. We show how the presented algorithm may be used to aid doctors in their practice.

1 Introduction

Processing of free-text clinical records plays an important role in computer-supported medicine [1, 13]. A detailed description of symptoms, examination and an interview is often stored in an unstructured way as free-text, hard to process but rich in important information. Although there exist some attempts to process medical notes for English and some other languages, in general, the problem is still challenging [23]. The most straightforward approach to the processing of clinical notes could be their clustering with respect to different features like diagnosis or type of treatment. The process can either concentrate on patients or on their particular visits.

Grouping of visits can fulfil many potential goals. If we are able to group visits into clusters based on interview with a patient and medical examination then we can: follow recommendations that were suggested to patients with similar history to create a list of possible diagnoses; reveal that the current diagnosis is unusual; identify subsets of visits with the same diagnosis but different symptoms. A desired goal in the patients' segmentation is to divide them into groups with

similar properties. In the case of segmentation hospitalized patients one of the most well-known examples are Diagnosis Related Groups [11] which aim to divide patients into groups with similar costs of treatment. Grouping visits of patients in health centers is a different issue. Here most of the information is unstructured and included in the visit’s description written by a doctor: the description of the interview with the patient and the description of a medical examination of the patient.

Segmentation (clustering) is a well studied task for structured data such as age, sex, place, history of diseases, ICD-10 code etc. (an example of patients segmentation based only on their history of diseases is introduced in [26]), but it is far from being solved for unstructured free-texts which requires undertaking many decision on how the text and its meaning is to be represented. Medical concepts to be extracted from texts very often are taken from Unified Medical Language System (UMLS, see [4]), which is a commonly accepted base of biomedical terminology. Representations of medical concepts are computed based on various medical texts, like medical journals, books, etc. [5, 8, 10, 21, 22] or based directly on data from Electronic Health Records [6–8]. Other approach for patient segmentation is given in [6]. A subset of medical concepts (e.g. diagnosis, medication, procedures) and embeddings is aggregated for all visits of a patient. This way we get patient embedding that summaries patient medical history.

In this work we present a different approach. Our data include medical records for the medical history, description of the examination and recommendations for the treatment. Complementary sources allow us to create a more comprehensive visit description. The second difference is grouping visits, not patients. In this way a single patient can belong to several clusters. Our segmentation is based on a dictionary of medical concepts created from data, as for Polish does not exist any classification of medical concepts like UMLS or SNOMED. Obtained segments are supplemented with several approaches to visual exploration that facilitate interpretation of segments. Some examples of visual exploration of supervised models for structured medical data are presented in [3, 14, 17]. In this article we deal with a problem of explainable machine learning for unsupervised models.

2 Corpus of Free-Text Clinical Records

The clustering method is developed and validated on a dataset of free-text clinical records of about 100,000 visits. The data set consists of descriptions of patients’ visits from different primary health care centers and specialist clinics in Poland. They have a free-text form and are written by doctors representing a wide range of medical professions, e.g. general practitioners, dermatologists, cardiologists or psychiatrists. Each description is divided into three parts: interview, examination, and recommendations.

3 Methodology

In this section we describe our algorithm for visits clustering. The process is performed in the following four steps: (1) Medical concepts are extracted from free-text descriptions of an interview and examination. (2) A new representation of identified concepts is derived with concepts embedding. (3) Concept embeddings are transformed into visit embeddings. (4) Clustering is performed on visit embeddings.

3.1 Extraction of Medical Concepts

As there are no generally available terminological resources for Polish medical texts, the first step of data processing was aimed at automatic identification of the most frequently used words and phrases. The doctors' notes are usually rather short and concise, so we assumed that all frequently appearing phrases are domain related and important for text understanding. The notes are built mostly from noun phrases which consist of a noun optionally modified by a sequence of adjectives or by another noun in the genitive. We only extracted sequences that can be interpreted as phrases in Polish.

To get the most common phrases, we processed 220,000 visits' descriptions. First, we preprocessed texts using Concraft tagger [28] which assigns lemmas, POS and morphological features values. It also guesses descriptions (apart from lemmas) for words which are not present in its vocabulary. Phrase extraction and ordering was performed by TermoPL [19]. The program allows for defining a grammar describing extracted text fragments and order them according to a version of the C-value coefficient [12], but we used the built-in grammar of noun phrases. The first 4800 phrases (all with C-value equal at least 20) from the obtained list were manually annotated with semantic labels. The list of 137 labels covered most general concepts like *anatomy*, *feature*, *disease*, *test*. Many labels were assigned to multi-word expressions (MWEs). In some cases phrases were also labeled separately, e.g. *left hand* is labeled as *anatomy* while *hand* is also labeled as *anatomy* and *left* as *lateralization*. The additional source of information was the list of 9993 names of medicines and dietary supplements.

The list of terms together with their semantic labels was then converted to the format of lexical resources of Categorical Syntactic-Semantic Parser "ENIAM" [15, 16]. The parser recognized lexemes and MWEs in texts according to the provided list of terms, then the longest sequence of recognized terms was selected, and semantic representation was created. Semantic representation of a visit has a form of a set of pairs composed of recognized terms and their labels (not recognized tokens were omitted). The average coverage of semantic representation was 82.06% of tokens and 75.38% of symbols in section *Interview* and 87.43% of tokens and 79.28% of symbols in section *Examination*.

Texts of visits are heterogeneous as they consist of: very frequent domain phrases; domain important words which are too infrequent to be at the top of the term list prepared by TermoPL; some general words which do not carry relevant information; numerical information; and words which are misspelled. In

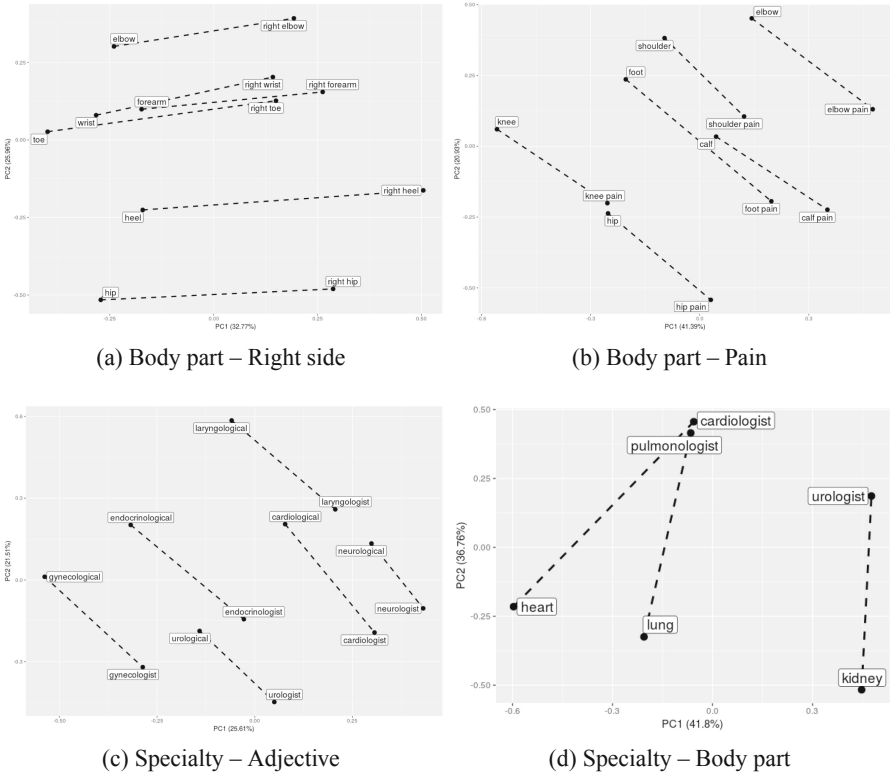


Fig. 1. Visualization of analogies between terms. The pictures show term embeddings projected into 2d-plane using PCA. Each panel shows a different type of analogy.

the clustering task we neglect the original text with inflected word forms and the experiments are solely performed on the set of semantic labels attached to each interview and examination.

3.2 Embeddings for Medical Concepts

Operating on relatively large amount of very specific texts, we decided not to use any general model for Polish. In the experiments, we reduce the description of visits to extracted concepts and train on them our own domain embeddings. During creating the term co-occurrence matrix the whole visit’s description is treated as the neighbourhood of the concept. Furthermore we choose only unique concepts and abandon their original order in the description (we follow this way due to simplicity).

We compute embeddings of concepts by GloVe [24] for interview descriptions and for examination descriptions separately. Computing two separate embeddings we aim at catching the similarity between terms in their specific context. For example, the nearest words to *cough* in the interview descriptions is *runny nose*, *sore throat*, *fever* but in the examination description it is *rash*, *sunny*, *laryngeal*.

3.3 Visit Embeddings

The simplest way to generate text embeddings based on term embeddings is to use some kind of aggregation of term embeddings such as an average. This approach was tested for example in [2] and [7]. In [9] the authors computed a weighted mean of term embeddings by the construction of a loss function and training weights by the gradient descent method. Thus, in our method we firstly compute embeddings of the descriptions (for interview and examination separately) as a simple average of concepts' embeddings. Then, the final embeddings for visits are obtained by concatenation of two descriptions' embeddings.

3.4 Visits Clustering

Based on Euclidean distance between vector representations of visits we applied and compared two clustering algorithms: k-means and hierarchical clustering with Ward's method for merging clusters [27]. The similarity of these clusterings was measured by the adjusted Rand index [25]. For the final results we chose the hierarchical clustering algorithm due to greater stability.

Table 1. The statistics of clusters for selected domains. The last column shows adjusted Rand index between k-means and hierarchical clustering.

Domain	# clusters	# visits	Clusters' size	K-means - hclust
Cardiology	6	1201	428, 193, 134, 303, 27, 116	0.87
Family medicine	6	11230	3108, 2353, 601, 4518, 255, 395	0.69
Gynecology	4	3456	1311, 1318, 384, 443	0.8
Internal medicine	5	6419	1915, 1173, 1930, 1146, 255	0.76
Psychiatry	5	1012	441, 184, 179, 133, 75	0.81

Table 2. The categories of questions in term analogy task with example pairs.

Type of relationship	# Pairs	Term pair 1		Term pair 2	
Body part – Pain	22	Eye	Eye pain	Foot	Foot pain
Specialty – Adjective	7	Dermatologist	Dermatological	Neurologist	Neurological
Body part – Right side	34	Hand	Right hand	Knee	Right knee
Body part – Left side	32	Thumb	Left thumb	Heel	Left heel
Spec. – Consultation	11	Surgeon	Surgical consult.	Gynecologist	g. consult.
Specialty – Body part	9	Cardiologist	Heart	Oculist	Eye
Man – Woman	9	Patient (male)	Patient (female)	Brother	Sister

Table 3. Mean accuracy of correct answers on term analogy tasks. Rows show different embeddings sizes, columns correspond to size of neighborhoods.

Dim./Context	1	3	5
10	0.1293	0.2189	0.2827
15	0.1701	0.3081	0.4123
20	0.1702	0.3749	0.4662
25	0.1667	0.4120	0.5220
30	0.1674	0.4675	0.5755
40	0.1460	0.5017	0.6070
50	0.1518	0.4966	0.6190
100	0.0435	0.4231	0.5483
200	0.0261	0.3058	0.4410

Table 4. The most common recommendations for each segment derived for gynecology. In brackets we present a percentage of visits in this cluster which contain a specified term. We skipped terms common in many clusters, like: *treatment, ultrasound treatment, control, morphology, hospital, lifestyle, zus (Social Insurance Institution)*.

Cluster	Size	Most frequent recommendations
1	1311	Recommendation (16.6%), general urine test (5.6%), diet (4.1%), vitamin (4%), dental prophylaxis (3.4%)
2	1318	Therapy (4.1%), to treat (4%), cytology (3.8%), breast ultrasound (3%), medicine (2.2%)
3	384	Acidum (31.5%), the nearest hospital (14.3%), proper diet (14.3%), health behavior (14.3%), obstetric control (10.2%)
4	443	To treat (2%), therapy (2%), vitamin (1.8%), diet (1.6%), medicine (1.6%)

For clustering, we selected visits where the description of recommendation and at least one of interview and examination were not empty (some concepts were recognized). It significantly reduced the number of considered visits. Table 1 gives basic statistics of obtained clusters. The last column contains the adjusted Rand index. It can be interpreted as a measure of the stability of the clustering. The higher similarity of the two algorithms, the higher stability of clustering. For determining the optimal number of clusters, for each specialty we consider the number of clusters between 2 and 15. We choose the number of clusters so that adding another cluster does not give a relevant improvement of a sum of differences between elements and clusters' centers (according to so called *Elbow method*).

4 Results

4.1 Analogies in Medical Concepts

To better understand the structure of concept embeddings and to determine the optimal dimension of embedded vectors we use word analogy task introduced in [20] and examined in details in a medical context in [22]. In the former work the authors defined five types of semantic and nine types of syntactic relationship.

We propose our own relationships between concepts, more related to the medical language. We exploit the fact that in the corpus we have a lot of multiword concepts and very often the same words are included in different terms. We would like the embeddings to be able to catch relationships between terms. A question in the term analogy task is computing a vector: $vector(left\ foot) - vector(foot) + vector(hand)$ and checking if the correct $vector(left\ hand)$ is in the neighborhood (in the metric of cosine of the angle between the vectors) of this resulting vector.

We defined seven types of such semantic questions and computed answers' accuracy in a similar way as in [20]: we created manually the list of similar term pairs and then we formed the list of questions by taking all two-element subsets of the pairs list. Table 2 shows the created categories of questions.

We created one additional task, according to the observation that sometimes two different terms are related to the same object. This can be caused for example by the different order of words in the terms, e.g. *left wrist* and *wrist left* (in Polish both options are acceptable). We checked if the embeddings of such words are similar.

We computed term embeddings for terms occurring at least 5 times in the descriptions of the selected visits. The number of chosen terms in interview descriptions was equal to 3816 and in examination descriptions – 3559. Among these there were 2556 common terms for interview and examination. Embeddings of the size 10 to 200 were evaluated. For every embedding of interview terms there was measured accuracy of every of eight tasks. Table 3 shows the mean of eight task results. The second column presents the results of the most restrictive rule: a question is assumed to be correctly answered only if the closest term of the vector computed by operations on related terms is the same as the desired answer. The total number of terms in our data set (about 900,000 for interviews) was many times lower than sets examined in [20]. Furthermore, words in medical descriptions can have a different context that we expect. Taking this into account, the accuracy of about 0.17 is very high and better than we expected. We then checked the closest 3 and 5 words to the computed vector and assumed a correct answer if in this neighbourhood there was the correct vector. In the biggest neighbourhood the majority of embeddings returned accuracy higher than 0.5.

For computing visit embeddings we chose embeddings of dimensionality 20, since this resulted in the best accuracy of the most restrictive analogy task and it allowed us to perform more efficient computations than higher dimensional representations. Figure 1 illustrates PCA projection of term embeddings from four categories of analogies.

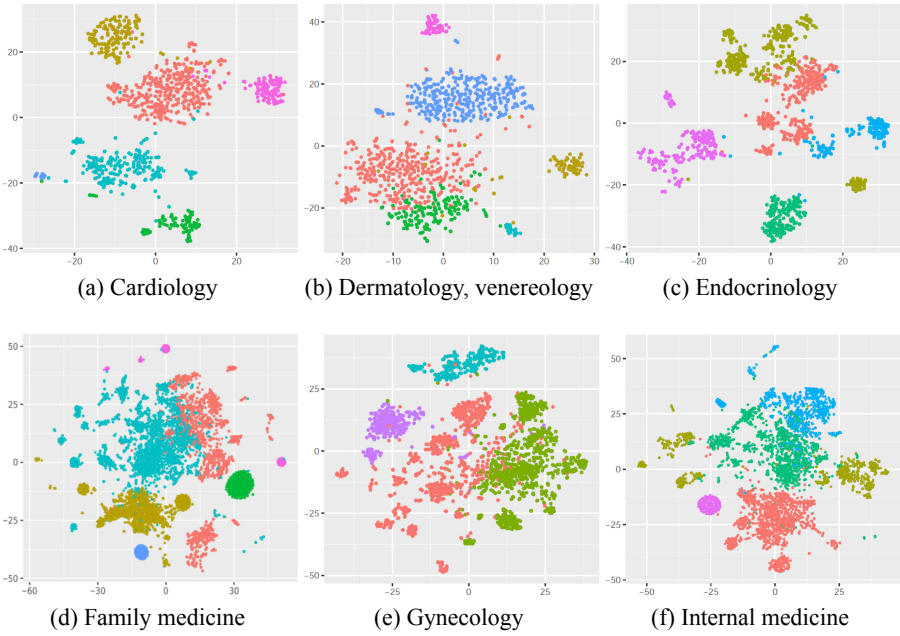


Fig. 2. Clusters of visits for selected domains. Each dot corresponds to a single visit. Colors correspond to segments. Visualization created with t-SNE. (Color figure online)

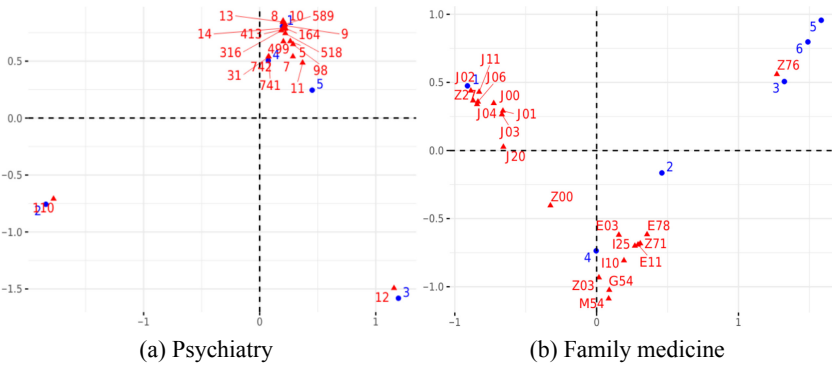


Fig. 3. Correspondence analysis between clusters and doctors’ IDs for psychiatry clustering (panel a) and between clusters and ICD-10 codes for family medicine clustering (panel b). Clusters 2 and 3 in panel a are perfectly fitted to a single doctor.

4.2 Visits Clustering

Clustering was performed separately for each specialty of doctors. Figure 2 illustrates two-dimensional t-SNE projections of visit embeddings coloured by clusters [18]. For some domains clusters are very clear and separated (Fig. 2a). This corresponds with the high stability of clustering measured by Rand index.

In order to validate the proposed methodology we evaluate how clear are derived segments when it comes to medical diagnoses (ICD-10). No information about recommendations nor diagnosis is used in the phase of clustering to prevent data leakage.

Figure 3(b) shows correspondence analysis between clusters and ICD-10 codes for family medicine clustering. There appeared two large groups of codes: the first related to diseases of the respiratory system (J) and the second related to other diseases, mainly endocrine, nutritional and metabolic diseases (E) and diseases of the circulatory system (I). The first group corresponds to Cluster 1 and the second to Cluster 4. Clusters 3, 5 and 6 (the smallest clusters in this clustering) covered Z76 ICD-10 code (encounter for issue of repeat prescription). We also examined the distribution of doctors' IDs in the obtained clusters. It turned out that some clusters covered almost exactly descriptions written by one doctor. This happens in the specialties where clusters are separated with large margins (e.g. psychiatry, pediatrics, cardiology). Figure 3(a) shows correspondence analysis between doctors' IDs and clusters for psychiatry clustering.

4.3 Recommendations in Clusters

According to the main goal of our clustering described in Introduction, we would like to obtain similar recommendations inside every cluster. Hence we examined the frequency of occurrence of the recommendation terms in particular clusters.

We examined terms of recommendations related to one of five categories: procedure to carry out by patient, examination, treatment, diet and medicament. Table 4 shows an example of an analysis of the most common recommendations in clusters in gynecology clustering. In order to find only characteristic terms for clusters we filtered the terms which belong to one of 15 the most common terms in at least three clusters.

5 Conclusions and Applications

We proposed a new method for clustering of visits in health centers based on descriptions written by doctors. We validated this new method on a new large corpus of Polish medical records. For this corpus we identified medical concepts and created their embeddings with GloVe algorithm. The quality of the embeddings was measured by the specific analogy task designed specifically for this corpus. It turns out that analogies work well, what ensures that concept embeddings store some useful information.

Clustering was performed on visits embedding created based on word embedding. Visual and numerical examination of derived clusters showed an interesting structure among visits. As we have shown obtained segments are linked with medical diagnosis even if the information about recommendations or diagnosis were not used for the clustering. This additionally convinces that the identified structure is related to some subgroups of medical conditions.

Obtained clustering can be used to assign new visits to already derived clusters. Based on descriptions of an interview or a description of patient examination we can identify similar visits and show corresponding recommendations.

Acknowledgments. This work was financially supported by NCBR Grant POIR.01.01.01-00-0328/17. PBi was supported by NCN Opus grant 2016/21/B/ST6/02176.

References

1. Apostolova, E., Channin, D.S., Demner-Fushman, D., Furst, J., Lytinen, S., Raicu, D.: Automatic segmentation of clinical texts. In: Proceedings of EMBC, pp. 5905–5908 (2009)
2. Banea, C., Chen, D., Mihalcea, R., Cardie, C., Wiebe, J.: Simcompass: using deep learning word embeddings to assess cross-level similarity. In: Proceedings of SemEval, pp. 560–565 (2014)
3. Biecek, P.: DALEX: explainers for complex predictive models in R. *J. Mach. Learn. Res.* **19**(84), 1–5 (2018)
4. Bodenreider, O.: The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic Acids Res.* **32**(suppl-1), D267–D270 (2004)
5. Chiu, B., Crichton, G., Korhonen, A., Pyysalo, S.: How to train good word embeddings for biomedical NLP. In: Proceedings of BioNLP, pp. 166–174 (2016)
6. Choi, E., et al.: Multi-layer representation learning for medical concepts. In: SIGKDD Proceedings, pp. 1495–1504. ACM (2016)
7. Choi, E., Schuetz, A., Stewart, W.F., Sun, J.: Medical concept representation learning from electronic health records and its application on heart failure prediction. arXiv preprint [arXiv:1602.03686](https://arxiv.org/abs/1602.03686) (2016)
8. Choi, Y., Chiu, C.Y.I., Sontag, D.: Learning low-dimensional representations of medical concepts. *AMIA Summits Transl. Sci.* **2016**, 41 (2016)
9. De Boom, C., Van Canneyt, S., Demeester, T., Dhoedt, B.: Representation learning for very short texts using weighted word embedding aggregation. *Pattern Recogn. Lett.* **80**, 150–156 (2016)
10. De Vine, L., Zuccon, G., Koopman, B., Sitbon, L., Bruza, P.: Medical semantic similarity with a neural language model. In: Proceedings of CIKM, pp. 1819–1822. ACM (2014)
11. Fetter, R.B., Shin, Y., Freeman, J.L., Averill, R.F., Thompson, J.D.: Case mix definition by diagnosis-related groups. *Med. Care* **18**(2), i-53 (1980)
12. Frantzi, K., Ananiadou, S., Mima, H.: Automatic recognition of multi-word terms: the C-value/NC-value method. *Int. J. Digit. Libr.* **3**, 115–130 (2000)
13. Ganesan, K., Subotin, M.: A general supervised approach to segmentation of clinical texts. In: IEEE International Conference on Big Data, pp. 33–40 (2014)
14. Gordon, L., Grantcharov, T., Rudzicz, F.: Explainable artificial intelligence for safe intraoperative decision support. *JAMA Surg.* **154**(11), 1064–1065 (2019)

15. Jaworski, W., Kozakoszczak, J.: ENIAM: categorial syntactic-semantic parser for Polish. In: Proceedings of COLING, pp. 243–247 (2016)
16. Jaworski, W., et al.: Categorial parser. CLARIN-PL digital repository (2018)
17. Kobylińska, K., Mikołajczyk, T., Adamek, M., Orłowski, T., Biecek, P.: Explainable machine learning for modeling of early postoperative mortality in lung cancer. In: Marcos, M., et al. (eds.) KR4HC/TEAAM -2019. LNCS (LNAI), vol. 11979, pp. 161–174. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-37446-4_13
18. Maaten, L.V.D., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**(Nov), 2579–2605 (2008)
19. Marciniak, M., Mykowiecka, A., Rychlik, P.: TermoPL – a flexible tool for terminology extraction. In: Proceedings of LREC, pp. 2278–2284. ELRA, Portorož, Slovenia (2016)
20. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint [arXiv:1301.3781](https://arxiv.org/abs/1301.3781) (2013)
21. Minarro-Giménez, J.A., Marin-Alonso, O., Samwald, M.: Exploring the application of deep learning techniques on medical text corpora. *Stud. Health Technol. Inform.* **205**, 584–588 (2014)
22. Newman-Griffis, D., Lai, A.M., Fosler-Lussier, E.: Insights into analogy completion from the biomedical domain. arXiv preprint [arXiv:1706.02241](https://arxiv.org/abs/1706.02241) (2017)
23. Orosz, G., Novák, A., Prószéky, G.: Hybrid text segmentation for Hungarian clinical records. In: Castro, F., Gelbukh, A., González, M. (eds.) MICAI 2013. LNCS (LNAI), vol. 8265, pp. 306–317. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-45114-0_25
24. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. In: Proceedings of EMNLP, pp. 1532–1543 (2014)
25. Rand, W.M.: Objective criteria for the evaluation of clustering methods. *J. Am. Stat. Assoc.* **66**(336), 846–850 (1971)
26. Ruffini, M., Gavaldà, R., Limón, E.: Clustering patients with tensor decomposition. arXiv preprint [arXiv:1708.08994](https://arxiv.org/abs/1708.08994) (2017)
27. Ward Jr., J.H.: Hierarchical grouping to optimize an objective function. *J. Am. Stat. Assoc.* **58**(301), 236–244 (1963)
28. Waszczuk, J.: Harnessing the CRF complexity with domain-specific constraints. The case of morphosyntactic tagging of a highly inflected language. In: Proceedings of COLING, pp. 2789–2804 (2012)



Decision-Making with Probabilistic Reasoning in Engineering Design

Stefan Plappert^(✉) , Paul Christoph Gembariski , and Roland Lachmayer 

Institute of Product Development, Leibniz University of Hannover, Welfengarten 1a,
30167 Hannover, Germany

{plappert, gembariski, lachmayer}@ipeg.uni-hannover.de

Abstract. The goal of decision making is to select the most suitable option from a number of possible alternatives. Which is easy, if all possible alternatives are known and evaluated. This case is rarely encountered in practice; especially in product development, decisions often have to be made under uncertainty. As uncertainty cannot be avoided or eliminated, actions have to be taken to deal with it. In this paper a tool from the field of artificial intelligence, decision networks, is used. Decision networks utilize probabilistic reasoning to model uncertainties with probabilities. If the influence of uncertainty cannot be avoided, a variation of the product is necessary so that it adjusts optimally to the changed situation. In contrast, robust products are insensitive to the influence of uncertainties. An application example from the engineering design has shown, that a conclusion about the robustness of a product for possible scenarios can be made by the usage of the decision network. It turned out that decision networks can support the designer well in making decisions under uncertainty.

Keywords: Probabilistic reasoning · Decision-making · Engineering design · Decision network · Bayesian network

1 Introduction

For the development of robust products, which are insensitive to uncertainties, it is important to assess possible effects of decisions made in the development process and to consider the relevant influencing factors of tolerances, environments and use cases [1]. Handling uncertainties in this context, e.g. by identifying design rules and learning about sensitivities is a major part in the management of product complexity [2]. Systems that support the designer in the decision-making process have to reduce the uncertainty by providing knowledge or help to estimate possible scenarios, so that the products are robust against possible uncertainties. But almost all computer aids like computer-aided design require discreet parameters [3]. Knowledge-based engineering systems are often used in the late phases of the development process, e.g. to derive variants quickly and check them for validity by tables and rules [4]. Knowledge-based engineering is also beneficial for the conservation of engineering knowledge which is accessible then for later design projects so that uncertainty can be tackled with experience [5].

In this paper, another approach is investigated, targeting at conditional probabilities that arise in the management of requirements that impact the design. In probabilistic reasoning, uncertainties can be represented by probabilities. For this purpose, Bayesian networks and decision networks will be examined in more detail. The structure of this paper is organized as follows: Sect. 2 describes decision making, uncertainty and probabilistic reasoning in engineering design. In Sect. 3 a decision network for the application example of a rotary valve is built. Section 4 provides a summary and describes approaches for further research.

2 Related Work

In the development of products, decisions often have to be made under uncertainty. A tool from the field of artificial intelligence, probabilistic reasoning, offers the possibility to model uncertainties by using probabilities.

2.1 Decision-Making in Product Development

In general, the goal of decision making is to select the most suitable option from a number of possible alternatives [6]. Decisions have to be made by the designer during the entire product development process which may be divided in *task clarification*, *concept*, *embodiment design* and *detailed design*. Since it encompasses an initial requirement management, *task clarification* has a major influence on the later stages, especially on *embodiment design* where first geometric considerations are made and the shape is defined [7]. Decisions in the development of a new product must take into account a selection of different criteria [9], which lead to different scopes of change in the product. Due to time pressure, decisions are often made at short notice in practice, which can lead to significant negative consequences, such as delays in deadlines, limitations in functionality, cost overruns and product quality defects [8]. Therefore, tools that support the designer in decision making have to quickly assess possible consequences.

2.2 Uncertainty in Engineering Design

Uncertainty cannot be avoided or eliminated within the product development process, therefore it is necessary to consider and react to uncertainty [10]. Kreye et al. [11] differentiate four types of manifestation of uncertainty: *context uncertainty*, *data uncertainty*, *model uncertainty* and *phenomenological uncertainty*. When creating a system, uncertainties arise from the input (data uncertainty), the used model (model uncertainty) and the results of the system (phenomenological uncertainty). Context uncertainty, in contrast, describes the influence of the environment on the system. It can be divided into *endogenous uncertainties*, which arise within the system and can be controlled, and *exogenous uncertainties*, which lie outside the system and typically arise during use of the product [12]. Despite their random occurrence, exogenous variables can be seen as a key to assessing the value of a design, because they reflect the way in which the engineer handles variables that cannot be controlled by himself [6].

According to Chalupnik et al. [13], there are different ways to deal with uncertainty. On the one hand, the uncertainty can be reduced by aiming for an increase in knowledge about the system. On the other hand, the system can be protected from the influence of uncertainty. *Active protection* ensures that the system adapts to uncertain situations. Where, in contrast, *passive protection* ensures that the system withstands the influence of uncertainty and therefore no changes need to be made [13]. Robust products are those that are insensitive to uncontrollable factors [14].

2.3 Modeling Uncertainty with Probabilistic Reasoning

Reasoning with uncertainty given limited resources is part of many technical applications in artificial intelligence [15]. Graph-based models have proven to be an important tool for dealing with uncertainty and complexity, as they build a complex system by combining simpler parts [16].

Bayesian Networks. Probabilities are very suitable for the modelling of reasoning with uncertainty [15]. Bayesian networks use the so-called Bayesian rule (1), because evidence is often perceived as an effect of an unknown cause and the goal is to determine the cause [17].

$$P(\text{cause} \mid \text{effect}) = \frac{P(\text{effect} \mid \text{cause}) P(\text{cause})}{P(\text{effect})} \quad (1)$$

The Bayesian rule is explained using a simplified application example for the dosing of powder for the preparation of hot drinks. To support the understanding, the system is reduced to one effect-cause pair. In practice, many effects have very different causes; the multi-causal relationships are discussed in more detail in Sect. 3.2.

In this example the following problem was noticed: the hot drink tastes watery. A possible cause was identified by an insufficient dosage of the powder. For the further solution of the problem it would be helpful to know with which probability the low dosage is the cause for the problem or effect. Based on a statistical analysis, the probability of the effect is $P(\text{watery}) = 0.2$. Since the dosing is carried out automatically, a sensor measures the required powder quantity, which is below its reference value with a probability of $P(\text{low}) = 0.1$. The probability that the hot drink tastes watery if too little powder is dosed is $P(\text{watery} \mid \text{low}) = 0.8$, because the taste is subjective. Based on the information obtained, the Bayesian rule can be used to determine the probability that the low dose of powder is the cause of the watery taste $P(\text{low} \mid \text{watery}) = 0.4$. This leads to the following conclusion that the insufficient dosage of the powder is with a probability of 40% the cause for the watery taste of the hot drink.

According to Russel and Norvig [17], the structure of a Bayesian network can be described by a directed acyclic graph (DAG) in which each node is annotated quantitative probability information. Figure 1 shows a Bayesian network with four nodes, which represents the probabilities for a *watery hot drink* in case of an *incorrect mixing ratio* of a machine for preparing hot drinks due to a *blocked powder supply* or *defective flow sensor* for liquids. The probabilities of the nodes *blocked powder supply* and *defective flow sensor* reflect the probability of their occurrence. The *incorrect mixing ratio* node has the nodes *blocked powder supply* and *defective flow sensor* as parent nodes, so

the probability is described in a conditional probability table (CPT) depending on the probability of the parent nodes. The *watery hot drink* node describes the probability of a watery taste of a hot drink depending on the probability of an *incorrect mixing ratio*.

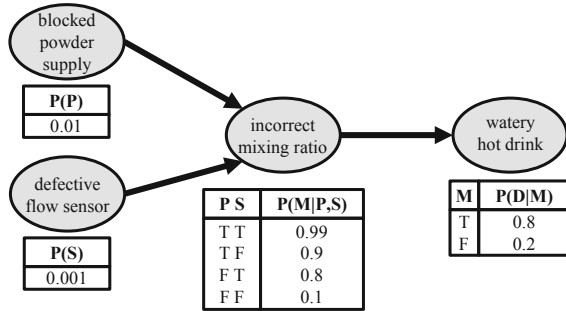


Fig. 1. Simple example for a Bayesian network for a watery hot drink

In a Bayesian network, the direct influences can be displayed by arcs without having to specify each probability manually [17]. The arcs in the DAG specify causal relations between the nodes [18], therefore the Bayesian rule can be applied. The main usage of Bayesian networks is inference, which involves updating the probability distribution of unobserved variables as new evidence or observed variables become available [19].

Decision Networks. A directed acyclic graph (DAG) model that combines chance nodes from a Bayesian network with additional node types for actions and utilities is called a decision network [17]. In general, decision networks can be used for optimal decision making, even if only partial observations of the world are given [20]. According to Zhu [18], a decision network represents the knowledge about an uncertain problem domain, as well as the available actions and desirability of each state. The following three node types, chance nodes, decision nodes and utility nodes form the basic structure of a decision network [17]:

- *Chance Nodes:* random variables as they are used in a Bayesian network, where each node is connected to a conditional distribution indexed by the state of the parent node
- *Decision Nodes:* points where the engineer has a choice of actions to make a decision
- *Utility Nodes:* points with a utility function that describes the preferred outcomes.

Actions are selected based on the evaluation of the decision network for each possible setting of the decision node [17]. Once a decision node is set, the probabilities of the parent nodes of the utility node are calculated by using a standard probabilistic inference algorithm [17]. As a result, the action that has the most added value based on the utility function is selected.

3 Application of Probabilistic Reasoning in Engineering Design

As an application for probabilistic reasoning, an example from engineering design is used to demonstrate a possible handling of uncertainty for the development of robust products.

3.1 Rotary Valve as Application Example

A rotary valve is to be used for the dosing of bulk food for hot drinks. Rotary valves or metering feeders are generally used for metering and conveying free-flowing bulk materials [21]. Figure 2 shows a rotary valve with its components. In this case, the rotary valve is driven via the shaft and receives and transports bulk food through the rotary valve pocket per rotation. To avoid bulk food being drawn in or jammed between the rotary valve and the housing, the gap between the rotary valve and the housing is kept as small as possible.

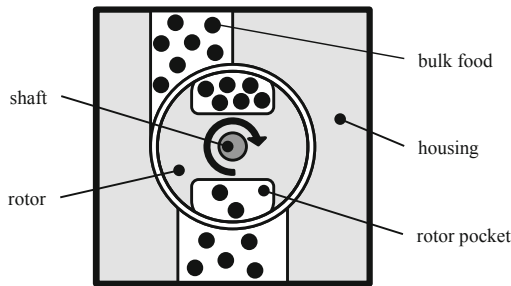


Fig. 2. Rotary valve for the dosing of bulk food

The rotary valve has many advantages in application, as it is easy to handle and provides reproducible results. On the other hand, the dimensioning or design of the rotary valve requires adaptation to the bulk material properties [21]. Especially for discontinuous and quantitative dosing, the rotor pocket size is decisive, which should also be filled as completely as possible during dosing.

The rotary valve in this application example is integrated in a machine for the preparation of various hot drinks. Each hot drink requires a different amount of bulk food to be dosed. In addition, the machine is to be placed at different locations, such as in the home kitchen, the office or a café. These boundary conditions result in uncertainties that cannot be influenced by the engineer. The aim for the engineer is to cover as many possible and probable scenarios with one size of the rotor pocket.

3.2 Modelling of a Decision Network for the Application Example

To support the design engineer in the decision making process for the optimal rotor pocket size, a decision network was established which also represents the given uncertainties due to the boundary conditions.

The decision network (Fig. 3) of the application example consists of a Bayesian network with six chance nodes. The nodes *bulk food* and *place of use* form the initial nodes. The nodes *weight* and *density* depend on the selected *bulk food*. Depending on the *place of use*, a different quantity of liquid is required for the hot drinks, because the number of hot drinks needed is different. The dosing volume depends on the *weight* and *density* of the *bulk food*, and on the required quantity of liquid. The additional chance node *filling level* represents the uncertainty when filling the rotor pocket size with *bulk food*. The decision node *rotor pocket size* represents the different pocket sizes which are available as possible actions. The utility node *utility function* represents the preferred outcomes, where design conditions are also included.

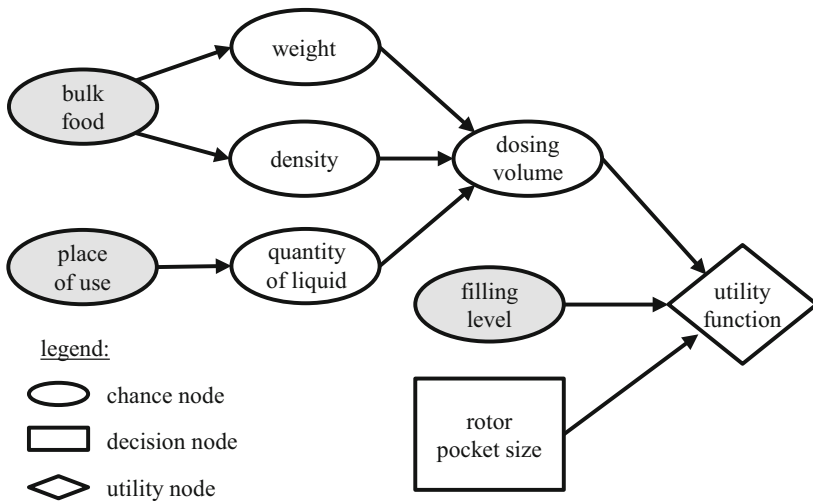


Fig. 3. Decision Network for rotary pocket size

The decision network for the application example was built within Matlab using an open-source package for directed graphical models called Bayes Net Toolbox (BNT). A great strength of BNT is the variety of implemented inference algorithms [20]. In addition, Matlab is very suitable for rapid prototyping, because the Matlab code is high level and easy to read [20].

In the first step, the Bayesian network was represented by six chance nodes (Fig. 4). For the chance nodes *bulk food* and *location* the occurrence probabilities were stored. For the chance nodes *weight*, *density* and *quantity of liquid* the probabilities were stored with conditional probability tables (CPT). CPTs were used, because the application example contains only discrete variables and thus the inference was simplified. The probability values are taken from a similar project and were determined empirically.

For the selection of the possible rotor pocket size, in addition to the possible dosing volumes, the filling level of the rotor pocket size has to be considered. For this purpose, the following assumptions are made that the rotor pocket size has a filling level of 0.9 at 80% and that filling levels 1.0 and 0.8 occur with a probability of 10%. To determine the possible rotor pocket sizes, all divisors with one decimal place of the possible dosing

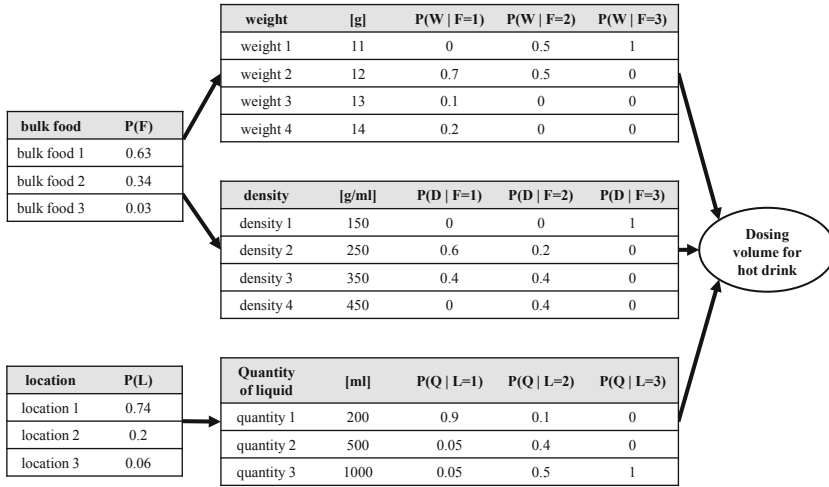


Fig. 4. Bayesian network for dosage in the preparation of hot drinks

volumes were determined. This ensures that all rotor pocket sizes are considered for the required dosing volume.

The general utility function describes the sum of the utilities of all possible outcomes, weighted according to their probability of occurrence [6]. As this general utility function also contains results that lead to an unsuitable layout of the design, the following conditions are also represented in the utility function:

- *High Variability*: One rotor pocket size should be able to cover many different dosing volumes
- *Fast Dosing*: The time for dosing should not take longer than the heating time of the liquid for the hot drink
- *Exact Dosage*: the rotor pocket size should dose exactly the required dosage volume.

The listed conditions lead to conflicts which have to be resolved by the algorithm. For example, high variability leads to the smallest possible rotor pocket size, whereas fast dosing requires the largest possible rotor pocket size. In the following Sect. 3.3 the results of the algorithm for the application example are presented and discussed.

3.3 Results of the Decision Network

The aim of the application example is to support the engineer with a decision network in the selection of the optimum rotor pocket size. Table 1 shows the results of the decision network in Matlab at different information levels. The column known information represents the different levels of evidence or observation for the decision network. The column rotor pocket size shows the optimal rotor pocket size for the given evidence, which is the best choice for dosing the most likely dosing volumes. The column utility probability describes the added probabilities of the dosing volumes, which can be dosed

with the rotor pocket size depending on the filling level. For the situation where no further information is available, the optimum rotor pocket size is 10.7 ml with a utility probability of 15.26%. It can also be noticed that a higher information level does not necessarily lead to a higher utility probability, as this depends on the uncertainty or diversity of the probabilities of the chance nodes. This can also be shown by comparing the utility probabilities of no known information with 15.26% and bulk food 2 with 15.16%.

Table 1. Results for the optimal rotary pocket size with utility probabilities

Known information	Rotor pocket size [ml]	Utility probability [%]
No known information	10.7	15.26
Bulk food 1	10.7	23.38
Bulk food 2	1.0	15.16
Bulk food 3	16.3	70.64
Place 1	10.7	19.75
Place 2	1.0	16.84
Place 3	53.3	21.80
Bulk food 1 & Place 1	10.7	30.38
Bulk food 1 & Place 2	1.0	20.84
Bulk food 1 & Place 3	53.3	33.60
Bulk food 2 & Place 1	1.0	17.30
Bulk food 2 & Place 2	0.8	24.00
Bulk food 2 & Place 3	0.8	34.00
Bulk food 3 & Place 1	16.3	76.00
Bulk food 3 & Place 2	16.3	48.00
Bulk food 3 & Place 3	81.5	80.00

In addition, the decision network can support the engineer in making decisions under uncertainty by allowing the utility probability to make a prediction about the robustness of a product:

- *High utility probability:* If a value has a high utility probability, it can be assumed that this value is a suitable solution for as many scenarios as possible and therefore no further changes are necessary. Furthermore, it can be concluded that the described uncertainty within the decision network has little influence on the outcome and therefore it represents a robust product.
- *Low utility probability:* With a low utility probability, only a part of the possible scenarios can be covered with one value. This indicates a high diversity within the decision network, which may be due to a higher influence of uncertainty. For this

reason, the product have to be adaptable to different conditions, i.e. it should have a high variability or modifiability.

The aim of the decision network is to minimize or even avoid the need for design changes at a late stage of product development or in the use phase. If the utility probability is high, the most robust variant is chosen as the solution, because it is insensitive to the assumed uncertainties. If the utility probability is low, a variant cannot cover the whole spectrum of possible results. In this case, a portfolio of variants can be compiled to cover as many scenarios as possible. This portfolio enables a fast reaction to changing conditions.

4 Conclusion and Future Research

Decisions in the product development process often have to be made under uncertainty. Uncertainties can occur during product development or arise from the environment when the product is used. There are two possibilities for dealing with uncertainty. On the one hand, the uncertainty can be reduced by increasing knowledge, on the other hand, the product can be protected from the influence of the uncertainty. To be able to react to the influence of uncertainty, the requirements have to be compared with the behavior of the product during production and use. Especially with a large number of requirements and broad requirement corridors, this requires a great effort from the engineers.

The method presented in this paper supports the engineer in decision-making by representing the uncertainties in a decision network using probabilities. With sensitivities, i.e. the minimum and maximum of a requirement corridor, possible use cases are determined with their probability of occurrence. Based on these, statements about the robustness of the product can be made. Furthermore, they enable a feedback with product management to improve product variety, as possible application scenarios and market segments of the product are already checked.

For future research, it is necessary to investigate how suitable the method with the decision network is for selecting appropriate variants. In addition, the volume of a rotor pocket size was exclusively used as the basis for the application example. For the variation of the rotor, the number of pockets on the circumference or a combination of two pocket sizes could also be interesting. For the illustration of different variants with given uncertainty a coupling between a decision network and a knowledge-based system (KBS) would also be conceivable. Here the requirements with their probabilities of occurrence could be represented in a decision network and the most probable result is transferred to the knowledge-based system for the configuration of the product.

References

1. Suh, N.P.: Complexity: Theory and Applications. Oxford University Press, Oxford (2005). On Demand
2. Gembariski, P.C., Lachmayer, R.: A business typological framework for the management of product complexity. In: Bellemare, J., Carrier, S., Nielsen, K., Piller, F.T. (eds.) *Managing Complexity*. SPBE, pp. 235–247. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-29058-4_18

3. Antonsson, E.K., Otto, K.N.: Imprecision in engineering design. *J. Vib. Acoust.* **117**(B), 25–32 (1995). <https://doi.org/10.1115/1.2838671>
4. Plappert, S., Gembariski, P.C., Lachmayer, R.: The use of knowledge-based engineering systems and artificial intelligence in product development: a snapshot. In: Świątek, J., Borzemski, L., Wilimowska, Z. (eds.) *ISAT 2019. AISC*, vol. 1051, pp. 62–73. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-30604-5_6
5. Gembariski, P.C., Bibani, M., Lachmayer, R.: Design catalogues: knowledge repositories for knowledge-based-engineering applications. In: Marjanović, D., Štorga, M., Pavković, N., Bojčetić, N., Škec, S. (eds.) *Proceedings of the DESIGN 2016 14th International Design Conference*, pp. 2007–2016 (2016)
6. Hazelrigg, G.A.: A framework for decision-based engineering design. *J. Mech. Des.* **120**(4), 653–658 (1998). <https://doi.org/10.1115/1.2829328>
7. Pahl, G., Beitz, W.: *Konstruktionslehre*, 1st edn. Springer, Berlin (1977). <https://doi.org/10.1007/978-3-662-02288-7>
8. Lenders, M.: *Beschleunigung der Produktentwicklung durch Lösungsraum-Management*. Apprimus-Verlag (2009)
9. Dostatni, E., Diakun, J., Grajewski, D., Wichniarek, R., Karwasz, A.: Multi-agent system to support decision-making process in design for recycling. *Soft. Comput.* **20**(11), 4347–4361 (2016). <https://doi.org/10.1007/s00500-016-2302-z>
10. Wiebel, M., Eifler, T., Mathias, J., Kloberdanz, H., Bohn, A., Birkhofer, H.: Modellierung von Unsicherheit in der Produktentwicklung. In: Jeschke, S., Jakobs, E.M., Dröge, A. (eds.) *Exploring Uncertainty*, pp. 245–269, Springer Gabler, Wiesbaden (2013). https://doi.org/10.1007/978-3-658-00897-0_10
11. Kreye, M.E., Goh, Y.M., Newnes, L.B.: Manifestation of uncertainty – a classification. In: *DS 68-6: Proceedings of the 18th International Conference on Engineering Design (ICED 2011), Impacting Society through Engineering Design*, pp. 96–107, Lyngby/Copenhagen, Denmark (2011)
12. De Weck, O., Eckert, C., Clarkson, J.: A classification of uncertainty for early product and system design. In: *DS 42: Proceedings of ICED 2007, the 16th International Conference on Engineering Design*, pp.159–160, Paris, France (2007)
13. Chalupnik, M.J., Wynn, D.C., Clarkson, P.J.: Approaches to mitigate the impact of uncertainty in development processes. In: *DS 58-1: Proceedings of ICED 09, the 17th International Conference on Engineering Design*, pp. 459–470, Palo Alto, CA, USA (2009)
14. Ullman, D.G.: Robust decision-making for engineering design. *J. Eng. Des.* **12**(1), 3–13 (2001). <https://doi.org/10.1080/09544820010031580>
15. Ertel, W.: *Grundkurs Künstliche Intelligenz – Eine praxisorientierte Einführung*, 4th edn. Springer, Wiesbaden (2016). <https://doi.org/10.1007/978-3-8348-2157-7>
16. Jordan, M.I. (eds) *Learning in Graphical Models*, pp. 1–5. Springer (1999). <https://doi.org/10.1007/978-94-011-5014-9>
17. Russel, S.J., Norvig, P.: *Artificial Intelligence – A Modern Approach*, 3rd edn. Pearson, London (2016)
18. Zhu, J.Y., Deshmukh, A.: Application of Bayesian decision networks to life cycle engineering in Green design and manufacturing. *Eng. Appl. Artif. Intell.* **16**(2), 91–103 (2003). [https://doi.org/10.1016/s0952-1976\(03\)00057-5](https://doi.org/10.1016/s0952-1976(03)00057-5)
19. Moullec, M.L., Bouissou, M., Jankovic, M., Bocquet, J.C.: Product architecture generation and exploration using Bayesian networks. In: *DS 70 - Proceedings of DESIGN 2012, the 12th International Design Conference*, pp. 1761–1770, Dubrovnik, Croatia (2012)
20. Murphy, K.P.: The Bayes net toolbox for matlab. *Comput. Sci. Stat.* **33**(2), 1024–1034 (2001)
21. Vetter, G. (eds) *Handbuch Dosieren*, 2nd edn. Vulkan-Verlag, Essen (2002). <https://doi.org/10.1002/cite.330671121>



Hyperbolic Embeddings for Hierarchical Multi-label Classification

Tomaz Stepišnik^{1,2(✉)} and Dragi Kocev^{1,2,3}

¹ Jožef Stefan Institute, Ljubljana, Slovenia
{tomaz.stepisnik, dragi.kocev}@ijs.si

² Jožef Stefan International Postgraduate School, Ljubljana, Slovenia

³ Bias Variance Labs, d.o.o., Ljubljana, Slovenia

Abstract. Hierarchical multi-label classification (HMC) is a practically relevant machine learning task with applications ranging from text categorization, image annotation and up to functional genomics. State of the art results for HMC are obtained with ensembles of predictive models, especially ensembles of predictive clustering trees. Predictive clustering trees (PCTs) generalize decision trees towards HMC and can be combined into ensembles using techniques such as bagging and random forests. There are two major issues that influence the performance of HMC methods: (1) the computational bottleneck imposed by the size of the label hierarchy that can easily reach tens of thousands of labels, and (2) the sparsity of annotations in the label/output space. To address these limitations, we propose an approach that combines graph node embeddings and a specific property of PCTs (descriptive, clustering and target attributes can be specified arbitrarily). We adapt Poincaré hyperbolic node embeddings to obtain low dimensional label set embeddings, which are then used to guide PCT construction instead of the original label space. This greatly reduces the time needed to construct a tree due to the difference in dimensionality. The input and output space remain the same: the tests in the tree use original attributes, and in the leaves the original labels are predicted directly. We empirically evaluate the proposed approach on 9 datasets. The results show that our approach dramatically reduces the computational cost of learning and can lead to improved predictive performance.

Keywords: Hierarchical Multi-label Classification · Hyperbolic embeddings · Ensemble methods · Predictive Clustering Trees

1 Introduction

In the typical supervised learning setting the goal is to predict the value of a single target variable. The tasks differ by the type of the target variable: binary classification deals with predicting a discrete variable with two possible values, multi-class classification deals with predicting a discrete variable with several possible values and regression deals with predicting a continuous variable. In

many real life problems of predictive modelling the target variable is structured. Examples can be labelled with multiple labels simultaneously and some dependencies (e.g., tree-shaped or directed acyclic graph hierarchy) among labels may exist. The former task is called multi-label classification (MLC), while the latter is called hierarchical multi-label classification (HMC). These types of problems occur in domains such as life sciences (finding the most important genes for a given disease, predicting toxicity of molecules, etc.), ecology (analysis of remotely sensed data, habitat modelling), multimedia (annotation and retrieval of images and videos) and the semantic web (categorization and analysis of text and web pages). The most prominent area in a need of efficient HMC models with premium predictive performance is gene function prediction, where the goal is to predict the functions of a given gene. Gene Ontology [3] organizes 45,000 gene functions into a directed acyclic graph. Hence, the task of gene function prediction can be naturally viewed as a task of HMC.

A significant amount of research effort has been dedicated to developing methods for predicting structured outputs. In this sense, the methods for MLC [5] are the most prominent. The methods that consider hierarchical dependencies among the labels during model learning are less abundant. In two overviews of the HMC task [9, 13], several methods are analyzed based on the amount of information they exploit from the hierarchy of labels during the learning of the models. The main conclusion is that global models (predicting the complete structure as a whole) generally have better predictive performance than the local models (predicting components of the output and then combining them). The success of the HMC methods is limited by two major factors: computational cost and sparsity of the output space. The number of labels (as well as the number of examples and features) for many domains presents a major *computational bottleneck* for all of the HMC methods and various methods cope differently with this. The *sparsity of the output space* pertains to the fact that the number of labels per example as well as the number of examples per label is very small.

We propose to address the two performance limiting issues by embedding the large hierarchical label space to a smaller space. Learning embeddings of complex data such as text, images, graphs and multi-relational data is currently a highly researched topic in artificial intelligence. Related to HMC are the embeddings of graph nodes (e.g., *Poincaré* embeddings [10], latent space embeddings [7], *NODE2VEC* [4]), as well as the embeddings for multi-relational data for information extraction and completion of knowledge graphs (e.g., *RESICAL*, *TRANSE*, *Universal Schema*). We exploit the learned embeddings within the learning of predictive clustering trees (PCTs) – a generalization of decision trees. They support different heuristic functions that guide the tree construction, and different prototype functions that make predictions in the leaves. With different choices of heuristic and prototype functions, they have been used for structured output predictions tasks [8], including HMC [14]. Additionally, PCTs yield state of the art predictive performance for the HMC task [2, 6, 11] and have been extensively used for gene function prediction [11, 12].

The main contributions of this paper are as follows. First, we **learn new embeddings for HMC** by adapting the Poincaré node embeddings to get low dimensional embeddings of label sets assigned to individual examples. Second, we **extend PCTs** so that the heuristic function guiding the tree construction only looks at the embeddings and ignores the original high dimensional label space. This significantly reduces the time needed to construct the trees. The prototype function in the leaves is the same as for standard PCTs when used for HMC and predicts the original labels directly. A mapping from embeddings back to labels or **decoding of the embeddings is not needed**. Third, we **empirically evaluate** our approach on 9 datasets for gene function prediction. The evaluation reveals that it **drastically reduces the computational cost** compared to standard PCTs and, given equal time budget, yields models with **superior predictive performance**.

The remainder of this paper is organized as follows. In Sect. 2, we present our approach and theoretically analyze the reduction in computational cost it offers. Next, we outline the experimental design used to evaluate the performance of the obtained predictive models and discuss the obtained results in Sect. 3. Finally, we conclude and provide directions for further work in Sect. 4.

2 Method Description and Analysis

In this section, we briefly describe the calculation of the HMC embeddings by using the Poincaré hyperbolic node embeddings. We then describe PCTs and their extension so they use the label set embeddings to guide the tree construction.

2.1 Hyperbolic Embedding of Label Sets

A recently proposed approach based on the Poincaré ball model of hyperbolic space was shown to be very successful at embedding hierarchical data into low dimensions [10]. The points in the d -dimensional Poincaré ball correspond to the open d -dimensional unit ball $\mathbb{B}^d = \{x \in \mathbb{R}^d; \|x\| < 1\}$, and the distance between them is given as

$$d(x, y) = \operatorname{arccosh} \left(1 + 2 \frac{\|x - y\|^2}{(1 - \|x\|^2)(1 - \|y\|^2)} \right).$$

This means that points close to the center have a relatively small distance to all other points in the ball, whereas the distances between points close to the border (as the denominator approaches 0) is much greater compared to its Euclidean counterpart). This property makes the space well suited for representing hierarchical data, as the norm of the embedding vector can naturally represent the depth of the node in the hierarchy. For example, the root of a tree hierarchy is an ancestor to all other nodes, and as such can be placed near the center, where the distance to all other points is relatively small. On the other hand, leaves of the tree can be placed close to the boundary.

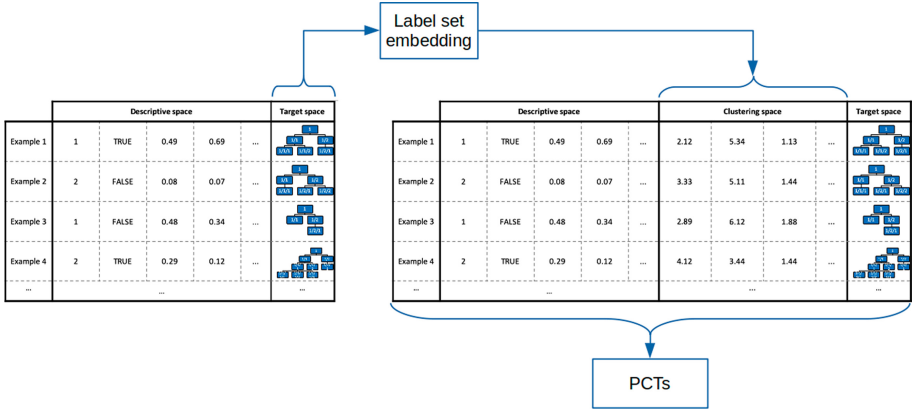


Fig. 1. An overview of our approach. We first calculate the embedding of label sets and add this information to our data. The embeddings are used for the clustering space, which guides the PCT construction, while the input and target space remain the same.

In the HMC task, labels are organized in a hierarchy, which can consist of thousands of nodes. First, we embed the hierarchy into the Poincaré ball, following the proposed method for embedding taxonomies [10]. This way we obtain vectors representing individual labels in the hierarchy. However, each example in a HMC dataset is associated with a set of labels. To get a vector representing the set of labels assigned to an example, we aggregate the vectors representing the individual labels. For the aggregation we have multiple options, for example calculating the component-wise mean vector or the medoid vector.

2.2 Predictive Clustering Trees

Predictive clustering trees (PCTs) are a generalization of decision trees towards predicting structured outputs including hierarchies of labels. In this work, we exploit a unique property of PCTs that allows arbitrary use of the various attributes. Specifically, for the learning of tree models there are three attribute types: descriptive, clustering and target (as illustrated in Fig. 1). The descriptive attributes are used to divide the space of examples; these are the variables encountered in the test nodes. The clustering attributes are used to guide the heuristic search of the best split at a given node. The target attributes are the ones we predict in the leaves.

PCTs are induced with a top-down induction of decision trees algorithm, outlined in Algorithm 1 that takes as input a set of examples and indices of descriptive, clustering and target attributes (can overlap). It goes through all descriptive attributes and searches for a test that maximizes the heuristic score. The heuristic that is used to evaluate the tests is the reduction of impurity caused by splitting the data according to a test. It is calculated on the clustering attributes. If no acceptable test is found (e.g., no test reduces the variance significantly, or the number of examples in a node is below a user-specified

Algorithm 1. Learning a PCT: The inputs are a set of learning examples E , indices of descriptive attributes D , indices of clustering attributes C and indices of target attributes T .

```

1: procedure GROW_TREE( $E, D, C, T$ )
2:    $test = best\_test(E, D, C)$ 
3:   if  $acceptable(test)$  then
4:      $E_1, E_2 = split(E, test)$ 
5:      $left\_subtree = grow\_tree(E1, D, C, T)$ 
6:      $right\_subtree = grow\_tree(E2, D, C, T)$ 
7:     return  $Node(test, left\_subtree, right\_subtree)$ 
8:   else
9:     return  $Leaf(prototype(E, T))$ 
10: procedure BEST_TEST( $E, D, C$ )
11:    $best = None$ 
12:   for  $d \in D$  do
13:     for  $test \in possible\_tests(E, d)$  do
14:       if  $score(test, E, C) > score(best, E, C)$  then
15:          $best = test$ 
16:   return  $best$ 
17: procedure SCORE( $test, E, C$ )
18:    $E_1, E_2 = split(E, test)$ 
19:   return  $impurity(E, C) - \frac{1}{2} impurity(E_1, C) - \frac{1}{2} impurity(E_2, C)$ 

```

threshold), then the algorithm creates a leaf and computes the prototype of the target attributes of the instances that were sorted to the leaf. The selection of the impurity and prototype functions depends on the types of clustering and target attributes (e.g., variance and mean for regression, entropy and majority class for classification). The support of multiple target and clustering attributes allows PCTs to be used for structured target prediction [8].

In existing uses of PCTs, clustering attributes include target attributes, i.e., the splits minimize the impurity of the target attributes. In this work, we take the attribute differentiation a step further and decouple the clustering and target attributes completely. We propose to use the learned embeddings as clustering attributes to guide the model learning and keep the original label vectors as the target attributes. This reduces the dimensionality and sparsity of the clustering space, which makes split evaluation and therefore tree construction faster. Additionally, we do not need to convert the embeddings to the original label space, since the predictions are already calculated in the label space.

We calculate the prototype in each leaf node as the mean of the label vectors (target variables) of the examples belonging to that leaf [14]. The prototype vectors present label probabilities in the corresponding leaves. The variance function is the same as for learning PCTs for multi-target regression (embeddings are continuous vectors), i.e., the weighted mean of variances of clustering attributes.

Like standard decision trees, the predictive performance of PCTs is typically much improved when used in an ensemble setting. Bagging is an ensemble

Table 1. Properties of the datasets used for the evaluation. The columns show the name of the dataset, the number of examples, the number of attributes describing the examples, and the number of labels in the target hierarchy.

dataset	N	D	L
celcycle	3751	77	4125
eisen	2418	79	3573
expr	3773	551	4131
gasch1	3758	173	4125
gasch2	3773	52	4131

dataset	N	D	L
hom	3837	47034	4126
seq	3900	478	4133
spo	3697	80	4119
struc	3824	19628	4132

method that constructs base classifiers by making bootstrap replicates of the training set and using each of these replicates to construct a predictive model. Bagging can give substantial gains in predictive performance when applied to an unstable learner, such as tree learners [1]. It reduces the variance component of the generalization error linearly with the number of ensemble members. This means that there is a limit to how much ensembles can improve the performance (the bias component of the error). At a point the ensemble is saturated, and adding additional trees no longer makes a notable difference [8].

The computational cost of learning a PCT is $\mathcal{O}(DN \log^2 N) + \mathcal{O}(CDN \log N)$, where N is the number of examples, D is the number of descriptive attributes and C is the number of clustering attributes [8]. For standard PCTs, C is the number of labels in the hierarchy (L), which in hierarchical classification is typically much greater than $\log N$, making the second term the main contributor.

In our approach, C is instead the dimensionality of the embeddings (E), which can be only a fraction of the number of labels. We must also take into account the time needed to calculate the embeddings. They are optimized with stochastic gradient descent with time complexity $\mathcal{O}(EL)$ [10].

Therefore, we have reduced the time complexity from $\mathcal{O}(LDN \log N)$ using the standard approach, to $\mathcal{O}(EDN \log N) + \mathcal{O}(EL)$ using our approach. For larger datasets with thousands of examples and/or labels in the hierarchy, using our approach with low dimensional embeddings will offer significant speed gains.

3 Evaluation and Discussion

3.1 Experimental Design

For the evaluation of our approach we use 9 benchmark HMC datasets [14], in which examples are *yeast* genes. In different datasets, different features are used to describe the genes. The goal is to predict gene functions as represented by gene ontology terms. Basic properties of the datasets can be found in Table 1.

We optimize the embeddings using a variant of stochastic gradient descent as recommended in [10]. The only difference is that we do not use the burn-in

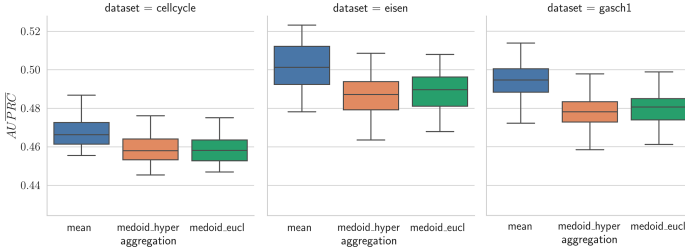


Fig. 2. Performance achieved using different aggregations of label embeddings. For brevity, only three datasets are shown; the results on the other datasets are very similar.

period, as it did not affect our results. We ran the optimization for 100 epochs with batch size 100, which took approximately 10–20 s per dataset using a single Nvidia Titan V graphics card.

In our first set of experiments, we aim to determine the best hyperparameters of our approach: the dimensionality of the embedding vectors and the aggregation function used to combine the embedding vectors of all the labels of an example. Embedding into higher dimensional space makes it easier for the optimization algorithm to find good embeddings, but increases the time required to learn them and learn PCTs on them. We consider dimensionalities from the set $\{2, 5, 10, 25, 50, 100\}$. We also consider three aggregation functions. The simplest one is to calculate the component-wise mean. The other approach is to select the medoid, i.e. the label embedding that is closest to all other label embeddings. Here, we examine both the Euclidean distance and the Poincaré distance as distance between the vectors. Note that the distances discussed here are only used to calculate the medoid of label embeddings. The embeddings themselves are always calculated and optimized in the hyperbolic space. We compare the performance of our approach to two methods: 1) the standard bagging of PCTs and 2) bagging of random PCTs, where at each step a random test to split the data is selected. Bagging of standard PCTs offer state of the art predictive performance for HMC tasks: It is what we would like to achieve or even exceed with our approach, but to learn the trees much faster. For this set of experiments, all ensembles consisted of 50 trees. We used 7 datasets, all but the largest two (hom and struc).

In the second set of experiments, we compare our approach to standard PCTs in a time budgeted manner. We add trees to an ensemble and record the performance and time needed after every couple of trees added. The ensemble is built for one hour or until 250 trees are built (at that point the performance gains are negligible). This evaluation compares our approach to ensembles of standard PCTs given equal time budget. We use the hyperparameters that worked well in the first set of experiments. For all experiments, we use area under the average precision-recall curve ($AUPRC$) [14] to measure the predictive performance. Higher values indicate better performance. To estimate $AUPRC$ we use 10-fold cross validation and report mean and standard deviation over folds. The entire

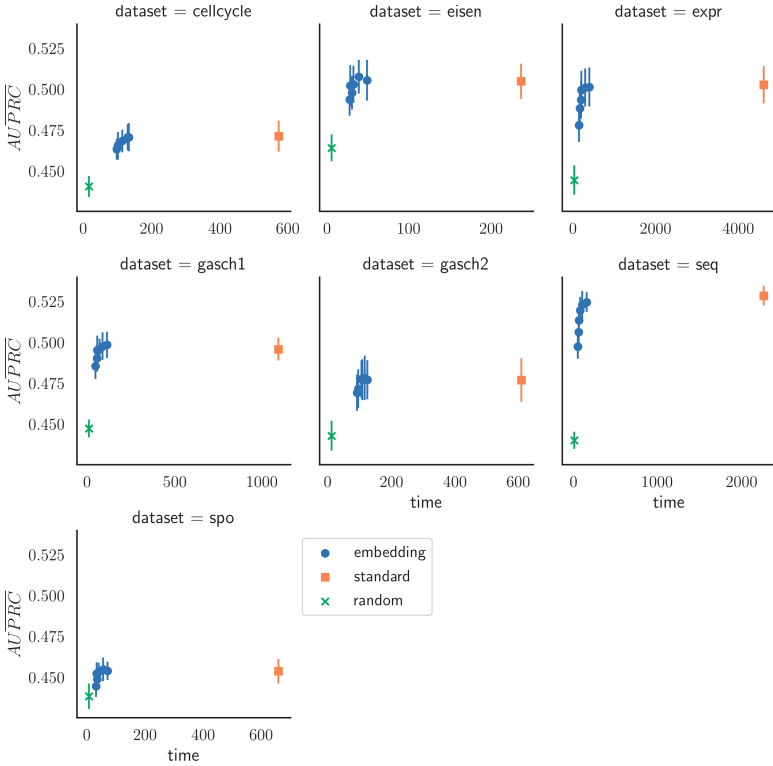


Fig. 3. The comparison of learning time (seconds) and performance of ensembles of standard PCTs, ensembles of random PCTs, and ensembles of PCTs learned on the embeddings. Vertical lines show standard deviation over folds.

experimental pipeline required to reproduce our experiments is available online at <http://kt.ijs.si/dragikocev/ISMIS2020/ISMIS2020code.zip>.

3.2 Results and Discussion

Figure 2 compares the results using different aggregation functions in the first set of experiments. Both Euclidean and Poincaré distances seem to work equally well for calculating the medoid embedding, in terms of the predictive performance achieved. However, mean aggregation typically offered better performance. Given that mean is also faster to calculate than medoid, especially when examples have many labels, we decided to proceed using mean aggregation.

Figure 3 shows the results obtained with mean aggregation and different embedding dimensionalities. Additionally, it shows the performance of ensembles of standard PCTs and ensembles of random PCTs. First thing to note is that our approach noticeably outperformed ensembles of random PCTs on all datasets.

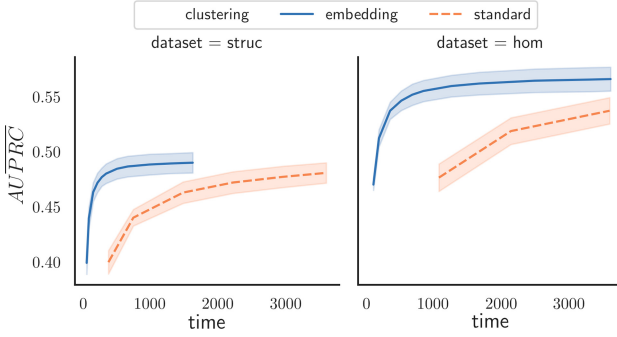


Fig. 4. The relationship between the performance and time (seconds) needed to build the ensemble. With the same time budget, our approach significantly outperforms ensembles of standard PCTs.

This means that the label set embeddings contain useful information about the label hierarchy. We can also see that with enough dimensions, our approach achieves performance that is on par with or exceeds the performance of standard PCTs. Increasing the dimensionality of the embeddings usually improves the performance of the ensemble, but increases the time needed to learn the trees. However, the performance seems to saturate rather quickly, and using more than 25 dimensions rarely results in significant improvement. Most importantly, our approach achieves the performance similar to that of ensembles of standard PCTs in barely a fraction of the time needed to construct them.

Considering the results in Fig. 3, we decided to use the 25-dimensional embeddings for the time-budgeted experiments. The performance improvements with higher dimensions were usually small, whereas the time complexity scales linearly. Figure 4 shows the results of the second set of experiments. On the struc dataset, our approach learned 250 trees well before the hour expired, and on the hom dataset, it learned 225 trees. With the standard approach, we managed to learn only 50 and 20 trees on these datasets, respectively. As discussed in Sect. 2.2, given a high enough time budget, the difference in the number of trees would not make much difference as both ensembles would be saturated. However, the results clearly show that our approach achieves significantly better results much faster. This is especially important when working with large datasets.

4 Conclusions

In this paper, we propose a new approach for solving the HMC task that combines Poincaré hyperbolic node embeddings and PCTs. We aggregate the embeddings of all the labels assigned to an example to obtain low dimensional label set embeddings. We then exploit the property of PCTs that allows us to use the label set embeddings to guide the tree construction, but still predict the original labels directly. Due to the difference in dimensionality between the embeddings

and standard vector representations of label sets, our approach constructs PCTs much faster. Because we still predict the original labels directly, we do not need a mapping from the embeddings back to the labels.

We empirically evaluate our approach on 9 benchmark datasets for gene function prediction. First, we compare the aggregation functions used to combine the label-wise embeddings and show that **aggregation with mean works better** than the medoid aggregation. Second, we show that the models learned with our approach are **much more efficient**: the learning time is 5 or more folds faster than learning in the original space. In some cases they even achieve better predictive performance. Third, in time budgeted experiments we show that our approach achieves **premium predictive performance** much sooner than standard ensembles of PCTs. In applications with limited computational resources, it is clear that the models learned on the embeddings should be preferred.

We plan to extend the work along several dimensions. First, we will look into other node embeddings to compare them to the Poincaré variant. Next, we plan to investigate the effect of different optimization criteria on the performance. For example, we could try optimizing embeddings so that the distance between them is similar to the distance between the labels in the graph. Finally, we will investigate the influence of the embeddings on a wider set of domains.

References

1. Breiman, L.: Bagging predictors. *Mach. Learn.* **24**(2), 123–140 (1996). <https://doi.org/10.1023/A:1018054314350>
2. Cerri, R., Barros, R.C., de Carvalho, A.C., Jin, Y.: Reduction strategies for hierarchical multi-label classification in protein function prediction. *BMC Bioinform.* **17**(1), 373 (2016). <https://doi.org/10.1186/s12859-016-1232-1>
3. Consortium, T.G.O.: The gene ontology resource: 20 years and still GOing strong. *Nucleic Acids Res.* **47**(D1), D330–D338 (2018)
4. Grover, A., Leskovec, J.: Node2vec: scalable feature learning for networks. In: *Proceedings of the 22 ACM SIGKDD Conference (KDD 2016)*, pp. 855–864. ACM (2016)
5. Herrera, F., Charte, F., Rivera, A.J., del Jesus, M.J.: *Multilabel Classification: Problem Analysis, Metrics and Techniques*. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-319-41111-8_2
6. Ho, C., Ye, Y., Jiang, C.R., Lee, W.T., Huang, H.: Hierlpr: decision making in hierarchical multi-label classification with local precision rates (2018)
7. Hoff, P., Raftery, A., Handcock, M.: Latent space approaches to social network analysis. *J. Am. Stat. Assoc.* **97**(460), 1090–1098 (2002)
8. Kocev, D., Vens, C., Struyf, J., Džeroski, S.: Tree ensembles for predicting structured outputs. *Pattern Recogn.* **46**(3), 817–833 (2013)
9. Levatić, J., Kocev, D., Džeroski, S.: The importance of the label hierarchy in hierarchical multi-label classification. *J. Intell. Inf. Syst.* **45**(2), 247–271 (2014). <https://doi.org/10.1007/s10844-014-0347-y>
10. Nickel, M., Kiela, D.: Poincaré embeddings for learning hierarchical representations. In: *Advances in Neural Information Processing Systems 30*, pp. 6338–6347. Curran Associates, Inc. (2017)

11. Radivojac, P.: colleagues: a large-scale evaluation of computational protein function prediction. *Nat. Methods* **10**, 221–227 (2013)
12. Schietgat, L., Vens, C., Struyf, J., Blockeel, H., Kocev, D., Džeroski, S.: Predicting gene function using hierarchical multi-label decision tree ensembles. *BMC Bioinform.* **11**(2), 1–14 (2010)
13. Silla, C., Freitas, A.: A survey of hierarchical classification across different application domains. *Data Min. Knowl. Disc.* **22**(1–2), 31–72 (2011). <https://doi.org/10.1007/s10618-010-0175-9>
14. Vens, C., Struyf, J., Schietgat, L., Džeroski, S., Blockeel, H.: Decision trees for hierarchical multi-label classification. *Mach. Learn.* **73**(2), 185–214 (2008). <https://doi.org/10.1007/s10994-008-5077-3>

Natural Language Processing



Joint Multiclass Debiasing of Word Embeddings

Radomir Popović¹(✉), Florian Lemmerich¹, and Markus Strohmaier^{1,2}

¹ RWTH Aachen University, Aachen, Germany
radomir.popovic@rwth-aachen.de

² Gesis-Leibniz Institute for Social Sciences, Mannheim, Germany

Abstract. Bias in Word Embeddings has been a subject of recent interest, along with efforts for its reduction. Current approaches show promising progress towards debiasing single bias dimensions such as gender or race. In this paper, we present a joint multiclass debiasing approach that is capable of debiasing *multiple* bias dimensions simultaneously. In that direction, we present two approaches, HardWEAT and SoftWEAT, that aim to reduce biases by minimizing the scores of the Word Embeddings Association Test (WEAT). We demonstrate the viability of our methods by debiasing Word Embeddings on three classes of biases (religion, gender and race) in three different publicly available word embeddings and show that our concepts can both reduce or even completely eliminate bias, while maintaining meaningful relationships between vectors in word embeddings. Our work strengthens the foundation for more unbiased neural representations of textual data.

Keywords: Word embedding · Bias reduction · Joint debiasing · WEAT

1 Introduction

Word Embeddings, i.e., the vector representation of natural language words, are key components of many state-of-the-art algorithms for a variety of Natural Language Processing tasks, such as Sentiment Analysis or Part of Speech Tagging. Recent research established that popular embeddings are prone to substantial biases, e.g., with respect to gender or race [2, 6], which demonstrated in results like “*Man is to Computer Programmer as Woman is to Homemaker*” [2] as results of basic analogy tasks. Since such biases can potentially have an effect on downstream tasks, several relevant approaches for debiasing existing word embeddings have been developed. A common deficit of existing techniques is that debiasing is limited to a single bias dimension (such as gender). Thus, in this paper, we propose two new post-processing methods for joint/simultaneous multiclass debiasing, which differ in their trade-off between maintaining word relationships and decreasing bias levels: HardWEAT completely eliminates contained bias as measured by the established Word Embedding Association Test

[3]. SoftWEAT has a stronger and tunable emphasis on maintaining the original relationships between words in addition to bias removal. We demonstrate the effectiveness of our approach on the bias dimensions gender, race and religion on three conventional Word Embedding models: FastText, GloVe and Word2Vec.

2 Background and Related Work

In this section, we introduce key concepts formally and discuss related work on debiasing word embeddings.

We assume a *Word Embedding* with vocabulary size v that maps each word w to a vector representation $\vec{w} \in \mathbf{R}^d$. Protected *classes* $c \in C$ are entities on which bias can exist, e.g., race, religion, gender, or age. *Subclasses* S or *target set of words* refer to directions of a class, e.g., *male, female* for $c = \text{gender}$, and are associated with a set of definitional words S_c . The set of *neutral words* N contains all words that should not be associated with any $S_c \forall c \in C$. Finally, *attribute sets of words* A ($A \subset N$) denote word sets that a target set of words can potentially be linked to, e.g., $\{\text{pleasant, nice, enjoyable}\}$ or $\{\text{science, research, academic}\}$.

The Word Embedding Association Test. The state-of-the-art way of measuring biases in embeddings is the Word Embedding Association Test (WEAT) [3]: It considers two sets of *attribute words* (A, B), e.g., family and career related words, and two target sets (X, Y), e.g., black and white names. The null hypothesis of this test states, that the relative association of target sets' words to both attribute sets' words are equally strong. Thus, rejecting the null hypothesis asserts bias. To examine this, a test statistic s quantifies how strongly X is associated to A in comparison to the association between Y and B . It is computed as $s(X, Y, A, B) = \sum_{\vec{x} \in X} h(\vec{x}, A, B) - \sum_{\vec{y} \in Y} h(\vec{y}, A, B)$, where $h(\vec{w}, A, B) = \text{mean}_{\vec{a} \in A} \cos(\vec{w}, \vec{a}) - \text{mean}_{\vec{b} \in B} \cos(\vec{w}, \vec{b})$ describes the relative association between a single target word $x \in X$ compared to the two attribute sets in a range $[-2, 2]$. Based on s , we assess the statistical significance via an one-sided permutation test through which we acquire p value; Additionally, an effect size d that quantifies the severity of the bias can be calculated as:

$$d(X, Y, A, B) = \frac{\text{mean}_{\vec{x} \in X} h(\vec{x}, A, B) - \text{mean}_{\vec{y} \in Y} h(\vec{y}, A, B)}{\text{std}_{w \in X \cup Y} h(\vec{w}, A, B)}$$

We will use this effect size in our novel debiasing approaches.

Existing Debiasing Techniques. To achieve reduction of bias, we will substantially extend the work of Bolukbasi et al. [2], which describes two ways of debiasing Word Embeddings in a post-processing step: *Hard and Soft Debiasing*. Both rely on identifying gender bias subspace B via Principal Component Analysis computed on gender word pairs differences, such as *he-she*, and *man-woman*. Hard debiasing employs a *neutralize* operation that removes, e.g., all non-gender related words N from a gender subspace by deducting from vectors their bias

subspace projection. Subsequently, an *equalize* operation positions opposing gender pair words (e.g., king, queen) to share the same angle with neutral words. By contrast, Soft Debiasing enables more gradual bias removal by utilizing a tuning parameter λ : An embedding $W \in \mathbb{R}^{d \times |\text{vocab}|}$ is being transformed by optimizing a transformation matrix $T \in \mathbb{R}^{d \times d}$:

$$\min_T \|(TW)^T(TW) - W^T W\|_F^2 + \lambda \|(TN)^T(TB)\|_F^2$$

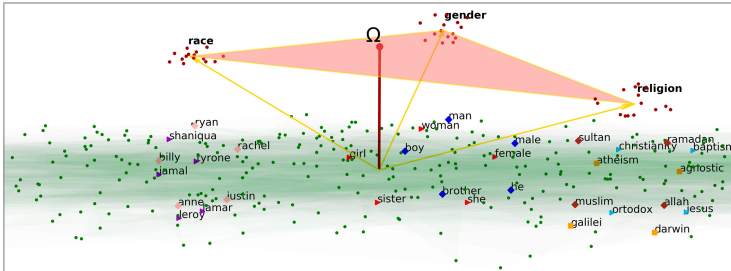


Fig. 1. Visual interpretation of multiclass debiasing via HardWEAT.

Another recent approach by Manzini et al. [10] incorporates these ideas, but expands and evaluates results not only on gender, but separately also on race and religion. It suggests a bias subspace definition for non-binary class environment, which is formulated via PCA of mean shifted k -tuples ($k = \text{number of subclasses}$) of the definitional words. There are also other recently proposed gender bias post-processing [4, 5] and pre-processing techniques [16].

In terms of existing joint multiclass debiasing techniques, *Conceptor Debiasing* [7], is based on applying Boolean-like logic operators using soft transformation matrix on a linear subspace where Word Embedding has the highest variance. We will use this technique for comparison in our experiments.

3 Approach

In this chapter, we present our novel debiasing techniques. Our methods substantially extend previous works of Bolukbasi et al. [2] and Caliskan et al. [3].

HardWEAT: To adapt the *neutralize* step from Bolubasi’s work [2] jointly in a multiclass debiasing setting, we first define the concept of *class definitional vectors* def_c of classes (e.g., race, gender, ...), which are computed as the top component of a PCA on the vector representations of definitional words $D_i \forall i \in \{1, \dots, n\}$ for n subclasses (e.g., male, black, ...), cf. [10]. Now, each particular class can vary in terms of bias amount according to WEAT tests with chosen sets of attribute and target words. Thus, we aggregate WEAT effect sizes d for each

class c into bias levels δ_c by averaging twice: First, we compute the mean value for all target/subclass pairs within a class. Second, we average those means for all results for class c . Then, we compute a *Centroid* $\vec{\Omega}$, as the average of the class definitional vectors weighted by their normalized bias levels: $\vec{\Omega} = \sum_{c \in C} \frac{(\delta_c \cdot \overrightarrow{def}_c)}{\sum_{c \in C} \delta_c}$. We use this centroid to perform *neutralization*, i.e., we re-embed all neutral words such that they perpendicular to it, cf. [2].

To adapt the *equalize* step of Bolubasi’s work, we move the definitional words of all subclasses in a way such that there is no relative preference of any attribute set towards a subclass. For that purpose, we generate equally spread out points e_1, \dots, e_n on some circle with radius r and center \vec{o} as the new center points of the subclasses. In three or more dimensions, this circle can be defined by two vectors (\vec{v}_1, \vec{v}_2) perpendicular to \vec{o} : $\vec{e}_i = \vec{o} + r \cos(\frac{2\pi i}{n}) \cdot \vec{v}_1 + r \sin(\frac{2\pi i}{n}) \cdot \vec{v}_2$.

Given this formula, we use *equidistancing* twice: First, we calculate new temporary central points for each class (e.g., gender) by neutralization (see above), then we determine new temporary central points $\overrightarrow{def}_{S_i}$ for each subclass (e.g., male) in a circle around that. Then, we compute new embeddings \vec{w}_i for all definitional words D_i in a circle around this subclass vector. Note that r_c, r_{S_c} are randomly generated with the condition $r_{S_c} \gg r_c$, whereas \vec{v}_1, \vec{v}_2 are obtained via SVD. Along with an illustration (see Fig. 1), we present the full procedure in Algorithm 1.

For successful equidistancing, i.e., equal dispersion of words, we also require that the length of all subclass/target words within a particular class must be

Algorithm 1: HardWEAT algorithm.

Input: Word Embedding matrix $\mathbf{W}_{v \times d}$, Words W , set of classes C , set of subclasses S , Bias levels δ_c

```

1 for  $c \in C$  do // Generating class definitional vectors
2    $\overrightarrow{def}_c := \text{PCA} \left( \bigcup_{i=1}^n \bigcup_{\vec{w} \in D_i} \vec{w} - \mu_i \right)$ 
3    $\vec{\Omega} = \sum_{c \in C} (\delta_c \cdot \overrightarrow{def}_c)$  // Computing centroid
4    $N = W \setminus B$  // Defining neutral words
5    $N := N - \frac{N \cdot \vec{\Omega}}{\|\vec{\Omega}\|}$  // Neutralizing
6    $\mathbf{W} = \frac{\mathbf{W}}{\|\mathbf{W}\|}$  // Normalizing vectors
7 for  $c \in C$  do // Equidistancing
8    $\vec{O}_c := \overrightarrow{def}_c - \frac{\overrightarrow{def}_c \cdot \vec{\Omega}}{\|\vec{\Omega}\|} \vec{\Omega}$ 
9   for  $i \in \{1, \dots, |S_c|\}$  do
10     $\overrightarrow{def}_{S_i} = \vec{O}_c + r_c \cos(\frac{2\pi i}{|S_i|}) \cdot \vec{v}_1 + r_c \sin(\frac{2\pi i}{|S_i|}) \cdot \vec{v}_2$ 
11    for  $w_j \in W_S$  do
12      $\vec{w}_j = \overrightarrow{def}_{S_i} + r_{S_c} \cos(\frac{2\pi j}{|W_{S_i}|}) \cdot \vec{v}_1 + r_{S_c} \sin(\frac{2\pi j}{|W_{S_i}|}) \cdot \vec{v}_2$ 
13  $\mathbf{W} = \frac{\mathbf{W}}{\|\mathbf{W}\|}$  // Normalizing vectors
```

equal. Violations of this requirement will result in inadequate WEAT scores. Furthermore, HardWEAT could in theory result in some equidistant words becoming angle-close to random words: Thus, to counter this, we design this process in an iterative manner, modifying r_{S_c} until there are no neutral words having an angle greater than certain threshold (e.g., we use 45° as a default). We discuss factors of success and open issues in more detail in Sect. 4.4. Overall, the HardWEAT method ultimately aims at complete bias elimination, but randomizes the topological structure of subclass words.

SoftWEAT is a more gradual debiasing alternative with a greater focus on quality preservation. It provides the user with a choice on how to debias and to what extent. Let us assume that a particular target set of words S_c is closely related in terms of their angle to some attribute set of words A_i (e.g., A_1 is the set of *pleasant* words, A_2 is the set of *intellectual* words). Hypothetically, bias would be minimized if the subclass words S_c would be perpendicular to the attribute sets A_i . Thus, moving S_c towards such a perpendicular space/vector (null vector), noted as \vec{n} , results in bias reduction. Since full convergence towards a vector \vec{n} may result in quality decrease we define a parameter $\lambda \in [0, 1]$ as a *level of removal*. Setting $\lambda = 1$ results in maximal angle decrease between S_c and A_i (perpendicular vectors), while $\lambda = 0$ makes no transformation at all. Note, however, that placing vectors from some subclass words S_i to be perpendicular with attribute words A_i may not necessarily result in overall bias level reductions since WEAT tests are relative measures. E.g., they take the relationship between *male* and *intellectual* words and the relationship between *female* and *appearance*-related words into account at the same time. Additionally, higher λ also pose a greater risk of producing new bias. Moving the representation of subclass words away from attribute words A_1 may result in moving them closer to some other A_2 without prior intent. For example, moving *male* away from *science* can get them closer to *aggressive*, resulting in other WEAT test d increase. To address these issues, we choose a nullspace via SVD that minimizes the average WEAT effect size for all tests.

SoftWEAT executes the following steps: Given some S_c and its corresponding related attribute set of words A_i, \dots, A_n , we generate a matrix $\mathbf{A}_{n \times d}$, where the n rows consist of the first principal components for each A_i respectively. For \mathbf{A} , we then find the nullspace vector \vec{n} that decreases WEAT test scores the most. As a goal, we aim to translate our S_c to this space. Since initially, there might be only few target set words S_c , we expand it by adding for all of them the closest, frequently occurring neighbor words (e.g., the 20 closest neighbors with minimum frequency of 200 in the English Wikipedia [1]). Afterwards, we calculate a transformation vector $\vec{\psi}$ for which it holds that: $\vec{\psi} = \vec{n} - \vec{m}$, where \vec{m} is the mean of S_c words. This transformation vector is then multiplied with a parameter $\lambda \in [0, 1]$ followed by conversion into linear translation matrix Ψ . By translating vectors from extended word sets, we preserve relationships of subclass words S_c to these words. Note that we only modify positions of words from the extended neighborhood of subclass words, but not all words from the vocabulary. Finally, as a last optional step, we normalize all vectors.

In SoftWEAT, the user can decide on the λ , but can also select target and attribute sets used for debiasing. By default, we will only take those into account, for which WEAT scores result in an aggregated effect size of $|d| > 0.6$. To compute this, we accumulate attribute sets per each target set that form matrix \mathbf{A} per each S_c . In case of positive d , S_1 gets removed from A_1 , and S_2 from A_2 . In case of negative d , removal is done other way around. The algebraic operations are formalized as follows: Out of the new matrix \mathbf{W}' with $|S_c|$ rows and $d + 1$ columns, all but the last row are taken as a new vector representation for the given target set of words S_c .

$$\mathbf{A}\vec{n} = 0 \quad \vec{\psi} = \lambda(\vec{n} - \vec{m})$$

$$\mathbf{W}'_{S_c} = \Psi \left[\frac{(W_{S_c})^T}{1} \right] = \begin{bmatrix} 1 & 0 & 0 & \dots & \psi_1 \\ \vdots & \vdots & \ddots & \dots & \vdots \\ 0 & 0 & \dots & 1 & \psi_d \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix} \begin{bmatrix} w_{11} & w_{21} & \dots & w_{n1} \\ \vdots & \vdots & \vdots & \ddots \\ w_{1d} & w_{2d} & \dots & w_{nd} \\ 1 & 1 & \dots & 1 \end{bmatrix}$$

4 Experimental Evaluation

This section presents example results of our methods in practical settings and compares it with Conceptor Debiasing [7] as the current state-of-the-art approach. We measure the bias decrease along with deterioration of embedding quality and show the effects of biased/debiased embeddings in a Sentiment Analysis as a downstream task example. Due to limited space, we also provide an extended set of results in an accompanying online appendix¹.

4.1 Experimental Setup

Datasets: Extending the experimental design from [8], we apply debiasing simultaneously on following target sets/subclasses: (*male, female*) - *gender*, (*islam, christianity, atheism*) - *religion* and (*black and white names*) - *race* with seven distinct attribute set pairs. We collected target, attribute sets, and class definitional sets from literature [3, 8, 10, 11, 15, 16], see our online appendix for a complete list. As in previous studies [7], evaluation was done on three pretrained Word Embedding models with vector dimension of 300: FastText² (English webcrawl and Wikipedia, 2 million words), GloVe³ (Common Crawl, Wikipedia and Gigaword, 2.2 million words) and Word2Vec⁴ (Trained on Google News, 3 million words). For the Sentiment Analysis task, we additionally employed a dataset of movie reviews [9].

¹ <https://git.io/JvL10>.

² <https://fasttext.cc/docs/en/english-vectors.html>.

³ <https://nlp.stanford.edu/projects/glove/>.

⁴ <https://drive.google.com/uc?id=0B7XkCwpI5KDYNINUTTISS21pQmM>.

Quality Assessment: First, we compared ranked lists of word pairs in terms of their vector similarity to human judgement via Spearman’s Rank correlation coefficient [13], by using the collection taken from the Conceptor Debiasing evaluation [7], i.e., *RG65*, *WS*, *RW*, *MEN*, *MTurk*, *SimLex*, *SimVerb*. In addition, we also utilize Mikolov Analogy Test [12].

Methods: Regarding HardWEAT, we specified neutral words through set difference between all words and ones from priorly defined target sets. In terms of SoftWEAT we provide details such as target-attribute sets structure and number of changed words in section F of online appendix. We applied the OR operator in the Conceptor Debiasing, using the same word set of subclasses within the three above defined classes (subspaces in Conceptor Debiasing).

Table 1. Bias levels for gender, race, and religion before debiasing (REG) and after debiasing with Hard/SoftWEAT (HW/SW) or Conceptor (CPT)

		REG	HW	SW $\lambda = 0.2$	SW $\lambda = 0.4$	SW $\lambda = 0.6$	SW $\lambda = 0.8$	SW $\lambda = 1$	CPT
GloVe	gender	0.62	0.0**	0.47	0.3	0.28	0.15	0.09*	0.24
	race	0.71	0.0**	0.6	0.51	0.39	0.3	0.19*	0.43
	religion	0.77	0.0**	0.62	0.46	0.29	0.2	0.16*	0.28
FastText	gender	0.52	0.0**	0.14*	0.17	0.32	0.31	0.32	0.46
	race	0.36	0.0**	0.13*	0.16	0.25	0.3	0.31	0.31
	religion	0.6	0.0**	0.27	0.21*	0.29	0.35	0.42	0.54
Word2Vec	gender	0.63	0.0**	0.48	0.37	0.32	0.2	0.23	0.12*
	race	0.56	0.0**	0.32	0.28	0.2	0.16	0.16*	0.38
	religion	0.47	0.0**	0.31	0.19	0.11*	0.18	0.2	0.28

4.2 Bias Levels and Quality of Word Embeddings

First, we focus on overall bias levels, see Table 1, by measuring WEAT scores before and after debiasing. We observe that HardWEAT removes the measured bias completely as indicated by zero WEAT scores. SoftWEAT also substantially reduces the bias measurements to different degrees for different datasets. In comparison, for example with $\lambda = 1$, SoftWEAT still leads to a stronger reduction in bias compared to the state-of-the-art Conceptor algorithm in all but one measurement.

Regarding quality assessment, Table 2, shows the complete results for seven rank similarity datasets as well as the Mikolov analogy score on the example of the GloVe Word Embedding. As expected, a significant quality drop appears with HardWEAT, most notably on the RG65 dataset. With SoftWEAT, greater λ induce larger modifications of the original embeddings, which corresponds to

a greater drop in embedding quality. However, in three out of eight test settings, SoftWEAT with lower λ settings achieved a similar or higher score and also leads to competitive results compared to Conceptor, see Sect. 4.4 for an in-depth discussion. Extended results can be found in the online appendix.

Table 2. Measurements of quality tasks after debiasing for GloVe embeddings.

	REG	HW	SW $\lambda = 0.2$	SW $\lambda = 0.4$	SW $\lambda = 0.6$	SW $\lambda = 0.8$	SW $\lambda = 1$	CPT
RG65	76.03*	63.42	75.68	75.39	74.34	73.25	71.42	68.73
WS	73.8	69.73	74.0*	73.96	73.15	71.95	70.34	73.48
RW	46.15	46.09	46.15	46.13	46.08	46.02	45.94	52.45*
MEN	80.13	77.52	80.34	80.34*	80.1	79.57	78.74	79.99
MTurk	71.51*	69.78	71.25	70.78	70.37	69.48	68.64	68.24
SimLex	40.88	40.44	41.46	42.0	42.07	42.21	42.18	47.36*
SimVerb	28.74	28.74	28.9	29.15	29.27	29.54	29.78	36.78*
Mikolov	0.65	0.64	0.65*	0.65	0.65	0.64	0.64	0.63

4.3 Sentiment Analysis

To analyze the effects of debiasing in downstream tasks, we study a sentiment analysis task in the context of movie reviews, i.e., we investigate whether we observe significant differences in predicted sentiments when using biased and debiased Word Embeddings. Modifying setup from Packer et al. [14], we added to the end of each test sentences randomized word from opposing set pairs (e.g., a typical black name or a typical white name), and calculated the difference in the prediction, i.e., the *polarity score* according to a simple neural network that takes the respective pretrained word embeddings as pretrained embedding, which is followed by Flatten-operation and a sigmoid layer. The first two layers were not trainable.

For training, we used a combination of binary-cross entropy loss and Ada-Delta optimizer. We trained our model only on sentences with a maximum length of 50 that did not contain any target sets of words. Shuffling test and training set, we then calculated *polarity score* on 15 different model instances using 6 different kinds of Embeddings: Regular (not debiased), SoftWEAT with $\lambda = 0.1, 0.5, 1$, HardWEAT and Conceptor Debiasing with $\alpha = 2$.

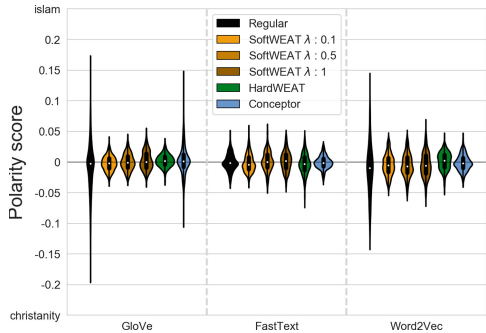


Fig. 2. Classifier Bias on (islam, christianity)

Results for *[christianity, islam]* are shown in Fig. 2. For the non debiased embeddings, we observe that our classifier has no clear trend such that the addition of a bias word in a sentence influences the polarity in a particular direction. However, we can see that bias words have a strong influence on the classifier, leading to a large variance of *polarity scores* in GloVe and Word2Vec-based models. When we apply our debiasing methods, we see that the respective variances are substantially decreased, e.g., around 5.45 times for SoftWEAT already with a small λ of 0.1 in GloVe. This is only the case with Conceptor Debiasing for the Word2Vec-based model, but not for the GloVe-based model.

4.4 Discussion

HardWEAT and SoftWEAT Success Factors: Based on our experiments, see also our online appendix, we could identify a variety of success factors for our methods. Regarding HardWEAT, higher embedding dimensionality and more dispersed vectors are more likely to output more desirable outcome. This is due to the possibility of random words appearing as close neighbors to a target words during the equidistancing procedure, which we counter iteratively as described above. Regarding centroid ($\vec{\Omega}$) neutralization, relevant factors of importance are number of classes, uniformity of bias levels and angles between all def_c : Having more classes with uniform bias levels and distant angles can output non-desirable results, e.g., too large distances between $\vec{\Omega}$ and \vec{def}_c points. However, the user could also manually define an alternative centroid $\vec{\Omega}$. We acknowledge that *equidistancing* comes with a main drawback, which is the partial loss of relationship between target and non-target words, which is also reflected in a quality drop according to different metrics. Regarding SoftWEAT, decreased angles between target and attribute sets after translation may not necessarily result in lower bias levels, since WEAT is a relative measure. However, we take one nullspace which minimizes these values. Furthermore, we note that in our experiments we used all target and attribute set pairs within each of the WEAT tests, which can be further optimized: We may not want to debias something, which isn't priorly biased. E.g., removing *male* from *science* may be necessary, whereas doing the same for *female* from *art* may not, thus we could exclude this latter pair. Also, we should bear in mind, that attribute sets of words are often correlated (as also shown by our experiments in the online appendix). This implies that by moving subclass words towards a specific set of attribute words, we automatically change the associations also with other attribute sets. Thus, the user plays a crucial role in deciding which sets should be used for debiasing.

Comparison with Conceptor Debiasing: Given the experimental results, we conclude that neither Conceptor Debiasing nor SoftWEAT outperform each other. Yet, SoftWEAT exhibits some distinctive advantages: (i) With SoftWEAT, relationships within the target set words remain completely the same, whereas in Conceptor Debiasing, overall angle distribution gets more narrow (See online appendix for more details). (ii) We argue that with SoftWEAT, user gets more freedom with choosing on which target/attribute set combination and to

which degree debiasing is applied. (iii) Given our method, there is no difference in word representation if one uses small subset of neutral words or complete vocabularies. Nevertheless, we acknowledge that Conceptor Debiasing does succeed in reducing bias equally well by applying it in more global behavior.

5 Conclusion

In this paper, we presented two novel approaches for multi-class debiasing of Word Embeddings. We demonstrated the general viability of these methods for reducing and/or eliminating biases while preserving meaningful relationships of the original vector representations. We also analyzed the effects of debiased representations in Sentiment Analysis as an example downstream task and find that debiasing leads to substantially decreased variance in the predicted polarity. Overall, our work contributes to ongoing efforts towards providing more unbiased neural representations of textual data.

Acknowledgements. We want to thank S. Karve, L. Ungar, L., and J. Sedoc for providing the code for Conceptor Debiasing and their support.

References

1. Arora, S., Liang, Y., Ma, T.: A simple but tough-to-beat baseline for sentence embeddings. In: International Conference on Learning Representations (2017)
2. Bolukbasi, T., Chang, K.W., Zou, J.Y., Saligrama, V., Kalai, A.T.: Man is to computer programmer as woman is to homemaker? Debiasing word embeddings. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) *Advances in Neural Information Processing Systems* 29, pp. 4349–4357 (2016)
3. Caliskan, A., Bryson, J.J., Narayanan, A.: Semantics derived automatically from language corpora contain human-like biases. *Science* **356**(6334), 183–186 (2017)
4. Dev, S., Phillips, J.: Attenuating bias in word vectors. In: International Conference on Artificial Intelligence and Statistics, pp. 879–887 (2019)
5. Font, J.E., Costa-jussà, M.R.: Equalizing gender bias in neural machine translation with word embeddings techniques. In: *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pp. 147–154 (2019)
6. Garg, N., Schiebinger, L., Jurafsky, D., Zou, J.: Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proc. Natl. Acad. Sci.* **115**(16), E3635–E3644 (2018)
7. Karve, S., Ungar, L., Sedoc, J.: Conceptor debiasing of word representations evaluated on WEAT. In: *Workshop on Gender Bias in NLP* (2019)
8. Knoche, M., Popović, R., Lemmerich, F., Strohmaier, M.: Identifying biases in politically biased wikis through word embeddings. In: *Conference on Hypertext and Social Media*, pp. 253–257 (2019)
9. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: *Annual Meeting of the ACL: Human Language Technologies*, pp. 142–150 (2011)
10. Manzini, T., Chong, L.Y., Black, A.W., Tsvetkov, Y.: Black is to criminal as caucasian is to police: detecting and removing multiclass bias in word embeddings. In: *Conference of the North American Chapter of the ACL*, pp. 615–621 (2019)

11. May, C., Wang, A., Bordia, S., Bowman, S., Rudinger, R.: On measuring social biases in sentence encoders. In: Conference of the North American Chapter of the ACL: Human Language Technologies, pp. 622–628 (2019)
12. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint [arXiv:1301.3781](https://arxiv.org/abs/1301.3781) (2013)
13. Myers, L., Sirois, M.J.: Spearman Correlation Coefficients, Differences Between. American Cancer Society (2006)
14. Packer, B., Mitchell, M., Guajardo-Céspedes, M., Halpern, Y.: Text embeddings contain bias. Here’s why that matters. Technical report, Google (2018)
15. Zhao, J., Wang, T., Yatskar, M., Ordonez, V., Chang, K.W.: Gender bias in coreference resolution: evaluation and debiasing methods. In: Conference of the North American Chapter of the ACL, pp. 15–20 (2018)
16. Zhao, J., Zhou, Y., Li, Z., Wang, W., Chang, K.W.: Learning gender-neutral word embeddings. In: Conference on Empirical Methods in Natural Language Processing, pp. 4847–4853 (2018)



Recursive Neural Text Classification Using Discourse Tree Structure for Argumentation Mining and Sentiment Analysis Tasks

Alexander Chernyavskiy^(✉) and Dmitry Ilvovsky

National Research University Higher School of Economics, Moscow, Russia
alschernyavskiy@gmail.com, dilv_ru@yahoo.com

Abstract. This paper considers sentiment classification of movie reviews and two argument mining tasks: verification of political statements and categorization of quotes from an Internet forum corresponding to argumentation (factual or emotional). In the case of the fact-checking problem, justifications can be used additionally in one of its sub-tasks. A strong model for solving these and similar problems still does not exist. It requires the style-based approach to achieve the best results. The proposed model effectively encodes parsed discourse trees due to the recursive neural network. The novel siamese model based on it is suggested to analyze discourse structures for the pairs of texts. In the paper, the comparison with state-of-the-art methods is given. Experiments illustrate that the proposed models are effective and reach the best results in the assigned tasks. The evaluation also demonstrates that discourse analysis improves quality for the classification of longer texts.

Keywords: Argument mining · Fact-checking · Sentiment analysis · Discourse tree structure · Recursive neural network · Siamese model

1 Introduction

Nowadays, social media users are inundated with factual texts about politics, economics, history and so on. Triggered by the fact that some sources can utilize fake statements for their purposes, it would be unwise to trust all of them. For instance, in the United States over a million tweets contained fake information by the end of the presidential election of 2016 (the “Pizzagate” scandal). Thus, the factual text categorization task is significant for public goods.

The analysis of texts from the point of view of psychology and rhetoric is also challenging. There are many papers devoted to analyzing messages from Internet users regarding the existence of factual argumentation, sarcasm and other factors. This area can be used in dialogue systems, sentiment classification and so forth.

The sentiment analysis consists in determining whether the emotional tone (sentiment) of the social-media text is positive or negative. This task has become more popular over the years. It can be applied in e-commerce, marketing and advertising.

In this work, we review the three following tasks:

1. The fact-checking task. The main goal is to categorize given factual text into several pre-defined classes corresponding to its veracity. Two sub-tasks are considered:
 - (a) Classification of statements alone
 - (b) Classification of pairs of texts (statement and its justification)
2. The detection of emotional and factual argumentation. It is a binary classification task, the main goal of which is to categorize texts from an Internet forum as either factually or emotionally justified.
3. The sentiment classification task. The main objective is to classify movie reviews into positive and negative texts.

This paper aims to develop a universal model capable of effectively solving these and similar tasks.

Keyword features seem to be insufficient to this problem but accounting for the writing style of text may be crucial. The idea of this research is to outperform existing methods due to using additional discourse features, which can be constructively represented as trees employing Rhetorical Structure Theory [15].

The developed method encodes discourse trees with the recursive neural network, specifically the binary TreeLSTM model. The analogous model was utilized only for unlabeled syntax trees in the sentiment analysis and semantic relatedness of two sentences task before and achieved top results. We also propose the siamese recursive neural network based on it to categorize pairs of texts which can be effectively used in the sub-task (b) of the fact-checking problem.

Our main contributions are the following:

- The TreeLSTM model was adopted to get the universal and effective model based on the discourse tree structure.
- The siamese recursive model based on discourse trees was proposed to analyze pairs of texts.

We demonstrate the quantitative benefits of these models applied to discourse trees for four popular datasets. Experiments confirmed that discourse analysis suffices to improve quality for relatively long texts. The siamese model was applied in the fact-checking problem where statements with justifications are given. Additionally, we compared two variants of text embeddings in the leaves of the discourse trees.

The remainder of the paper is organized as follows. Initially, related work is observed. This is followed by a description of discourse features and text representation associated with it. Afterwards, the main models are introduced and described in detail. Finally, datasets, implementation details, obtained results and directions for further research are discussed.

2 Related Work

In this paper, we utilize the style-based approach. The main idea is that the writing style of texts in the aforementioned tasks is different. Rashkin and Choi [21] showed that additional lexical features extracted by the LSTM model improve the accuracy of simple models such as Naive Bayes and Logistic Regression. Further, Galitsky et al. [6, 7] suggested the usage of discourse trees that are encoded by the Tree kernel SVM model to determine disinformation. Bhatia et al. [2], Ji and Smith [10] considered the same idea and presented structures of the recursive neural networks for these trees, where the latter suggested modification of the former. These models were successfully applied in sentiment analysis and similar tasks. Finally, Tai et al. [23] proposed the N-ry TreeLSTM structure that obtained the best results in different text classification tasks using syntax trees.

Besides recurrent and recursive neural networks, the usage of convolutional neural networks is being researched [8, 25]. Deligiannis et al. [4] suggested the graph convolutional neural network which utilizes data about users and publishers for the fake news detection. Huge number of current state-of-the-art approaches consider BERT-based models for text categorization, question answering and other tasks [5, 13, 14]. For instance, Trautmann et al. [24] showed that BERT achieved top results in the argumentation mining task.

Petz et al. [20] investigated the influence of the quality of social media texts on the performance of popular opinion mining algorithms.

We provide a method based on the discourse structure of texts which combines discourse trees and the recursive neural network. We adopted the method described in [23] for the labeled discourse structure.

3 Methods

3.1 Rhetorical Structure Theory

The main advantage of the proposed method is the usage of additional discourse features. According to the Rhetorical Structure Theory [15] text can be represented as a labeled tree. Its leaves are spans of text named Elementary Discourse Units (denote as EDUs) and connected by corresponding discourse relations. As a result, inner nodes are formed and these nodes contain longer spans. Inner nodes are also connected according to relations and so forth. The most popular discourse relations are Elaboration, Condition, Joint and Attribution.

Spans are said to be Nucleus and Satellite, where Nucleus is more essential for understanding the meaning of the text and Satellite involves some additional information. Figure 1 demonstrates an example of a discourse tree. Arrows are drawn from Nucleus vertices to Satellite vertices.

3.2 EDU Embeddings

We consider two variants of text embedding in EDUs.

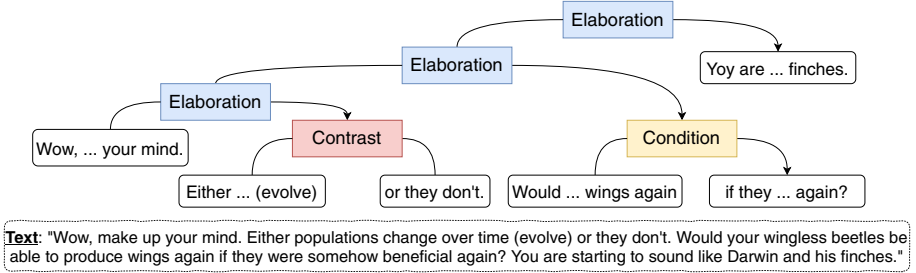


Fig. 1. Discourse tree for text from [4forums.com](https://www.4forums.com)

1. Mean value of word embeddings

We utilized 300-dimensional pre-trained GloVe vectors [19] for word embeddings.

2. Sentence embedding

Universal Sentence Encoder [3] is the open-source model proposed by Google. It achieves the best results in different transfer learning tasks. There are two modifications of its architecture: Transformer network and Deep Averaging Network (DAN), where the former is more qualitative, and the latter is more memory- and time-efficient. One benefit of this model to EDU embeddings is that it ensures that the semantics of the context will be considered, not just the word meaning.

In addition to the aforementioned embeddings, we utilize part-of-speech tags. We used the sum function to aggregate one-hot vectors based on the corresponding POS-tags. The resulting vector representation of the EDU is a concatenation of the text and POS-tag embeddings.

3.3 Recursive Neural Model

The recursive neural network encodes the discourse parse tree as a vector. It calculates embeddings of sub-trees sequentially from the leaves to the root.

Let x_i denote the text embedding corresponding to the node i .

$$x_i = \begin{cases} \text{EDU embedding} & \text{if } i \text{ is the leaf} \\ \text{Embedding of the empty text} & \text{if } i \text{ is the inner node} \end{cases}$$

Text Encoder applies a fully-connected layer to this pre-trained vector:

$$\text{Text_Enc}(i) = \text{FC}(x_i) \quad (1)$$

Let nodes denoted as j and k be children indices for the node i , and r be the name of the discourse relation that characterizes the link between them. The vector representation of the input associated with i concatenates four vectors as follows:

$$t_i = \text{Concat}[\mathbb{I}[j \text{ is Nucleus}], \mathbb{I}[k \text{ is Nucleus}], \text{OneHot}(r), \text{Text_Enc}(i)] \quad (2)$$

In (2) \mathbb{I} is the indicator function. The OneHot function performs a one-hot encoding of the respective relation’s name.

An embedding of the tree which has root in the node i is computed based on embeddings of its left and right sub-trees due to the binary TreeLSTM model [23]. In fact, it is a hidden state of an associated TreeLSTM cell:

$$h_i = \text{TreeLSTM}(t_i, h_j, h_k) \quad (3)$$

In the case of the leaf node, both h_j and h_k are considered to be equal to zero.

We utilize the dropout regularization strategy proposed by Semeniuta et al. [22]. Let c denote a memory cell and α denote a dropout factor. Formally, TreeLSTM with the dropout is expressed with the following equations:

$$\begin{pmatrix} \mathbf{i}_i \\ \mathbf{f}_{i0} \\ \mathbf{f}_{i1} \\ \mathbf{o}_i \\ \mathbf{u}_i \end{pmatrix} = \begin{pmatrix} \sigma(W_i[t_i, h_j, h_k] + b_i) \\ \sigma(W_{f_0}[t_i, h_j, h_k] + b_{f_0}) \\ \sigma(W_{f_1}[t_i, h_j, h_k] + b_{f_1}) \\ \sigma(W_o[t_i, h_j, h_k] + b_o) \\ D(\tanh(W_u[t_i, h_j, h_k] + b_u), \alpha) \end{pmatrix} \quad (4)$$

$$c_i = c_j * \mathbf{f}_{i0} + c_k * \mathbf{f}_{i1} + \mathbf{i}_i * \mathbf{u}_i \quad (5)$$

$$h_i = \mathbf{o}_i * c_i \quad (6)$$

Here, σ is the sigmoid function, D is the Dropout function and $*$ is the element-wise multiplication.

The embedding of the discourse tree obtained in its root is utilized in the categorization task. Two fully-connected layers are applied to the output of the recursive neural network. The softmax layer produces the probabilities of the classes followed by the cross-entropy loss function.

We propose the siamese model to classify pairs of texts. Firstly, it applies the recursive model to calculate the embeddings of the discourse trees constructed for the given texts. At this step, the weights of the base model are shared between two branches. Secondly, it concatenates the resulting embeddings and applies two fully-connected layers to get the final predictions. The architecture of this model is shown in Fig. 2.

The main strength of the resulting models is that they are capable of end-to-end learning.

4 Results

4.1 Detection of Factual and Emotional Argumentation

Internet Argumentation Corpus. This dataset contains messages from the Internet forum 4forums.com. These messages were divided into questions and answers and each text was labeled as “factual” and “feeling”, depending on whether its argumentation is based on facts or emotions. Oraby et al. [16]

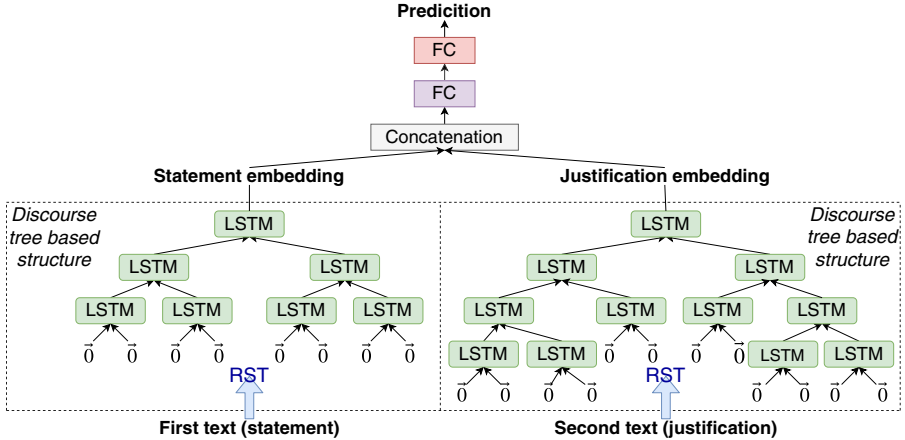


Fig. 2. Siamese model. Here, “LSTM” applies the TreeLSTM cell. Cells with the same color use the same weights. Each cell applied to EDUs receives zero vectors as embeddings of its children.

describe this dataset in detail and consider several simple models to solve the problem. The Internet Argument Corpus (IAC) contains 5848 texts, and 3466 of them are labeled as “factual”. The data is split into train, validation and test sets approximately in the ratio 4:2:1.

Implementation Details. In this research, we utilized ALT Document-level Discourse Parser [11]. Output was transformed into the tree’s format described in Sect. 3.1. We utilized the python Dynet library to implement the recursive neural network. The size of the hidden layer in LSTM cells was established at 300, the dropout rate α at 0.1, the learning rate at 0.001 and the number of units in the fully-connected layer in the Text Encoder at the dimension of x_i . For the model with GloVe embeddings, the best results are achieved when POS-tag embeddings are not considered. We chose the Adagrad optimizer which is less prone to overfitting for the needed task. The optimal number of epochs is 4–9. The model was trained by mini-batches of 60 texts.

Experiments. The parser identified 18 unique discourse relations. It selects “Elaboration” by default. “Attribution”, “Joint” and “Same-Unit” are also popular. “Attribution” usually corresponds to the introductory phrases: “I think that”, “I suppose that” and so forth. “Joint” is indicating the frequent use of the unions. “Same-Unit” indicates that the main meaning of the text in both sub-trees is the same.

We investigated the difference between the distributions of discourse relations in the two classes. The most representative relative frequencies are presented in Table 1. It shows that users employ Elaboration and Same-Unit relations

more often for the argumentation based on facts and Attribution and Condition relations to express their feelings.

Table 1. The most different relative frequencies of the relations in the IAC classes

Discourse relation	For feeling	For factual
Elaboration	0.449	0.483
Attribution	0.132	0.107
Same-Unit	0.078	0.087
Condition	0.031	0.023

Table 2. Model performance on the IAC test set

Model	Features	Fact. P	Fact. R	Feel. P	Feel. R	Macro avg. F1
Random baseline	–	59.08	59.08	40.59	40.59	49.83
Oraby et al. (2015)	patterns	79.9	40.1	63.0	19.2	41.4
Naïve Bayes	unigrams, binary	73.0	67.0	57.0	65.0	65.0
SVM	unigrams	76.14	74.86	64.31	65.81	70.24
CNN	word2vec	82.58	84.72	76.96	74.06	79.56
	dep. embed.	78.49	77.81	68.18	69.04	73.38
	fact. embed.	76.24	74.93	64.49	66.12	70.43
	all embed.	81.98	81.27	73.14	74.06	77.61
LSTM	word2vec	80.60	77.81	69.32	72.80	75.10
	dep. embed.	78.70	76.66	67.34	69.87	73.12
	fact. embed.	78.77	81.27	71.49	68.20	74.90
	all embed.	77.09	82.42	71.63	64.43	73.75
BERT	–	84.64	80.98	74.02	76.27	79.51
Recursive model	GloVe	81.61	81.84	73.53	73.22	77.55
	DAN	83.47	85.88	78.60	75.31	80.79

We compared the proposed approach with the CNN-based and RNN-based models that were described in [8] and with other models provided by the authors of the dataset [16]. Despite it, we trained the BERT model. Precision (P), Recall (R) and macro-averaged F1-score were used to compare the results.

Table 2 shows the results of the comparison. According to the F1-score metric, the recursive model with DAN embeddings obtained the best results. The model utilized the GloVe embeddings got lower quality. The fully-connected layer in the Text Encoder is important since it suffers to improve F1-score by 1–2 points. However, the performance is limited by the mistakes in the discourse trees.

Numerous false predictions are observed for the short texts for which discourse trees have only a few nodes. In this case, discourse does not suffice to categorize it. The quality for deeper trees is presented in Fig. 3. It illustrates that for the big trees results will be much better. On average, the BERT model has not such an improvement in quality.

4.2 Fact-Checking Task

LIAR and LIAR-PLUS Datasets. The LIAR dataset [25] contains short statements collected from the website politifact.com. Each of them was rated by experts on a 6-point scale depending on truthfulness. Binary classification is also possible when all labels less than four indicate lie and the rest indicate truth. The dataset contains 12,782 statements which were split into the train, validation and test samples in the ratio of 10:1:1. This dataset is balanced and the accuracy metric can be used to compare results. The LIAR-PLUS dataset [1] additionally contains human-provided justification for each statement evidence.

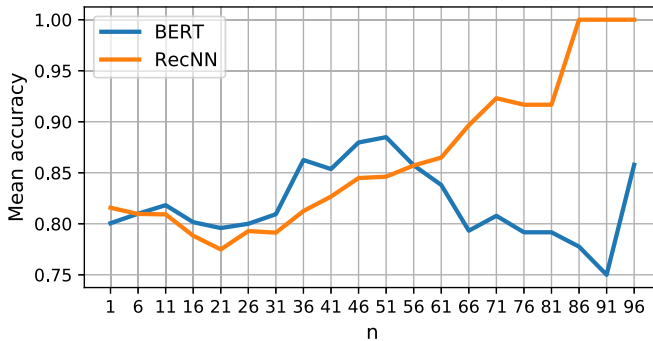


Fig. 3. Performance for the long texts from IAC. Only discourse trees that have at least n nodes are considered

Implementation Details. The implementation details are the same as described in Sect. 4.1 except some hyperparameters. The learning rate was fixed at 0.004 and the size of mini-batches at 150.

Experiments. The discourse parser distinguished 16 unique discourse relations in the LIAR dataset. The most popular relations are “Elaboration”, “Same-Unit” and “Attribution”. More than a quarter of trees (3,787) contain only one node. We investigated the difference between distributions for the instances labeled as “true” and instances labeled as “pants-fire”. The most representative relative frequencies are shown in Table 3. It illustrates that speakers use “Elaboration” and “Contrast” relations more often for the truthful statements whereas “Attribution” and “Enablement” relations are more popular for persuasion in some misleading information.

Table 3. The most different relative frequencies of the relations in classes in the LIAR

Discourse relation	“pants-fire”	“true”
Elaboration	0.377	0.412
Attribution	0.158	0.121
Enablement	0.060	0.043
Contrast	0.015	0.031

The recursive model was compared with the methods discussed in [25] and [17] for the LIAR dataset. Here, we consider models that consume only text data. Table 4 demonstrates the results for the 6-class categorization task. The recursive model with DAN embeddings in EDUs got the highest results. However, the quality is limited by short claims that have trivial discourse structure.

Table 4. Model performance on the LIAR test set

Model	Accuracy
Majority baseline	0.208
SVMs	0.255
Logistic Regression	0.247
LSTM + Attention	0.255
Bi-LSTMs	0.233
CNNs	0.270
MMFD [12]	0.291
BERT	0.288
Recursive model (GloVe)	0.276
Recursive model (DAN)	0.302

We also investigate the performance of the proposed model for pairs of texts in the LIAR-PLUS dataset. We applied the siamese recursive model with DAN embeddings in EDUs to predict results in this case. The comparison with methods from [1] with input SJ (statement + justification) is shown in Table 5. The proposed model obtained the best macro-avg. F1-score for both tasks. It has a significant boost in quality when compared to processing the claim alone.

The fully-connected layer in the Text Encoder is crucial since it adds up to 0.02 to accuracy. The usage of the POS-tags embeddings also improves the overall quality approximately by 0.003-0.01.

Table 5. Model performance (macro-avg. F1-score) on the LIAR-PLUS dataset

Model	Binary		Six-way	
	Valid	Test	Valid	Test
LR	0.68	0.67	0.37	0.37
SVM	0.65	0.66	0.34	0.34
BiLSTM	0.70	0.68	0.34	0.31
P-BiLSTM	0.69	0.67	0.36	0.35
Siamese RecNN	0.71	0.69	0.40	0.40

4.3 Sentiment Classification

Movies Dataset. The main goal is to categorize positive and negative movie reviews from the corpus constructed by Pang and Lee [18]. The dataset contains 1000 instances for each class. The quality is calculated based on the 10-fold cross-validation.

Implementation Details. The implementation details are almost the same as described in Sect. 4.1. The learning rate was fixed at 0.003. The number of units in the FC layer in the Text Encoder was established at 350.

Experiments. The parser identified 18 unique discourse relations. The most popular relations are “Elaboration”, “Same-Unit” and “Joint”. It indicates that humans used mainly multi-nuclear relations to give their opinion about movies. Distributions of the relations are not significantly different for the given classes.

The comparison of the suggested model with recent methods works on discourse is shown in Table 6. These methods got state-of-the-art results before. The accuracy metric was used to compare results. The closest work to our research is the paper by Ji and Smith [10]. In this paper, the recursive neural network based on discourse is also proposed but has another structure. Our model outperformed other methods, and DAN embeddings availed to obtain the best quality.

Table 6. Model performance on the Movies dataset

Model	Accuracy
Hogenboom et al. [9]	71.9
Bhatia et al. [2]	82.9
Ji and Smith [10]	83.1
Our model (GloVe emb.)	84.5
Our model (DAN emb.)	85.2

5 Conclusion and Future Work

This paper presented an efficient model for text categorization using the style-based approach. The architecture of the proposed recursive neural network is developed to encode parsed discourse trees. Additionally, we suggested the siamese model to classify pairs of texts.

Experiments were performed for the sentiment analysis of movie reviews, the analysis of argumentation from the IAC and the automatic verification of political statements from the LIAR and LIAR-PLUS datasets. The proposed model achieved the best quality in all tasks. It was experimentally confirmed that the quality of the model increases greatly as the tree's size increases for the IAC. The siamese model obtained the highest results for the LIAR-PLUS dataset which contains statements with justifications. The DAN embeddings in EDUs suffered to get better results than GloVe embeddings in all cases.

There are possible directions for future work: modification of the architecture of the siamese model, application of CNNs or BERT to obtain more complex text embeddings in EDUs and investigation of the performance of the proposed models in other various tasks where discourse analysis may be helpful. For instance, the siamese model can be used in question-answering systems.

Acknowledgments. The article was prepared within the framework of the HSE University Basic Research Program and funded by the Russian Academic Excellence Project '5-100'.

References

1. Alhindi, T., Petridis, S., Muresan, S.: Where is your evidence: improving fact-checking by justification modeling. In: Proceedings of the First Workshop on Fact Extraction and VERification (FEVER), pp. 85–90 (2018)
2. Bhatia, P., Ji, Y., Eisenstein, J.: Better document-level sentiment analysis from RST discourse parsing. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 2212–2218 (2015)
3. Cer, D., et al.: Universal sentence encoder. arXiv, abs/1803.11175 (2018)
4. Deligiannis, N., Huu, T.D., Nguyen, D.M., Luo X.: Deep learning for geolocating social media users and detecting fake news. In: NATO Workshop (2018)
5. Devlin, J., Chang, M., Lee, K., Toutanova K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT (2018)
6. Galitsky, B., Ilvovsky, D., Kuznetsov, S.: Text classification into abstract classes based on discourse structure. In: RANLP, pp. 200–207 (2015)
7. Galitsky, B., Ilvovsky, D.: Discovering disinformation: discourse level approach. In: 15th National Conference on Artificial Intelligence with International Participation (CAI), pp. 23–32 (2016)
8. Guggilla, C., Miller, T., Gurevych, I.: CNN- and LSTM-based claim classification in online user comments. In: Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers, pp. 2740–2751 (2016)
9. Hogenboom, A., Frasinca, F., de Jong, F., Kaymak, U.: Using rhetorical structure in sentiment analysis. *Commun. ACM* **58**(7), 69–77 (2015)

10. Ji, Y., Smith, N.: Neural discourse structure for text categorization. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pp. 996–1005 (2017)
11. Joty, S., Carenini, G., Ng, R.: A novel discriminative framework for sentence-level discourse analysis. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 904–915 (2012)
12. Karimi, H., Roy, P., Saba-Sadiya, S., Tang, J.: Multi-source multi-class fake news detection. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 1546–1557 (2018)
13. Lan, Z., et al.: ALBERT: a lite BERT for self-supervised learning of language representations. arXiv, abs/1909.11942 (2019)
14. Liu, Y., et al.: Roberta: a robustly optimized BERT pretraining approach. arXiv, abs/1907.11692 (2019)
15. Mann, W., Thompson, S.: Rhetorical structure theory: a theory of text organization. University of Southern California, Information Sciences Institute (1987). https://www.researchgate.net/publication/319394264_Rhetorical_Structure_Theory_A_Theory_of_Text_Organization
16. Oraby, S., Reed, L., Compton, R., Riloff, E., Walker, M., Whittaker, S.: And that’s a fact: distinguishing factual and emotional argumentation in online dialogue. In: Proceedings of the 2nd Workshop on Argumentation Mining (2015)
17. Oshikawa, R., Qian, J., Wang, W.: A survey on natural language processing for fake news detection. arXiv, abs/1811.00770 (2018)
18. Pang, B., Lee, L.: A sentimental education. In: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics - ACL 2004 (2004)
19. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
20. Petz, G., Karpowicz, M., Fuerschuss, H., Auinger, A., Stritesky, V., Holzinger, A.: Computational approaches for mining user’s opinions on the Web 2.0. *Inf. Process. Manag.* **50**(6), 899–908 (2014)
21. Rashkin, H., Choi, E., Jang, J., Volkova, S., Choi, Y.: Truth of varying shades: analyzing language in fake news and political fact-checking. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 2931–2937 (2017)
22. Semeniuta, S., Severyn, A., Barth, E.: Recurrent dropout without memory loss. In: Proceedings of the 26th International Conference on Computational Linguistics (COLING), pp. 1757–1766 (2016)
23. Tai, K., Socher, R., Manning, C.: Improved semantic representations from tree-structured long short-term memory networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (2015)
24. Trautmann, D., Daxenberger, J., Stab, C., Schutze, H., Gurevych, I.: Robust argument unit recognition and classification. arXiv, abs/1904.09688 (2019)
25. Wang, W.: “Liar Pants on Fire”: a new benchmark dataset for fake news detection. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, vol. 2. arXiv, abs/1705.00648 (2017)



Named Entity Recommendations to Enhance Multilingual Retrieval in Europeana.eu

Sergiu Gordea¹(✉), Monica Lestari Paramita², and Antoine Isaac³

¹ AIT Austrian Institute of Technology GmbH, Vienna, Austria
`sergiu.gordea@ait.ac.at`

² University of Sheffield, Sheffield, UK
`m.paramita@sheffield.ac.uk`

³ Free University Amsterdam, Amsterdam, The Netherlands
`aisaac@few.vu.nl`

Abstract. In the past years significant research efforts were invested towards the usage of Named Entity recommendation for improving information retrieval in large and heterogeneous data repositories. Such technology is employed nowadays to better understand user's search intention, to improve search precision and to enhance user experience in web portals. Within the current paper we present a case study on recommending Named Entities for enhancing multilingual retrieval in Europe's digital platform for cultural heritage. The challenges of designing an entity auto-suggestion service able to effectively support users searching for information in [Europeana.eu](https://www.europeana.eu) are described in the paper together with a preliminary experimental evaluation and the outline indicating the directions for future development.

Keywords: Semantic query suggestion · Named Entities · Knowledge graph

1 Introduction

[Europeana.eu](https://www.europeana.eu) currently provides access to more than 50 million (digitized) cultural heritage objects provided by 3700+ partner institutions across Europe. Most often, each individual item is described in one of the European languages, using data structures specific to the individual domains: galleries, libraries, archives and museums. The rich degree of multilinguality within these items create inherent issues for users seeking for particular cultural heritage objects within [Europeana.eu](https://www.europeana.eu) web portal. Therefore, it is difficult for visitors to find the most relevant content items when using search terms in their preferred search language (often their native language). While query recommendations are a popular way of supporting users in web portals, the recommendation of Named Entities (e.g., people, places, subject categories) is more appropriate for semantic repositories, in which the information is structured according to a well-defined ontology (i.e.,

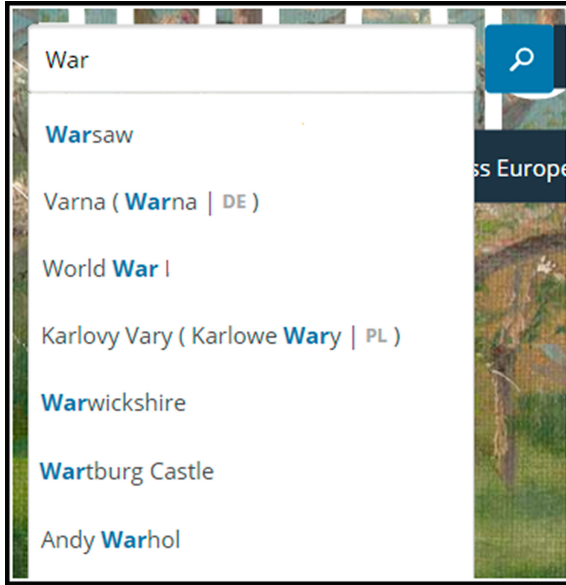


Fig. 1. Auto-suggestion service on [Europeana.eu](https://pro.europeana.eu)

[Europeana.eu](https://pro.europeana.eu) is a semantic repository using the EDM as the underlying data model¹).

In this paper, we describe the implementation of the Europeana auto-suggestion service. This feature utilizes a knowledge graph to aid users in formulating their queries by recommending relevant Named Entities (see Fig. 1). These entity recommendations are used to support end users in building search queries and to improve multilingual retrieval. At the same time the service can improve discovery within the web portal, as it may suggest entities that users are unaware of.

The knowledge graph, further referred to as the Europeana Entity Collection (EC), contains a set of entities related to the Cultural Heritage domain that were extracted from a number of Linked Data repositories. We propose a novel ranking algorithm for the auto-suggestion by combining an internal Europeana feature (i.e., Europeana hit count) and an external feature (i.e., Wikidata Page-Rank). By integrating the EC with external knowledge sources, this approach overcomes the cold start problem that is common within recommendation algorithms [10].

The use of Entity Collection (EC) is intended to address the following issues in [Europeana.eu](https://pro.europeana.eu):

- the *multilinguality problem* when searching for information within the web portal (i.e. closing the gap between the language used for writing search queries and the language used for describing the cultural heritage objects).

¹ See <https://pro.europeana.eu/resources/standardization-tools/edm-documentation>.

- the need for *semantic enrichment* of cultural heritage objects. Utilizing a knowledge graph in the enrichment will link these objects to EC and, consequently, it will enable semantic search to be employed in the future.

In this study, we aim to answer two research questions:

1. Can the knowledge graph (i.e., extracted from Geonames, DBpedia and Wiki-data) be utilized to perform auto-suggestion within the cultural heritage domain?
2. Can the auto-suggestion feature (i.e., utilizing Europeana relevance and Wiki-data PageRank) effectively support users in building their search queries?

The challenges of designing an entity auto-suggestion functionality, which is able to effectively support users searching for information in a multilingual data repository, are described in this paper (Sect. 3) together with a preliminary experimental evaluation (Sect. 4), followed by a discussion (Sect. 5) and an outline of future development (Sect. 6).

2 Related Work

An auto-suggestion feature supports users in formulating their queries by suggesting relevant terms that match the prefix entered by users in the search box [8]. In previous work, the suggested terms were computed globally by extracting popular queries from query logs [15]. Others personalized the suggestions by using user-specific information (e.g., recent queries submitted by the user) [1], or demographic-based information [14]. In both approaches, a large amount of data are needed to compute the popular queries, which is often not available. In this work, we investigated the use of a knowledge graph to auto-suggest relevant named entities.

The use of a knowledge graph has been investigated in various areas in Information Retrieval. In previous work, it has been utilized to support users in disambiguating and expanding their query terms [12, 16]. It has also been used to improve the indexing process and query representation [6, 13], and to improve the ranking and presentation of search results [5, 6, 13].

The auto-suggestion algorithm proposed in this study utilizes the PageRank algorithm [2], which was originally created for the purpose of improving scalability and precision of web search engines. In fact, the algorithm is an effective measure for computing the popularity of web pages. The development of the Wikipedia platform (i.e., including Wikimedia and Wikidata) as an open multilingual data repository, enabled the usage of PageRank algorithm to compute popularity measures for Named Entities and to use them for retrieval, ranking and disambiguation purposes [5]. Similar to our approach, Prasad et al. [12] employed the PageRank algorithm for named entity recognition and linking based on search click-log data. However, the proposed algorithm is language agnostic and not context aware. The current work proposes a language-aware entity suggestion algorithm, which takes into account also the context of its usage (i.e., Europeana.eu as a semantic repository of cultural heritage objects).

The usage of Europeana access logs for computing recommendations on semantic search shortcuts was investigated by Cecarelli et al. [4]. The authors analyzed the user search sessions recorded from August 2010 to January 2012 and suggested Named Entities from the English version of Wikipedia only. The experimental evaluation was carried out against a subset of manually annotated query logs, using relatedness and diversity as quality metrics. In contrast, the current work uses a different recommendation algorithm based on overall popularity of entities and their relevancy for Europeana.eu. Moreover, the recommendations are language-aware and based on the Europeana Entity Collection, which includes only items relevant for the cultural heritage domain.

3 Entity Suggestions for Enhanced Search Experience

The entity auto-suggestion functionality provides recommendations for writing more precise search queries and therefore improves the user experience in Europeana.eu platform. These suggestions are computed based on the Europeana knowledge graph (i.e., Europeana Entity Collection). The following subsections provide an overview of the Entity Collection and the algorithm used to compute the entity recommendations.

3.1 Europeana Entity Collection

The Europeana Entity Collection was curated for three main purposes: i) to support the semantic enrichment for the description of cultural heritage objects available in Europeana, ii) to improve Europeana search effectiveness (i.e., by using names entities in search queries and by enabling a semantic search in the future), and iii) to enable browsing in Europeana.eu platform by offering alternative entry points, through the entity pages (e.g. like the one of Leonardo Da Vinci for example²).

The EC was curated using data extracted from GeoNames, DBpedia, and Wikidata. To avoid redundancy and ambiguous data, only a subset of these resources was selected based on the relevance of entities to Europeana and the cultural heritage domain [9]. Each entity contains the `skos:prefLabel` property which lists the entity names in different languages. This property is utilized by the auto-suggestion feature, therefore, availability of the property values will enable better suggestion and retrieval across languages. The statistics of the EC are shown in Table 1 indicating the number of available entities by their type and the average multilingual coverage of their labels (i.e., as present in the `skos:prefLabel` property).

3.2 Europeana Auto-Suggestion Algorithm

Considering the utility of the suggested entities for end users, the entity recommendations can be split in two stages: the search and the ranking of suggestions.

² <http://data.europeana.eu/agent/base/146741>.

Table 1. Statistics of the Entity Collection

Entity type	Total	Number of average EU languages available for the <code>skos:prefLabel</code> property
Agent	165,005	2.82 EU language (SD = 3.2, min = 1, max = 23, median = 2)
Concept	1,572	15.30 EU languages (SD = 7.11, min = 1, max = 24, median = 17)
Place*	215,802	0.59 EU languages (SD = 1.8, min = 0, max = 24, median = 0)

* All Place entities contain labels, but most were not associated to any EU languages, hence the low average.

The search stage (i.e., the identification of a candidate set of named entities) is implemented based on string matching over the labels available for Named Entities. Additional filtering by entity type, entity language or use in Europeana, are provided in order to further enhance the precision of recommendations. However, the effectiveness of filtering options was not evaluated in the experiments presented in Sect. 4. Given that the list of recommendations is limited to the top 10 entries, the ranking problem is more challenging. In order to ensure high precision, the ranking function takes into account the overall popularity of the Named Entities and their relevance to the Europeana repository. The following function is used to rank the list of recommendations:

$$\begin{aligned}
 score(i) &= \ln(relevance_{ER}(i) * popularity(i)) | i \in EC \\
 relevance_{ER}(i) &= hits_{ER}(i) * \max(hits_{ER}()) / \max(hits_{ER}(t_i)) \\
 popularity(i) &= pageRank(i) * \max(pageRank()) / \max(pageRank(t_i)),
 \end{aligned}$$

where i represents an item from the Entity Collection EC and t_i represents the type of item i (see Table 1). The overall ranking function is computed as the natural logarithm of the product between Europeana relevance $relevance_{ER}$ and the overall popularity $popularity()$, where these two metrics are computed based on the europeana hit count $hits_{er}(i)$ and Wikidata page rank $pageRank(i)$ (see [7]). As the distribution of these metrics over different entity types is quite heterogeneous, both metrics are normalized to facilitate the inclusion of different types of entities within the Top10 recommendations. This normalization is achieved by multiplying the metrics of individual items (i.e. $hits_{ER}(i)$ and $pageRank(i)$) with the ratio between the maximum value of the metric (i.e. $hits_{ER}()$ and $\max(pageRank())$ respectively) and the maximum value of the metric for items having the same type (i.e. $\max(hits_{ER}(t_i))$ and $\max(pageRank(t_i))$ respectively). This normalization ensures that the most relevant item of each type will have the same relevancy score and the most popular item of each type will have the same popularity score.

4 Experiments

Two experiments were conducted to evaluate the performance of the auto-suggestion service to indicate its relevancy and its effectiveness for building enhanced search queries.

4.1 Entity Coverage in Query Logs

Firstly, the auto-suggestion feature is only able to recommend correct entities if the search terms are found in the EC. Therefore, we first evaluated the proportion of Europeana queries that can be mapped to known Named Entities. A reference dataset containing 500 randomly-chosen queries was created by analyzing query logs and extracting the search terms submitted by end users between 1 March and 1 August 2017. Manual annotation was performed on this dataset to indicate if the search queries were involving entities (i.e., Agents, Places or Concepts) or not. For example, the query “Obra compuesta por Lucio Marineo Siculo” (which means “work composed by Lucio Marineo Siculo” in English) was annotated as being related to the person “Lucio Marineo Siculo”. The computed coverage indicates that 74.8% of dataset entries (374 queries) were identified as referring to Named Entities. The contents of the EC were shown to be capable of matching 45.7% of these search queries, which represents 34.2% of the entire reference dataset [11].

4.2 Effectiveness of Entity Auto-Suggestion

Whilst the first experiment focused on the feasibility of employing entity suggestions for building search queries, the second experiment measured the effectiveness of the auto-suggestion service. Provided that the relevant named entities do exist in the EC, we evaluated whether or not the service was able to suggest the relevant entities to end users (i.e., by using the top 10 recommendations). To investigate this aspect, a multilingual test dataset was created by evaluating the most frequent queries searched by Europeana users in 26 languages (i.e., 24 official EU languages³, Norwegian and Russian). The 10 most popular search queries (as reported by Google Analytics for January-February 2018) were collected for each language. Only the top two queries that matched Entity labels for the given language were included in the evaluation dataset. These entries were manually annotated with the IDs of the entity in the EC. The statistics of the resulting dataset are shown in Table 2.

Table 2. Overview of the multilingual test dataset

Number of languages	26 languages
Number of queries	52 queries (2 queries per language)
Type of queries	19 Agents; 29 Places; 4 Concepts
Length of queries (in number of characters)	Mean = 9.10 (SD = 3.92), Min = 4, Max = 18

The performance of the auto-suggestion service (referred to as *Europeana*) is compared to three other methods:

³ See <https://europa.eu/european-union/topics/multilingualism.en>.

- *Baseline*: standard TF-IDF based ranking which is extensively used in information retrieval (i.e. builtin Solr functionality⁴)
- *Relevance*: suggestion ranking based on Europeana relevance function (see 3.2)
- *PageRank*: suggestion ranking based on popularity function (see 3.2)

The auto-suggestion performance was recursively evaluated using the first k characters of each query (entity). Effectiveness was measured using Mean Reciprocal Rank (MRR) and the Success Rate at top- n (SR@10), which indicates the average percentage of queries that retrieve the relevant entity in the top 10 results [3].

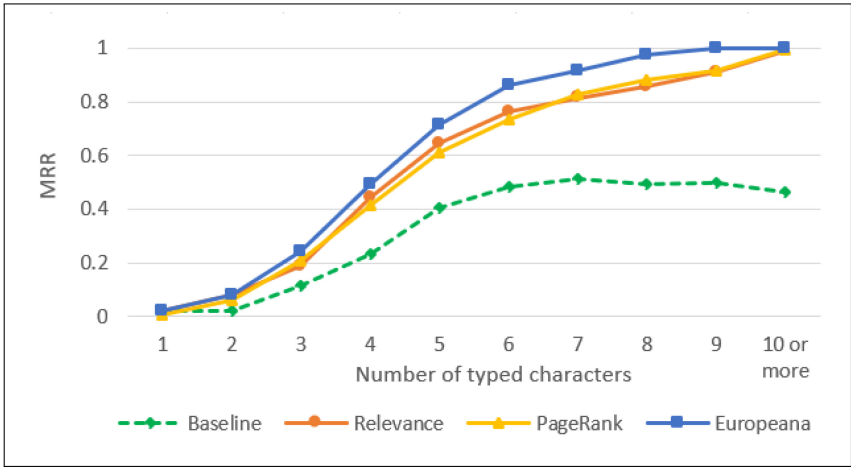
Evaluation Results. The experimental results, presented in Fig. 2, indicate that the overall accuracy of the Europeana auto-suggestion algorithm outperforms the TF-IDF baseline. These results are statistically significant ($p < 0.01$). Given the first 3 characters of users' queries, the auto-suggestion was able to recommend the correct entities for more than half of the cases (SR@10 = 0.48), compared to 31% success rate achieved by the TF-IDF ranking (SR@10 = 0.31). Over 95% cases were completed (i.e., the correct entities were recommended by the auto-suggestion in the top 10) when users have typed the first 5 characters (SR@10 = 0.95). On the given test data set, the *Relevance* and *PageRank* methods provide comparable results, which indicate that the evaluated Entities are both popular in broader sense and relevant for Europeana.eu repository. However, there is a large number of items in the EC, which are not very well known by the public users or they are not referenced by a large number of cultural heritage object available in Europeana.eu. These are subject of investigation for future work. The combination of these two scoring functions in the auto-suggestion algorithm proves its effectiveness especially with regard to the MRR metrics, indicating a better ranking of searched entities within the auto-suggestion list.

Overall, the auto-suggestion was able to recommend the correct entities for all 52 queries with an input (on average) of 3.73 characters (SD = 1.33, min = 1, max = 7). The baseline method was able to suggest the correct entities for 34 queries with an average of 3.74 characters (SD = 1.44, min = 1, max = 7). However, it was not able to suggest the correct entities in the Top 10 list for the remaining 18 queries. TF-IDF algorithm appears to be sensitive to the heterogeneous distribution of item labels over different languages and language scripts (see Table 1), term frequency being low for items with low language coverage and document frequency being high in the case of common words, like person first names for example. The drop in the precision encountered at 8th letter in the query string is explained by the low ranking precision for the first 1 to 3 letters in the second word of the query.

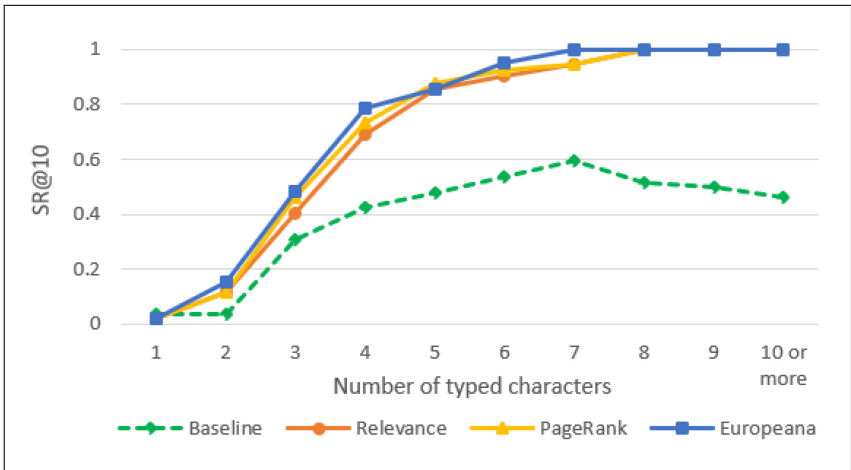
The performance of the auto-suggestion features for the different entity types were also evaluated, as shown in Table 3. The results show that the auto-suggestion achieved a high SR@10 for Agents and Places for an input of 5 characters or more (0.72 and 0.92, respectively). In all four cases, the Concepts in

⁴ <https://lucene.apache.org/core/7.6.0/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html>.

the dataset were retrieved using the first 3 characters only, but this achievement is also related to the relatively small number of concepts included in the EC. A comparison of the MRR scores also show a better accuracy for Concept queries, followed by Places, and Agents. However, in all cases, high MRR scores were achieved by providing 7 characters or more for all entity types.



(a) MRR



(b) SR@10

Fig. 2. Auto-suggestion accuracy (n = 52)

The overall results show a much higher SR@10 compared to MRR, which suggest that, although the correct entity was found in the top 10, it was not suggested as the first ranked item. After further analysis, we found two reasons

Table 3. Auto-suggestion accuracy by entity type

No of chars	MRR			SR@10		
	Agents	Concepts	Places	Agents	Concepts	Places
3	0.26	0.71	0.17	0.47	1	0.41
5	0.68	1	0.69	0.72	1	0.92
7	0.87	1	0.94	1	1	1
9	1	1	1	1	1	1

that caused the lower MRR scores. The first is the *strict one-to-one matching between the queries and the entities*. In some cases, more than one entities may be relevant for each query. For example, a user searching for “Palermo” may intend to search for the city or the province. We also found cases where duplicate entries were found in the EC (often found in Place entities). For example, “Izola” (a town in Slovenia) was described in two entities. Since the test dataset only allows one entity to be linked for each user query, retrieving multiple entries in the suggestion results may cause the rank of the correct entity to appear to be lower.

Another issue identified is due to the *auto-suggest support for multilingual queries*. To enable retrieval across languages, the auto-suggest function currently retrieves candidate queries by matching user queries against entity labels in all available languages in the EC. In some cases, however, this method also decreased the precision due to finding items that are not relevant to user’s queries or preferred language. E.g., users using Dutch language for search, when typing “Maa” with the intention to look for “Maastricht”, retrieved the following suggestion “Maatalous” (a Finnish term for “Agriculture”) and “Maalikunst” (an Estonian term for “Painting”) in higher ranks.

5 Discussion

In this section, we consider the research questions and highlight insights gained through experimentation.

(RQ1) *Can the knowledge graph (i.e., extracted from Geonames, DBpedia and Wikidata) be utilized to perform auto-suggestion within the cultural heritage domain?* We found that the current coverage of the EC extracted from these sources satisfied less than half of Europeana’s entity queries. The remaining out-of-scope entity queries come from genealogy searches, and queries on infrequent entities, e.g., fashion brands, modern artists, history photographs of politicians, which are currently not available in the EC. Our findings also show that the chosen linked data sources have low multilingual availability for the entity labels. As a result, the language filter cannot be used effectively using the current EC, as the recall drops dramatically due to the poor language coverage on entity labels. However, the rapid development on Wikidata may address this data gap in the near future.

(RQ2) Does the auto-suggestion feature (i.e., utilizing Europeana relevance and Wikidata PageRank) effectively support users in building their search queries? Our results indicate that the combination of Europeana relevance and Wikidata PageRank can support the auto-suggestion functionality to retrieve entities that are relevant to users' queries with high accuracy, regardless of the entity type. However, a bigger dataset containing more queries for each entity type and language is needed to further investigate these results. The use of Wikidata rank also overcomes the cold-start problem, due to the lack of user model in the Europeana site to populate the ranking of the items. In some cases, the ranking of the relevant entities can be further improved. A learning-to-rank approach will be investigated in the future work to enable the personalization of auto-suggestion.

6 Conclusion and Future Work

In this study, we have reported on the auto-suggestion service implemented in the Europeana.eu web portal, which uses an entity ranking algorithm combining Wikidata page rank and Europeana relevance scores. Initial results indicate that the EC was able to match 45.7% Entity queries available in a sample of Europeana.eu access logs. This suggests that more work is needed to improve the coverage of the Entity Collection by integrating more knowledge sources, like increasing the size of classification schemes or adding more entity types, like historical events or popular works (e.g. named paintings, sculptures, books, etc.). However, for cases where the entities do exist in the EC, the auto-suggestion functionality was able to suggest entities with high success rate (SR@10) and good ranking (MRR@10) after users typed 5 characters, regardless of the entity types. Overall the experimental results indicate that the Europeana knowledge graph provides effective support for building more precise search queries and improving the multilingual retrieval on Europeana.eu.

Future research will focus on improving the accuracy of suggestions by enabling filtering options (e.g., by Entity Type, or by usage in Europeana) and prioritizing suggestions that match the user's preferred language (e.g., the language selected for the UI). Building a personalized recommendation based on a user model will also be employed by evaluating the effective usage of the auto-suggestion service (e.g., by recording original query, clicked entity and its ranking).

References

1. Bar-Yossef, Z., Kraus, N.: Context-sensitive query auto-completion. In: Proceedings of the 20th International Conference on World Wide Web, WWW 2011, pp. 107–116. ACM, New York (2011). <http://doi.acm.org/10.1145/1963405.1963424>
2. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.* **30**(1), 107–117 (1998). <http://www.sciencedirect.com/science/article/pii/S016975529800110X>. Proceedings of the Seventh International World Wide Web Conference

3. Cai, F., De Rijke, M., et al.: A survey of query auto completion in information retrieval. *Found. Trends® Inf. Retr.* **10**(4), 273–363 (2016)
4. Ceccarelli, D., Gordea, S., Lucchese, C., Nardini, F.M., Perego, R.: When entities meet query recommender systems: semantic search shortcuts. In: *Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC 2013, Coimbra, Portugal, 18–22 March 2013*, pp. 933–938 (2013). <http://doi.acm.org/10.1145/2480362.2480540>
5. Diefenbach, D., Thalhammer, A.: PageRank and generic entity summarization for RDF knowledge bases. In: Gangemi, A., et al. (eds.) *ESWC 2018*. LNCS, vol. 10843, pp. 145–160. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_10. https://2018.eswc-conferences.org/wp-content/uploads/2018/02/ESWC2018_paper_108.pdf
6. Elhadad, M., Gabay, D., Netzer, Y.: Automatic evaluation of search ontologies in the entertainment domain using text classification. In: *Applied Semantic Technologies: Using Semantics in Intelligent Information Processing*, pp. 351–367 (2011)
7. Eom, Y., Aragón, P., Laniado, D., Kaltenbrunner, A., Vigna, S., Shepelyansky, D.: Interactions of cultures and top people of Wikipedia from ranking of 24 language editions. *CoRR abs/1405.7183* (2014). <http://arxiv.org/abs/1405.7183>
8. Hearst, M.: *Search User Interfaces*. Cambridge University Press, Cambridge (2009)
9. Isaac, A., Manguinhas, H., Charles, V., Stiller, J.: Task force on evaluation and enrichment: selecting target datasets for semantic enrichment (2015). https://pro.europeana.eu/files/Europeana_Professional/EuropeanaTech/EuropeanaTech_taskforces/Enrichment_Evaluation/EvaluationEnrichment_SelectingDatasets_102015.pdf. Accessed 2 May 2018
10. Lika, B., Kolomvatsos, K., Hadjiefthymiades, S.: Facing the cold start problem in recommender systems. *Expert Syst. Appl.* **41**(4, Part 2), 2065–2073 (2014). <http://www.sciencedirect.com/science/article/pii/S0957417413007240>
11. Paramita, M., Clough, P., Hill, T.: D6.3: search improvement report (2017). https://pro.europeana.eu/files/Europeana_Professional/Projects/Project_list/Europeana_DSI-2/Deliverables/d6.3-search-improvement-report.pdf. Accessed 2 May 2018
12. Prasad, S.N.V.A.S.R.K., Gupta, K.G., Manasa, M.: Lightweight multilingual named entity resource extremely extraction and linking using page rank and semantic graphs. *IJSRCSEIT* **2**(4), 182–188 (2017)
13. Schreiber, G., et al.: Semantic annotation and search of cultural-heritage collections: the MultimediaN E-Culture demonstrator. *Web Semant. Sci. Serv. Agents World Wide Web* **6**(4), 243–249 (2008)
14. Shokouhi, M.: Learning to personalize query auto-completion. In: *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2013*, pp. 103–112. ACM, New York (2013). <http://doi.acm.org/10.1145/2484028.2484076>
15. Whiting, S., Jose, J.M.: Recent and robust query auto-completion. In: *Proceedings of the 23rd International Conference on World Wide Web, WWW 2014*, pp. 971–982. ACM, New York (2014). <http://doi.acm.org/10.1145/2566486.2568009>
16. Zhang, L., Färber, M., Rettinger, A.: XKnowSearch!: exploiting knowledge bases for entity-based cross-lingual information retrieval. In: *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 2425–2428. ACM (2016)



A Deep Learning Approach to Fake News Detection

Elio Masciari^(✉), Vincenzo Moscato, Antonio Picariello, and Giancarlo Sperli

University Federico II of Naples, Naples, Italy
{elio.masciari,vincenzo.moscato,antonio.picariello,
giancarlo.sperli}@unina.it

Abstract. The uncontrolled growth of fake news creation and dissemination we observed in recent years causes continuous threats to democracy, justice, and public trust. This problem has significantly driven the effort of both academia and industries for developing more accurate fake news detection strategies. Early detection of fake news is crucial, however the availability of information about news propagation is limited. Moreover, it has been shown that people tend to believe more fake news due to their features [11]. In this paper, we present our framework for fake news detection and we discuss in detail a solution based on deep learning methodologies we implemented by leveraging Google Bert features. Our experiments conducted on two well-known and widely used real-world datasets suggest that our method can outperform the state-of-the-art approaches and allows fake news accurate detection, even in the case of limited content information.

1 Introduction

Social media are nowadays the main medium for large-scale information sharing and communication and they can be considered the main drivers of the Big Data revolution we observed in recent years [1]. Unfortunately, due to malicious user having fraudulent goals fake news on social media are growing quickly both in volume and their potential influence thus leading to very negative social effects. In this respect, identifying and moderating fake news is a quite challenging problem. Indeed, fighting fake news in order to stem their extremely negative effects on individuals and society is crucial in many real life scenarios. Therefore, fake news detection on social media has recently become an hot research topic both for academia and industry.

Fake news detection dates back long time ago [12], for a very long time journalist and scientists fought against misinformation, however, the pervasive use of internet for communication allows for a quicker spread of false information. Indeed, the term *fake news* has grown in popularity in recent years, especially after the 2016 United States elections but there is still no standard definition of fake news [9].

Aside the definition that can be found in literature, one of the most well accepted definition of fake news is the following: *Fake news is a news article that*

is intentionally and verifiable false and could mislead readers [2]. There are two key features of this definition: authenticity and intent. First, fake news includes false information that can be verified as such. Second, fake news is created with dishonest intention to mislead consumers [9].

The content of fake news exhibits heterogeneous topics, styles and media platforms, it aims to mystify truth by diverse linguistic styles while insulting true news. Fake news are generally related to newly emerging, time-critical events, which may not have been properly verified by existing knowledge bases due to the lack of confirmed evidence or claims. Thus, fake news detection on social media poses peculiar challenges due to the inherent nature of social networks that requires both the analysis of their content [6,8] and their social context [3,10].

Indeed, as mentioned above fake news are written on purpose to deceive readers to believe false information. For this reason, it is quite difficult to detect a fake news analysing only the news content. Therefore, we should take into account auxiliary information, such as user social engagement on social media to improve the detection accuracy. Unfortunately, the usage of auxiliary information is a non-trivial task as users social engagements with fake news produce data that are big, noisy, unstructured and incomplete.

Moreover, the diffusion models for fake news changed deeply in recent years. Indeed, as mentioned above, some decades ago, the only medium for information spreading were newspapers and radio/television but recently, the phenomenon of fakes news generation and diffusion take advantage of the internet pervasive diffusion and in particular of social media quick pick approach to news spreading. More in detail, user consumption behaviours have been affected by the inherent nature of these social media platforms: 1) They are more pervasive and less expensive when compared to traditional news media, such as television or newspapers; 2) It is easier to share, comment on, and discuss the news with friends and followers on social media by overcoming geographical and social barrier.

Despite the above mentioned advantages of social media news sharing, there are many drawbacks. First of all, the quality of news on social media is lower than traditional news organizations due to the lower control of information sources. Moreover, since it is cheaper to provide news online and much faster and easier to spread through social media, larger volumes of fake news are produced online for a variety of purposes, such as political gain and unfair competition to cite a few.

Our Approach in a Nutshell. Fake news detection problem can be formalized as a classification task thus requiring features extraction and model construction. The detection phase is a crucial task as it is devoted to guarantee users to receive authentic information. We will focus on finding clues from news contents.

Our goal is to improve the existing approaches defined so far when fake news is intentionally written to mislead users by mimicking true news. More in detail, traditional approaches are based on verification by human editors and expert journalists but do not scale to the volume of news content that is generated in online social networks. As a matter of fact, the huge amount of data to be

analyzed calls for the development of new computational techniques. It is worth noticing that, such computational techniques, even if the news is detected as fake, require some sort of expert verification before being blocked. In our framework, we perform an accurate pre-processing of news data and then we apply three different approaches. The first approach is based on classical classification approaches. We also implemented a deep learning approach that leverages neural network features for fake news detection. Finally, for the sake of completeness we implemented some multimedia approaches in order to take into account misleading images. Due to space limitation, we discuss in this paper the deep learning approach.

2 Our Fake News Detection Framework

Our framework is based on news flow processing and data management in a pre-processing block which performs filtering and aggregation operation over the news content. Moreover, filtered data are processed by two independent blocks: the first one performs natural language processing over data while the second one performs a multimedia analysis.

The overall process we execute for fake news detection is depicted in Fig. 1.

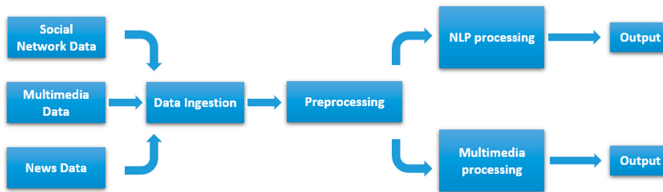


Fig. 1. The overall process at a glance

In the following, we describe each module in more detail.

Data Ingestion Module. This module take care of data collection tasks. Data can be highly heterogeneous: social network data, multimedia data and news data. We collect the news text and eventual related contents and images.

Pre-processing Module. This component is devoted to the acquisition of the incoming data flow. It performs filtering, data aggregation, data cleaning and enrichment operations.

NLP Processing Module. It performs the crucial task of generating a binary classification of the news articles, i.e., whether they are fake or reliable news. It is split in two submodules. The *Machine Learning* module performs classification using an ad-hoc implemented Logistic Regression algorithm after an extensive process of feature extraction and selection TF-IDF based in order to reduce the number of extracted features. The *Deep Learning* module classify data using

Google Bert algorithm after a tuning phase on the vocabulary. It also perform a binary transformation and eventual text padding in order to better analyze the input data.

Multimedia Processing Module. This module is tailored for Fake Image Classification through Deep Learning algorithms, using ELA (Error Level Analysis) and CNN.

Due to space limitation, we discuss in the following only the details of the deep learning module and the obtained results.

2.1 The Software Architecture

The software implementation of the framework described above is shown in Fig. 2.

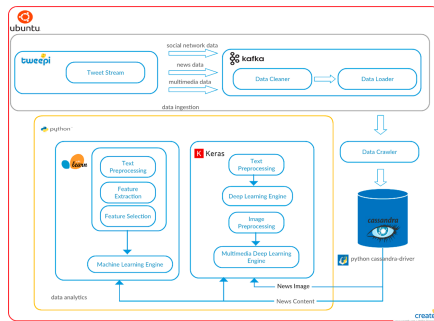


Fig. 2. Our fake news detection framework

Herein: the data ingestion block is implemented by using several tools. As an example for Twitter data we leverage Tweepy, a Python library to access the Twitter API. All tweets are downloaded through this library. Filtering and aggregation is performed using Apache Kafka which is able to build real-time data pipelines and streaming apps. It is scalable, fault-tolerant and fast thus making our prototype well-suited for huge amount of data.

The data crawler uses the Newspaper Python library whose purpose is extracting and curating articles. The analytical data archive stores pre-processed data that are used for issuing queries by traditional analytical tools. We leverage Apache Cassandra as datastore because it provides high scalability, high availability, fast writing, fault- tolerance on commodity hardware or cloud infrastructure. The data analytics block retrieves news contents and news images from Cassandra DB that are pre-processed by the Machine Learning module using Scikit Learn library and by Deep Learning module using Keras library. Image content is processed by the Multimedia Deep Learning module using Keras library.

In the following, we will briefly describe how the overall process is executed. Requests to the Cassandra DB are made through remote access. Each column

in Cassandra refers to a specific topic and contains all news belonging to that topic. Among all news, those having a valid *external link* value are selected. In this way, the news content can be easily crawled. As the link for each news is obtained, a check is performed in order to verify the current state of the website. If the website is still running, we perform the article scraping. The algorithm works by downloading and parsing the news article, then, for each article, title, text, authors, top image link, news link data are extracted and saved as a JSON file in Cassandra DB.

Finally, three independent analysis are then performed by three ad-hoc Python modules we implemented. The first two perform text classification, and the last one images classification. Concerning the text analysis, the problem being solved is a binary classification one where class 0 refers to reliable news and class 1 refers to fake ones.

2.2 The Deep Learning Module

The Deep Learning Module computes a binary classification on a text datasets of news that will be labelled as 0 if a news is marked as Real, and as 1 if it is marked as Fake. The Deep Learning Module classifies news content using a new language model called *B.E.R.T.* (Bidirectional Encoder Representations from Transformers) developed and released by Google. Prior to describing the algorithm features in detail, we briefly describe the auxiliary tools being used, while in Sect. 3 we describe the experimental evaluation that lead to our choice on BERT.

Colaboratory. Colab¹ is intended for machine learning education and research, it requires no setup and runs entirely on the cloud. By using Colab it's possible to write and execute code, save and share analytics and it provides access to expensive and powerful computing resources for free by a web interface.

More in detail, Colab's hardware is powered by: Intel(R) Xeon(R) CPU @ 2.00 GHz, nVidia T4 16GB GDDR6 @ 300 GB/sec, 15 GB RAM and 350 GB storage. This setting is able to speed-up the learning task execution up to 35X and 16X faster in deep learning training compared to a CPU-only server.

Tensor Flow. It is devoted to train and run neural networks for image recognition, word embeddings, recurrent neural networks, and natural language processing. It is a cross-platform tool and runs on CPUs, GPUs, even on mobile and embedded platforms. TensorFlow uses dataflow graphs to represent the computation flow, i.e., these structures describe the data flow through the processing nodes. Each node in the graph represents a mathematical operation, and each connection between nodes is a multidimensional data array called tensor. The TensorFlow Distributed Execution Engine abstracts from the supported devices and provides a high performance-core implemented in C++ for the TensorFlow platform. On top there are Python and C++ frontends. The Layers API provides a simple interface for most of the layers used in deep learning models. Finally,

¹ <https://research.google.com/colaboratory>.

higher-level APIs, including Keras, makes training and evaluating distributed models easier.

Keras. It is a high-level neural network API, implemented in Python and capable of running on top of TensorFlow. It allows for easy and fast prototyping through: 1) User Friendliness as it offers consistent and simple APIs that minimizes the number of user actions required for common use cases; 2) Modularity as neural layers, cost functions, optimizers, initialization schemes, activation functions and regularization schemes are all standalone modules that can be combined to create new models; 3) Extensibility as new modules are simple to add as new classes and functions.

Google BERT. This tool has been developed in order to allow an easier implementation of two crucial tasks for Natural Language Processing (NLP): Transfer Learning through unsupervised pre-training and Transformer architecture. The idea behind Transfer Learning is to train a model in a given domain on a large text corpus, and then leverage the gathered knowledge to improve the model's performance in a different domain. In this respect, BERT has been pre-trained on Wikipedia and BooksCorpus. On the opposite side, the Transformer architecture processes all elements simultaneously by linking individual elements through a process known as attention. This mechanism allows a deep parallelization and guarantee higher accuracy across a wide range of tasks. BERT outperforms previous proposed approaches as it is the first unsupervised, fully bidirectional system for NLP pre-training. Pre-trained representations can be either context-free or context based dependig on user needs. Due to space limitations we do not describe in detail the BERT's architecture and the encoder mechanism.

3 Our Benchmark

In this section we will describe the fake news detection process for the deep learning module and the datasets we used as a benchmark for our algorithms.

3.1 Dataset Description

Liar Dataset. This dataset includes 12.8K human labelled short statements from fact-checking website [Politifact.com](https://www.politifact.com). Each statement is evaluated by a [Politifact.com](https://www.politifact.com) editor for its truthfulness. The dataset has six fine-grained labels: pants-fire, false, barely-true, half-true, mostly-true, and true. The distribution of labels is relatively well-balanced [12]. For our purposes the six fine-grained labels of the dataset have been collapsed in a binary classification, i.e., label 1 for fake news and label 0 for reliable ones. This choice has been made due to binary Fake News Dataset feature. The dataset is partitioned into three files: 1) Training Set: 5770 real news and 4497 fake news; 2) Test Set: 1382 real news and 1169 fake news; 3) Validation Set: 1382 real news and 1169 fake news. The three subsets are well balanced so there is no need to perform oversampling or undersampling.

The processed dataset has been uploaded in Google Drive and, then, loaded in Colab's Jupyter as a Pandas Dataframe. It has been added a new column with the number of words for each row article. Using the command `df.describe()` on this column it is possible to print the following statistical information: count 15389.000000, mean 17.962311, std 8.569879, min 1.000000, 25% 12.000000, 50% 17.000000, 75% 22.000000, max 66.000000. These statistics show that there are articles with only one word in the dataset, so it has been decided to remove all rows with less than 10 words as they are considered poorly informative. The resulting dataset contains 1657 less rows than the original one. The updated statistics are reported in what follows: count 13732.000000, mean 19.228663, std 8.192268, min 10.000000, 25% 14.000000, 50% 18.000000, 75% 23.000000, max 66.000000. Finally, the average number of words per article is 19.

FakeNewsNet. This dataset has been built by gathering information from two fact-checking websites to obtain news contents for fake news and real news such as PolitiFact and GossipCop. In PolitiFact, journalists and domain experts review the political news and provide fact-checking evaluation results to claim news articles as fake or real. Instead, in GossipCop, entertainment stories, from various media outlets, are evaluated by a rating score on the scale of 0 to 10 as the degree from fake to real. The dataset contains about 900 political news and 20k gossip news and has only two labels: true and false [14].

This dataset is publicly available by the functions provided by the FakeNewsNet team and the Twitter API. As mentioned above, FakeNewsNet can be split in two subsets: GossipCop and [Politifact.com](https://www.politifact.com). We decided to analyse only political news as they produce worse consequences in real world than gossip ones. The dataset is well balanced and contains 434 real news and 367 fake news. Most of the news regards the US as it has already been noticed in LIAR. Fake news topics concern Obama, police, Clinton and Trump while real news topics refer to Trump, Republicans and Obama. Such as the LIAR dataset, it has been added a new column and used the command `df.describe()` to print out the following statistical information: count 801, mean 1459.217228, std 3141.157565, min 3, 25% 114, 50% 351, 75% 893, max 17377.

The average number of words per articles in Politifact dataset is 1459, which is far longer than the average sentence length in Liar Dataset that is 19 words per articles. Such a statistics suggested us to compare the model performances on datasets with such different features.

3.2 Pre-elaboration Steps

The above mentioned datasets are available in CSV format and are composed of two columns: text and label. The news text need to be pre-processed for our analysis. In this respect, an ad-hoc Python function has been developed for unnecessary IP and URL addresses removal, HTML tags checking and words spell-check. Due to neural features, we decide to maintain some stop words in order to allow a proper context analysis. Thus, to ameliorate the noise problem, we created a custom list of stop words. We leverage Keras Tokenizer for preparing

text documents for subsequent deep learning steps. More in detail, we create a vocabulary index based on word frequency, e.g., given the sentence *The cat sat on the mat* we create the following dictionary $word_index[the] = 1$, $word_index[cat] = 2$ so every word gets a unique integer value; the 0 value is reserved for padding. Lower integer means more frequent word. After this encoding step, we obtain for each text a sequence of integers. As BERT needs a more elaborated input than other neural networks we need to produce a *tsv* file, with four columns, and no header. The columns to be added to dataset are: 1) *guid*, i.e., a row ID; 2) *label*, i.e., the label for the row (it should be an int); 3) *alpha*, a dummy column containing the same letter for all rows, it is not used for classification but it is needed for proper running of the algorithm and 4) *text*, i.e., the news content. The data needs to be converted in InputFeature object to be compatible with Transformer Architecture. The conversion process includes tokenization and converting all sentences to a given sequence length (truncating longer sequences, and padding shorter sequences). Tokenization is performed using WordPiece tokenization, where the vocabulary is initialized with all the individual characters in the language, and then the most frequent/likely combinations of the existing words in the vocabulary are iteratively added. Words that does not occur in the vocabulary are broken down into sub-words in order to search for possible matches in the collection.

4 Evaluation

In order to show that the Google BERT model we implemented outperforms the results of the current performance state of art both on Liar dataset and Polifact dataset, we report in Figs. 3 and 4 the best results obtained for the other approaches commonly used in literature for those datasets.

Neural Network	Accuracy	Precision	Recall	F1	TP	FP	TN	FN	AUC
BERT	0.619	0.583	0.596	0.628	697	472	884	498	0.617
C-HAN	0.557	0.514	0.628	0.565	688	694	735	434	0.574
Bi-LSTM	0.586	0.554	0.495	0.523	579	465	917	590	0.607
CNN	0.536	0.489	0.275	0.352	1046	847	322	336	0.539

Fig. 3. Comparison for Google BERT against state of the art approaches on LIAR dataset

We compared the performances on well-established evaluation measure like: Accuracy, Precision, Recall, F1 measure, Area Under Curve (AUC) [5] and the values reported in the obtained confusion matrices for each algorithm, i.e., True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN).

We hypothesize that our results are quite better due to a fine hyper parameter tuning we performed, a better pre-processing step and the proper transformation.

For the sake of completeness, we report in Figs. 5 and 6 the detailed confusion matrices obtained for LIAR and Polifact datasets.

Neural Network	Accuracy	Precision	Recall	F1	TP	FP	TN	FN	AUC
BERT	0.588	0.565	0.449	0.628	165	127	306	202	0.578
C-HAN	0.448	0.443	0.795	0.569	67	75	292	367	0.436
Bi-LSTM	0.511	0.466	0.455	0.460	167	200	243	191	0.532
CNN	0.540	0.498	0.405	0.447	284	218	149	150	0.520

Fig. 4. Comparison for Google BERT against state of the art approaches on Polifact dataset

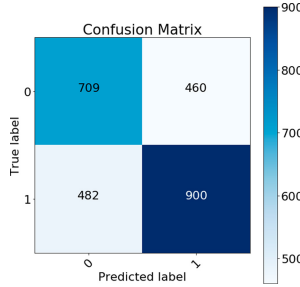


Fig. 5. Confusion Matrix for LIAR dataset

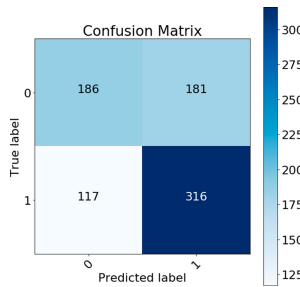


Fig. 6. Confusion matrix for polifact dataset

5 Conclusion and Future Work

In this paper, we investigated the problem of fake news detection by deep learning algorithms. We developed a framework the leverage Google BERT for analyzing real-life datasets and the results we obtained are quite encouraging. As for future work, we would like to extend our analysis by considering also user profiles' features and some kind of dynamic analysis of news diffusion mechanism in our fake news detection model [7].

Acknowledgement. Elio Masciari has been supported by POR Calabria project SPIDASEC.

References

1. Agrawal, D., et al.: Challenges and opportunities with big data. A community white paper developed by leading researchers across the United States. Technical report, Purdue University, March 2012
2. Allcott, H., Gentzkow, M.: Social media and fake news in the 2016 election. Working Paper 23089, National Bureau of Economic Research, January 2017
3. Cassavia, N., Masciari, E., Pulice, C., Saccà, D.: Discovering user behavioral features to enhance information search on big data. *TiiS* **7**(2), 7:1–7:33 (2017)
4. Culpepper, J.S., Moffat, A., Bennett, P.N., Lerman, K. (eds.) Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, 11–15 February , 2019. ACM (2019)
5. Flach, P.A., Kull, M.: Precision-recall-gain curves: PR analysis done right. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (eds.) *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015*, Montreal, Quebec, Canada, 7–12 December 2015, pp. 838–846 (2015)
6. Guo, C., Cao, J., Zhang, X., Shu, K., Yu., M.: Exploiting emotions for fake news detection on social media. *CoRR*, abs/1903.01728 (2019)
7. Masciari, E.: SMART: stream monitoring enterprise activities by RFID tags. *Inf. Sci.* **195**, 25–44 (2012)
8. Potthast, M., Kiesel, J., Reinartz, K., Bevendorff, J., Stein, B.: A stylometric inquiry into hyperpartisan and fake news. *CoRR*, abs/1702.05638 (2017)
9. Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H.: Fake news detection on social media: a data mining perspective. *CoRR*, abs/1708.01967 (2017)
10. Shu, K., Wang, S., Liu, H.: Beyond news contents: the role of social context for fake news detection. In: Culpepper et al. [4], pp. 312–320
11. Vosoughi, S., Roy, D., Aral, S.: The spread of true and false news online. *Science* **359**(6380), 1146–1151 (2018)
12. Zhou, X., Zafarani, R., Shu, K., Liu, H.: Fake news: fundamental theories, detection strategies and challenges. In: Culpepper et al. [4], pp. 836–837



Satirical News Detection with Semantic Feature Extraction and Game-Theoretic Rough Sets

Yue Zhou¹, Yan Zhang^{1(✉)}, and JingTao Yao²

¹ School of Computer Science and Engineering, California State University, San Bernardino, San Bernardino, CA, USA

{Yue.Zhou,Yan.Zhang}@csusb.edu

² Department of Computer Science, University of Regina, Regina, SK, Canada
jtyao@cs.uregina.ca

Abstract. Satirical news detection is an important yet challenging task to prevent spread of misinformation. Many feature based and end-to-end neural nets based satirical news detection systems have been proposed and delivered promising results. Existing approaches explore comprehensive word features from satirical news articles, but lack semantic metrics using word vectors for tweet form satirical news. Moreover, the vagueness of satire and news parody determines that a news tweet can hardly be classified with a binary decision, that is, satirical or legitimate. To address these issues, we collect satirical and legitimate news tweets, and propose a semantic feature based approach. Features are extracted by exploring inconsistencies in phrases, entities, and between main and relative clauses. We apply game-theoretic rough set model to detect satirical news, in which probabilistic thresholds are derived by game equilibrium and repetition learning mechanism. Experimental results on the collected dataset show the robustness and improvement of the proposed approach compared with Pawlak rough set model and SVM.

Keywords: Satirical news detection · Social media · Feature extraction · Game-theoretic rough sets

1 Introduction

Satirical news, which uses parody characterized in a conventional news style, has now become an entertainment on social media. While news satire is claimed to be pure comedic and of amusement, it makes statements on real events often with the aim of attaining social criticism and influencing change [15]. Satirical news can also be misleading to readers, even though it is not designed for falsifications. Given such sophistication, satirical news detection is a necessary yet challenging natural language processing (NLP) task. Many feature based fake or satirical news detection systems [3, 11, 14] extract features from word relations given by statistics or lexical database, and other linguistic features. In addition, with

the great success of deep learning in NLP in recent years, many end-to-end neural nets based detection systems [6, 12, 16] have been proposed and delivered promising results on satirical news article detection.

However, with the evolution of fast-paced social media, satirical news has been condensed into a satirical-news-in-one-sentence form. For example, one single tweet of “If earth continues to warm at current rate moon will be mostly underwater by 2400” by The Onion is largely consumed and spread by social media users than the corresponding full article posted on The Onion website. Existing detection systems trained on full document data might not be applicable to such form of satirical news. Therefore, we collect news tweets from satirical news sources such as The Onion, The New Yorker (Borowitz Report) and legitimate news sources such as Wall Street Journal and CNN Breaking News. We explore the syntactic tree of the sentence and extract inconsistencies between attributes and head noun in noun phrases. We also detect the existence of named entities and relations between named entities and noun phrases as well as contradictions between the main clause and corresponding prepositional phrase. For a satirical news, such inconsistencies often exist since satirical news usually combines irrelevant components so as to attain surprise and humor. The discrepancies are measured by cosine similarity between word components where words are represented by Glove [9]. Sentence structures are derived by Flair, a state-of-the-art NLP framework, which better captures part-of-speech and named entity structures [1].

Due to the obscurity of satire genre and lacks of information given tweet form satirical news, there exists ambiguity in satirical news, which causes great difficulty to make a traditional binary decision. That is, it is difficult to classify one news as satirical or legitimate with available information. Three-way decisions, proposed by YY Yao, added an option - deferral decision in the traditional yes-and-no binary decisions and can be used to classify satirical news [21, 22]. That is, one news may be classified as satirical, legitimate, and deferral. We apply rough sets model, particularly the game-theoretic rough sets to classify news into three groups, i.e., satirical, legitimate, and deferral. Game-theoretic rough set (GTRS) model, proposed by JT Yao and Herbert, is a recent promising model for decision making in the rough set context [18]. GTRS determine three decision regions from a tradeoff perspective when multiple criteria are involved to evaluate the classification models [25]. Games are formulated to obtain a tradeoff between involved criteria. The balanced thresholds of three decision regions can be induced from the game equilibria. GTRS have been applied in recommendation systems [2], medical decision making [19], uncertainty analysis [24], and spam filtering [23].

We apply GTRS model on our preprocessed dataset and divide all news into satirical, legitimate, or deferral regions. The probabilistic thresholds that determine three decision regions are obtained by formulating competitive games between accuracy and coverage and then finding Nash equilibrium of games. We perform extensive experiments on the collected dataset, fine-tuning the model by different discretization methods and variation of equivalent classes. The

experimental result shows that the performance of the proposed model is superior compared with Pawlak rough sets model and SVM.

2 Related Work

Satirical news detection is an important yet challenging NLP task. Many feature based models have been proposed. Burfoot et al. extracted features of headline, profanity, and slang using word relations given by statistical metrics and lexical database [3]. Rubin et al. proposed a SVM based model with five features (absurdity, humor, grammar, negative affect, and punctuation) for fake news document detection [11]. Yang et al. presented linguistic features such as psycholinguistic feature based on dictionary and writing stylistic feature from part-of-speech tags distribution frequency [17]. Shu et al. gave a survey in which a set of feature extraction methods is introduced for fake news on social media [14]. Conroy et al. also uses social network behavior to detect fake news [4]. For satirical sentence classification, Davidov et al. extract patterns using word frequency and punctuation features for tweet sentences and amazon comments [5]. The detection of a certain type of sarcasm which contracts positive sentiment with a negative situation by analyzing the sentence pattern with a bootstrapped learning was also discussed [10]. Although word level statistical features are widely used, with advanced word representations and state-of-the-art part-of-speech tagging and named entity recognition model, we observe that semantic features are more important than word level statistical features to model performance. Thus, we decompose the syntactic tree and use word vectors to more precisely capture the semantic inconsistencies in different structural parts of a satirical news tweet.

Recently, with the success of deep learning in NLP, many researchers attempted to detect fake news with end-to-end neural nets based approaches. Ruchansky et al. proposed a hybrid deep neural model which processes both text and user information [12], while Wang et al. proposed a neural network model that takes both text and image data [16] for detection. Sarkar et al. presented a neural network with attention to both capture sentence level and document level satire [6]. Some research analyzed sarcasm from non-news text. Ghosh and Veale [7] used both the linguistic context and the psychological context information with a bi-directional LSTM to detect sarcasm in users' tweets. They also published a feedback-based dataset by collecting the responses from the tweets authors for future analysis. While all these works detect fake news given full text or image content, or target on non-news tweets, we attempt bridge the gap and detect satirical news by analyzing news tweets which concisely summarize the content of news.

3 Methodology

In this section, we will describe the composition and preprocessing of our dataset and introduce our model in detail. We create our dataset by collecting legitimate and satirical news tweets from different news source accounts. Our model aims

to detect whether the content of a news tweet is satirical or legitimate. We first extract the semantic features based on inconsistencies in different structural parts of the tweet sentences, and then use these features to train game-theoretic rough set decision model.

3.1 Dataset

We collected approximately 9,000 news tweets from satirical news sources such as The Onion and Borowitz Report and about 11,000 news tweets from legitimate news sources such as Wall Street Journal and CNN Breaking News over the past three years. Each tweet is a concise summary of a news article. The duplicated and extreme short tweets are removed. A news tweet is labeled as satirical if it is written by satirical news sources and legitimate if it is from legitimate news sources. Table 1 gives an example of tweet instances that comprise our dataset.

Table 1. Examples of instances comprising the news tweet dataset

Content	Source	Label
The White House confirms that President Donald Trump sent a letter to North Korean leader Kim Jong Un	CNN	0
Illinois Senate plans vote on bills that could become the state's first budget in more than two years	WSJ	0
Naked Andrew Yang emerges from time vortex to warn debate audience about looming threat of automation	TheOnion	1
New study shows majority of late afternoon sleepiness at work caused by undetected carbon monoxide leak	TheOnion	1
Devin Nunes accuses witnesses of misleading American people with facts	BorowitzReport	1

3.2 Semantic Feature Extraction

Satirical news is not based on or does not aim to state the fact. Rather, it uses parody or humor to make statement, criticisms, or just amusements. In order to achieve such effect, contradictions are greatly utilized. Therefore, inconsistencies significantly exist in different parts of a satirical news tweet. In addition, there is a lack of entity or inconsistency between entities in news satire. We extracted these features at semantic level from different sub-structures of the news tweet.

Different structural parts of the sentence are derived by part-of-speech tagging and named entity recognition by Flair. The inconsistencies in different structures are measured by cosine similarity of word phrases where words are represented by Glove word vectors. We explored three different aspects of inconsistency and designed metrics for their measurements. A word level feature using tf-idf [13] is added for robustness.

Inconsistency in Noun Phrase Structures. One way for a news satire to obtain surprise or humor effect is to combine irrelevant or less jointly used attributes and the head noun which they modified. For example, noun phrase such as “rampant accountability”, “posthumous apology”, “self-imposed mental construct” and other rare combinations are widely used in satirical news, while individual words themselves are common. To measure such inconsistency, we first select all leaf noun phrases (NP) extracted from the trees to avoid repeated calculation. Then for each noun phrase, each adjacent word pair is selected and represented by 100-dim Glove word vector denoted as (v_t, w_t) . We define the averaged cosine similarity of noun phrase word pairs as:

$$S_{NP} = \frac{1}{T} \sum_{t=1}^T \cos(v_t, w_t) \quad (1)$$

where T is a total number of word pairs. We use S_{NP} as a feature to capture the overall inconsistency in noun phrase uses. S_{NP} ranges from -1 to 1 , where a smaller value indicates more significant inconsistency.

Inconsistency Between Clauses. Another commonly used rhetoric approach for news satire is to make contradiction between the main clause and its prepositional phrase or relative clause. For instance, in the tweet “Trump boys counter Chinese currency manipulation *by* adding extra zeros to \$20 Bills”, contradiction or surprise is gained by contrasting irrelevant statements provided by different parts of the sentence. Let q and p denote two clauses separated by main/relative relation or preposition, and (w_1, w_1, \dots, w_q) and (v_1, v_1, \dots, v_p) be the vectorized words in q and p . Then we define inconsistency between q and p as:

$$S_{QP} = \cos\left(\sum_{q=1}^Q w_q, \sum_{p=1}^P v_p\right) \quad (2)$$

Similarly, the feature S_{QP} is measured by cosine similarity of linear summations of word vectors, where smaller value indicates more significant inconsistency.

Inconsistency Between Named Entities and Noun Phrases. Even though many satirical news tweets are made based on real persons or events, most of them lack specific entities. Rather, because the news is fabricated, news writers use the words such as “man”, “woman”, “local man”, “area woman”, “local

family” as subject. However, the inconsistency between named entities and noun phrases often exists in a news satire if a named entity is included. For example, the named entity “Andrew Yang” and the noun phrases “time vortex” show great inconsistency than “President Trump”, “Senate Republicans”, and “White House” do in the legitimate news “President Trump invites Senate Republicans to the White House to talk about the funding bill.” We define such inconsistency as a categorical feature that:

$$C_{NERN} = \begin{cases} 0 & \text{if } S_{NERN} < \bar{S}_{NERN} \\ 1 & \text{if } S_{NERN} \geq \bar{S}_{NERN} \\ -1 & \text{if there's no named entity} \end{cases} \quad (3)$$

S_{NERN} is the cosine similarity of named entities and noun phrases of a certain sentence and \bar{S}_{NERN} is the mean value of S_{NERN} in corpus.

Word Level Feature Using TF-IDF. We calculated the difference of tf-idf scores between legitimate news corpus and satirical news corpus for each single word. Then, the set S_{voc} that includes most representative legitimate news words is created by selecting top 100 words given the tf-idf difference. For a news tweet and any word w in the tweet, we define the binary feature B_{voc} as:

$$B_{voc} = \begin{cases} 1 & \text{if } w \in S_{voc} \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

3.3 GTRS Decision Model

We construct a Game-theoretic Rough Sets model for classification given the extracted features. Suppose $E \subseteq U \times U$ is an equivalence relation on a finite nonempty universe of objects U , where E is reflexive, symmetric, and transitive. The equivalence class containing an object x is given by $[x] = \{y \in U | xEy\}$. The objects in one equivalence class all have the same attribute values. In the satirical news context, given an undefined concept *satire*, probabilistic rough sets divide all news into three pairwise disjoint groups i.e., the satirical group $POS(\textit{satire})$, legitimate group $NEG(\textit{satire})$, and deferral group $BND(\textit{satire})$, by using the conditional probability $Pr(\textit{satire}|[x]) = \frac{|satire \cap [x]|}{|[x]|}$ as the evaluation function, and (α, β) as the acceptance and rejection thresholds [20–22], that is,

$$\begin{aligned} POS_{(\alpha, \beta)}(\textit{satire}) &= \{x \in U \mid Pr(\textit{satire}|[x]) \geq \alpha\}, \\ NEG_{(\alpha, \beta)}(\textit{satire}) &= \{x \in U \mid Pr(\textit{satire}|[x]) \leq \beta\}, \\ BND_{(\alpha, \beta)}(\textit{satire}) &= \{x \in U \mid \beta < Pr(\textit{satire}|[x]) < \alpha\}. \end{aligned} \quad (5)$$

Given an equivalence class $[x]$, if the conditional probability $Pr(\textit{satire}|[x])$ is greater than or equal to the specified acceptance threshold α , i.e., $Pr(\textit{satire}|[x]) \geq \alpha$, we accept the news in $[x]$ as *satirical*. If $Pr(\textit{satire}|[x])$ is less than or equal to the specified rejection threshold β , i.e., $Pr(\textit{satire}|[x]) \leq \beta$

we reject the news in $[x]$ as *satirical*, or we accept the news in $[x]$ as *legitimate*. If $Pr(satire|[x])$ is between α and β , i.e., $\beta < Pr(satire|[x]) < \alpha$, we defer to make decisions on the news in $[x]$. Pawlak rough sets can be viewed as a special case of probabilistic rough sets with $(\alpha, \beta) = (1, 0)$.

Given a pair of probabilistic thresholds (α, β) , we can obtain a news classifier according to Eq. (5). The three regions are a partition of the universe U ,

$$\pi_{(\alpha, \beta)}(Satire) = \{POS_{(\alpha, \beta)}(Satire), BND_{(\alpha, \beta)}(Satire), NEG_{(\alpha, \beta)}(Satire)\} \quad (6)$$

Then, the accuracy and coverage rate to evaluate the performance of the derived classifier are defined as follows [25],

$$Acc_{(\alpha, \beta)}(Satire) = \frac{|Satire \cap POS_{(\alpha, \beta)}(Satire)| + |Satire^c \cap NEG_{(\alpha, \beta)}(Satire)|}{|POS_{(\alpha, \beta)}(Satire)| + |NEG_{(\alpha, \beta)}(Satire)|} \quad (7)$$

$$Cov_{(\alpha, \beta)}(Satire) = \frac{|POS_{(\alpha, \beta)}(Satire)| + |NEG_{(\alpha, \beta)}(Satire)|}{|U|} \quad (8)$$

The criterion coverage indicates the proportions of news that can be confidently classified. Next, we will obtain (α, β) by game formulation and repetition learning.

Game Formulation. We construct a game $G = \{O, S, u\}$ given the set of game players O , the set of strategy profile S , and the payoff functions u , where the accuracy and coverage are two players, respectively, i.e., $O = \{acc, cov\}$.

The set of strategy profiles $S = S_{acc} \times S_{cov}$, where S_{acc} and S_{cov} are sets of possible strategies or actions performed by players *acc* and *cov*. The initial thresholds are set as $(1, 0)$. All these strategies are the changes made on the initial thresholds,

$$\begin{aligned} S_{acc} &= \{\beta \text{ no change}, \beta \text{ increases } c_{acc}, \beta \text{ increases } 2 \times c_{acc}\}, \\ S_{cov} &= \{\alpha \text{ no change}, \alpha \text{ decreases } c_{cov}, \alpha \text{ decreases } 2 \times c_{cov}\}. \end{aligned} \quad (9)$$

c_{acc} and c_{cov} denote the change steps used by two players, and their values are determined by the concrete experiment date set.

Payoff Functions. The payoffs of players are $u = (u_{acc}, u_{cov})$, and u_{acc} and u_{cov} denote the payoff functions of players *acc* and *cov*, respectively. Given a strategy profile $p = (s, t)$ with player *acc* performing s and player *cov* performing t , the payoffs of *acc* and *cov* are $u_{acc}(s, t)$ and $u_{cov}(s, t)$. We use $u_{acc}(\alpha, \beta)$ and $u_{cov}(\alpha, \beta)$ to show this relationship. The payoff functions $u_{acc}(\alpha, \beta)$ and $u_{cov}(\alpha, \beta)$ are defined as,

$$\begin{aligned} u_{acc}(s, t) &\Rightarrow u_{acc}(\alpha, \beta) = Acc_{(\alpha, \beta)}(Satire), \\ u_{cov}(s, t) &\Rightarrow u_{cov}(\alpha, \beta) = Cov_{(\alpha, \beta)}(Satire), \end{aligned} \quad (10)$$

where $Acc_{(\alpha,\beta)}(Satire)$ and $Cov_{(\alpha,\beta)}(Satire)$ are the accuracy and coverage defined in Eqs. (7) and (8).

Payoff Table. We use payoff tables to represent the formulated game. Table 2 shows a payoff table example in which both players have 3 strategies defined in Eq. (9).

Table 2. An example of a payoff table

		cov		
		α	$\alpha \downarrow c_{cov}$	$\alpha \downarrow 2c_{cov}$
acc	β	$\langle u_{acc}(\alpha, \beta), u_{cov}(\alpha, \beta) \rangle$	$\langle u_{acc}(\alpha - c_{cov}, \beta), u_{cov}(\alpha - c_{cov}, \beta) \rangle$	$\langle u_{acc}(\alpha - 2c_{cov}, \beta), u_{cov}(\alpha - 2c_{cov}, \beta) \rangle$
	$\beta \uparrow c_{acc}$	$\langle u_{acc}(\alpha, \beta + c_{acc}), u_{cov}(\alpha, \beta + c_{acc}) \rangle$	$\langle u_{acc}(\alpha - c_{cov}, \beta + c_{acc}), u_{cov}(\alpha - c_{cov}, \beta + c_{acc}) \rangle$	$\langle u_{acc}(\alpha - 2c_{cov}, \beta + c_{acc}), u_{cov}(\alpha - 2c_{cov}, \beta + c_{acc}) \rangle$
	$\beta \uparrow 2c_{acc}$	$\langle u_{acc}(\alpha, \beta + 2c_{acc}), u_{cov}(\alpha, \beta + 2c_{acc}) \rangle$	$\langle u_{acc}(\alpha - c_{cov}, \beta + 2c_{acc}), u_{cov}(\alpha - c_{cov}, \beta + 2c_{acc}) \rangle$	$\langle u_{acc}(\alpha - 2c_{cov}, \beta + 2c_{acc}), u_{cov}(\alpha - 2c_{cov}, \beta + 2c_{acc}) \rangle$

The arrow \downarrow denotes decreasing a value and \uparrow denotes increasing a value. On each cell, the threshold values are determined by two players.

Repetition Learning Mechanism. We repeat the game with the new thresholds until a balanced solution is reached. We first analyzes the pure strategy equilibrium of the game and then check if the stopping criteria are satisfied.

Game Equilibrium. The game solution of pure strategy Nash equilibrium is used to determine possible game outcomes in GTRS. The strategy profile (s_i, t_j) is a pure strategy Nash equilibrium, if

$$\begin{aligned} \forall s'_i \in S_{acc}, u_{acc}(s_i, t_j) \geq u_{acc}(s'_i, t_j), \text{ where } s_i \in S_{acc} \wedge s'_i \neq s_i, \\ \forall t'_j \in S_{cov}, u_{cov}(s_i, t_j) \geq u_{cov}(s_i, t'_j), \text{ where } t_j \in S_{cov} \wedge t'_j \neq t_j. \end{aligned} \quad (11)$$

This means that none of players would like to change his strategy or they would loss benefit if deriving from this strategy profile, provided this player has the knowledge of other player’s strategy.

Repetition of Games. Assuming that we formulate a game, in which the initial thresholds are (α, β) , and the equilibrium analysis shows that the thresholds corresponding to the equilibrium are (α^*, β^*) . If the thresholds (α^*, β^*) do not satisfy the stopping criterion, we will update the initial thresholds in the subsequent games. The initial thresholds of the new game will be set as (α^*, β^*) . If the thresholds (α^*, β^*) satisfy the stopping criterion, we may stop the repetition of games.

Stopping Criterion. We define the stopping criteria so that the iterations of games can stop at a proper time. In this research, we set the stopping criterion as the thresholds are inside the valid range or the increase of one player’s payoff is less than the decrease of the other player’s payoff.

4 Experiments

There are 8757 news records in our preprocessed data set. We use Jenks natural breaks [8] to discretize continuous variables S_{NP} and S_{QP} both into five categories denoted by nominal values from 0 to 4, where larger values still fall into bins with larger nominal value. Let D_{NP} and D_{QP} denote the discretized variables S_{NP} and S_{QP} , respectively. We derived the information table that only contains discrete features from our original dataset. A fraction of the information table is shown in Table 3.

Table 3. The information table

Id	D_{NP}	D_{QP}	C_{NERN}	B_{voc}	target
1	0	2	0	0	1
2	1	2	0	0	1
3	2	2	0	1	0
4	2	4	1	1	0
5	2	3	0	0	1
6	4	3	-1	1	0
7	2	3	0	0	0
8	3	2	-1	0	1

The news whose condition attributes have the same values are classified in an equivalence class X_i . We derived 149 equivalence classes and calculated the corresponding probability $Pr(X_i)$ and condition probability $Pr(Satire|X_i)$ for each X_i . The probability $Pr(X_i)$ denotes the ratio of the number of news contained in the equivalence class X_i to the total number of news in the dataset, while the conditional probability $Pr(Satire|X_i)$ is the proportion of news in X_i that are satirical. We combine the equivalence classes with the same conditional probability and reduce the number of equivalence classes to 108. Table 4 shows a part of the probabilistic data information about the concept *satire*.

Table 4. Summary of the partial experimental data

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9
$Pr(X_i)$	0.0315	0.0054	0.0026	0.0071	0.0062	0.0018	0.0015	0.0098	0.0009
$Pr(Satire X_i)$	1	0.9787	0.9565	0.9516	0.9444	0.9375	0.9231	0.9186	0.875
	X_{100}	X_{101}	X_{102}	X_{103}	X_{104}	X_{105}	X_{106}	X_{107}	X_{108}
$Pr(X_i)$	0.0121	0.0138	0.0095	0.0065	0.0383	0.0078	0.0107	0.0163	0.048
$Pr(Satire X_i)$	0.0283	0.0248	0.0241	0.0175	0.0149	0.0147	0.0106	0.007	0

4.1 Finding Thresholds with GTRS

We formulated a competitive game between the criteria accuracy and coverage to obtain the balanced probabilistic thresholds with the initial thresholds $(\alpha, \beta) = (1, 0)$ and learning rate 0.03. As shown in the payoff table Table 5, the cell at the right bottom corner is the game equilibrium whose strategy profile is (β increases 0.06, α decreases 0.06). The payoffs of the players are (0.9784,0.3343). We set the stopping criterion as the increase of one player’s payoff is less than the decrease of the other player’s payoff when the thresholds are within the range. When the thresholds change from (1,0) to (0.94, 0.06), the accuracy is decreased from 1 to 0.9784 but the coverage is increased from 0.0795 to 0.3343. We repeat the game by setting (0.94, 0.06) as the next initial thresholds.

Table 5. The payoff table

		<i>cov</i>		
		α	$\alpha \downarrow 0.03$	$\alpha \downarrow 0.06$
<i>acc</i>	β	< 1, 0.0795 >	< 0.9986, 0.0849 >	< 0.9909, 0.1008 >
	$\beta \uparrow 0.03$	< 0.9868, 0.2337 >	< 0.9866, 0.2391 >	< 0.9843, 0.255 >
	$\beta \uparrow 0.06$	< 0.9799, 0.3130 >	< 0.9799, 0.3184 >	< 0.9784, 0.3343 >

The competitive games are repeated eight times. The result is shown in Table 6. After the eighth iteration, the repetition of game is stopped because the further changes on thresholds may cause the thresholds lay outside of the range $0 < \beta < \alpha < 1$, and the final result is the equilibrium of the seventh game $(\alpha, \beta) = (0.52, 0.48)$.

Table 6. The repetition of game

	Initial(α, β)	Strategies	Result(α, β)	Payoffs
1	(1, 0)	($\beta \uparrow 0.03, \alpha \downarrow 0.03$)	(0.94, 0.06)	< 0.9784, 0.3343 >
2	(0.94, 0.06)	($\beta \uparrow 0.03, \alpha \downarrow 0.03$)	(0.88, 0.12)	< 0.9586, 0.4805 >
3	(0.88, 0.12)	($\beta \uparrow 0.03, \alpha \downarrow 0.03$)	(0.82, 0.18)	< 0.9433, 0.554 >
4	(0.82, 0.18)	($\beta \uparrow 0.03, \alpha \downarrow 0.03$)	(0.76, 0.24)	< 0.9218, 0.6409 >
5	(0.76, 0.24)	($\beta \uparrow 0.03, \alpha \downarrow 0.03$)	(0.7, 0.3)	< 0.8960, 0.7467 >
6	(0.7, 0.3)	($\beta \uparrow 0.03, \alpha \downarrow 0.03$)	(0.64, 0.36)	< 0.8791, 0.8059 >
7	(0.64, 0.36)	($\beta \uparrow 0.03, \alpha \downarrow 0.03$)	(0.58, 0.42)	< 0.8524, 0.8946 >
8	(0.58, 0.42)	($\beta \uparrow 0.03, \alpha \downarrow 0.03$)	(0.52, 0.48)	< 0.8271, 0.9749 >

4.2 Results

We compare Pawlak rough sets, SVM, and our GTRS approach on the proposed dataset. Table 7 shows the results on the experimental data. The SVM classifier achieved an accuracy of 78% with a 100% coverage. The Pawlak rough set model using $(\alpha, \beta) = (1, 0)$ achieves a 100% accuracy and a coverage ratio of 7.95%, which means it can only classify 7.95% of the data. The classifier constructed by GTRS with $(\alpha, \beta) = (0.52, 0.48)$ reached an accuracy 82.71% and a coverage 97.49%. which indicates that 97.49% of data are able to be classified with accuracy of 82.71%. The remaining 2.51% of data can not be classified without providing more information. To make our method comparable to other baselines such as SVM, we assume random guessing is made on the deferral region and present the modified accuracy. The modified accuracy for our approach is then $0.8271 \times 0.9749 + 0.5 \times 0.0251 = 81.89\%$. Our methods shows significant improvement as compared to Pawlak model and SVM.

Table 7. The comparison results

	(α, β)	Accuracy	Coverage	Modified accuracy
SVM	-	78%	100%	78%
Pawlak	(1, 0)	100%	7.95%	53.98%
GTRS	(0.52, 0.48)	82.71%	97.49%	81.89%

5 Conclusion

In this paper, we propose a satirical news detection approach based on extracted semantic features and game-theoretic rough sets. In our model, the semantic features extraction captures the inconsistency in the different structural parts of the sentences and the GTRS classifier can process the incomplete information based on repetitive learning and the acceptance and rejection thresholds. The experimental results on our created satirical and legitimate news tweets dataset show that our model significantly outperforms Pawlak rough set model and SVM. In particular, we demonstrate our model’s ability to interpret satirical news detection from a semantic and information trade-off perspective. Other interesting extensions of our paper may be to use rough set models to extract the linguistic features at document level.

References

1. Akbik, A., Blythe, D., Vollgraf, R.: Contextual string embeddings for sequence labeling. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 1638–1649. Springer (2018)



2. Azam, N., Yao, J.T.: Game-theoretic rough sets for recommender systems. *Knowl.-Based Syst.* **72**, 96–107 (2014)
3. Burfoot, C., Baldwin, T.: Automatic satire detection: are you having a laugh? In: *Proceedings of the 2009 International Conference on Natural Language Processing*, pp. 161–164. ACL (2009)
4. Conroy, N.J., Rubin, V.L., Chen, Y.: Automatic deception detection: methods for finding fake news. In: *Proceedings of the Association for Information Science and Technology*, pp. 1–4. Wiley Online Library (2015)
5. Davidov, D., Tsur, O., Rappoport, A.: Semi-supervised recognition of sarcastic sentences in Twitter and Amazon. In: *Proceedings of the 14th Conference on Computational Natural Language Learning*, pp. 107–116. ACL (2010)
6. De Sarkar, S., Yang, F., Mukherjee, A.: Attending sentences to detect satirical fake news. In: *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 3371–3380. Springer (2018)
7. Ghosh, A., Veale, T.: Magnets for sarcasm: making sarcasm detection timely, contextual and very personal. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 482–491 (2017)
8. Jenks, G.F.: The data model concept in statistical mapping. *Int. Yearb. Cartography* **7**, 186–190 (1967)
9. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: *Empirical Methods in Natural Language Processing*, pp. 1532–1543. ACL (2014)
10. Riloff, E., Qadir, A., Surve, P., De Silva, L., Gilbert, N., Huang, R.: Sarcasm as contrast between a positive sentiment and negative situation. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 704–714 (2013)
11. Rubin, V., Conroy, N., Chen, Y., Cornwell, S.: Fake news or truth? using satirical cues to detect potentially misleading news. In: *Proceedings of the 2nd Workshop on Computational Approaches to Deception Detection*, pp. 7–17. ACM (2016)
12. Ruchansky, N., Seo, S., Liu, Y.: CSI: a hybrid deep model for fake news detection. In: *Proceedings of the 2017 Conference on Information and Knowledge Management*, pp. 797–806. ACM (2017)
13. Salton, G., McGill, M.J.: *Introduction to Modern Information Retrieval*. McGraw-Hill, New York (1986)
14. Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H.: Fake news detection on social media: a data mining perspective. *ACM SIGKDD Explor. Newsl.* **19**(1), 22–36 (2017)
15. Sterling, C.H.: *Encyclopedia of Journalism*. Sage Publications, New York (2009)
16. Wang, Y., et al.: Eann: event adversarial neural networks for multi-modal fake news detection. In: *Proceedings of the 24th SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 849–857. ACM (2018)
17. Yang, F., Mukherjee, A., Dragut, E.: Satirical news detection and analysis using attention mechanism and linguistic features. arXiv preprint [arXiv:1709.01189](https://arxiv.org/abs/1709.01189) (2017)
18. Yao, J.T., Herbert, J.P.: A game-theoretic perspective on rough set analysis. *J. Chongqing Univ. Posts Telecommun.* **20**(3), 291–298 (2008)
19. Yao, J.T., Azam, N.: Web-based medical decision support systems for three-way medical decision making with game-theoretic rough sets. *IEEE Trans. Fuzzy Syst.* **23**(1), 3–15 (2015)
20. Yao, Y.Y.: The superiority of three-way decisions in probabilistic rough set models. *Inf. Sci.* **181**(6), 1080–1096 (2011)

21. Yao, Y.Y.: An outline of a theory of three-way decisions. In: Proceedings of International Conference on Rough Sets and Current Trends in Computing, pp. 1–17. Springer (2012)
22. Yao, Y.Y.: Three-way decisions and cognitive computing. *Cognitive Comput.* **8**(4), 543–554 (2016)
23. Zhang, Y., Liu, P.F., Yao, J.T.: Three-way email spam filtering with game-theoretic rough sets. In: Proceedings of the 2019 International Conference on Computing, Networking and Communications, pp. 552–556. IEEE (2019)
24. Zhang, Y., Yao, J.T.: Determining three-way decision regions by combining gini objective functions and GTRS. In: Yao, Y., Hu, Q., Yu, H., Grzymala-Busse, J.W. (eds.) RSFDGrC 2015. LNCS (LNAI), vol. 9437, pp. 414–425. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25783-9_37
25. Zhang, Y., Yao, J.T.: Multi-criteria based three-way classifications with game-theoretic rough sets. In: Kryszkiewicz, M., Appice, A., Ślęzak, D., Rybinski, H., Skowron, A., Raś, Z.W. (eds.) ISMIS 2017. LNCS (LNAI), vol. 10352, pp. 550–559. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-60438-1_54

Deep Learning and Embeddings



Comparing State-of-the-Art Neural Network Ensemble Methods in Soccer Predictions

Tiago Mendes-Neves¹ and João Mendes-Moreira^{1,2}

¹ Faculdade de Engenharia, Universidade do Porto, Porto, Portugal
tiago.m.neves@inesctec.pt

² LIAAD-INESC TEC, Porto, Portugal

Abstract. For many reasons, including sports being one of the main forms of entertainment in the world, online gambling is growing. And in growing markets, opportunities to explore it arise. In this paper, neural network ensemble approaches, such as bagging, random subspace sampling, negative correlation learning and the simple averaging of predictions, are compared. For each one of these methods, several combinations of input parameters are evaluated. We used only the expected goals metric as predictors since it is able to have good predictive power while keeping the computational demands low. These models are compared in the soccer (also known as association football) betting context where we have access to metrics, such as rentability, to analyze the results in multiple perspectives. The results show that the optimal solution is goal-dependent, with the ensemble methods being able to increase the accuracy up to +3 % over the best single model. The biggest improvement over the single model was obtained by averaging dropout networks.

Keywords: Sports betting · Neural networks · Ensemble learning

1 Introduction

Gambling was always an interesting concept to human beings. If we ask a person if they want to trade 1€ for 0.95€ they will immediately reject the proposal. Being guaranteed to lose money is something that is not usually accepted without being rewarded. In betting, the reward comes from the existing probability of winning money. Even though in the long term more money is lost than won, the human brain is blinded by the prospect of a big win.

This paper is about soccer betting. Unlike other forms of gambling, in soccer betting, the probabilities are not predefined or easily calculated.

Bookmakers have the advantage of having access to the wisdom of the crowd that when combined with the ability for the market to self regulate, leaves them making a consistent profit regardless of the outcome.

This work is financed by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project UIDB/50014/2020.

The increase of available data on the soccer pitch along time and in terms of information richness allow nowadays the use of more data demanding machine learning algorithms to predict probabilities.

Academia has embraced the problem of predicting soccer matches very heavily. We are going to focus on machine learning approaches. A great amount of work uses Bayesian networks, such as [2] where they attempt to forecast the 2011/2012 season of the English Premier League and evaluate their model using the bookmaker odds. Also in the scope of Bayesian networks, [7] compares Bayesian networks with other state of the art machine learning algorithms, such as decision trees and k-nearest neighbors, concluding that machine learning algorithms had a better performance in the seasons tested (95/96 and 96/97, using Tottenham Hotspurs games on the English Premier League). Another algorithm used was the fuzzy-based models with genetic and neural tuning [11], tested on the Finnish Football League from 1991 to 1993, where the goal was to predict the score difference between both teams. Predictions using neural networks are also present in academia. [9] tested their hypothesis on several sports data, such as rugby and soccer, achieving the significant result of consistently being in the 99th percentile of a tipsters competition, meaning that it was able to beat a great part of human-made predictions. [3] attempts to predict price movements on betting exchanges using Artificial Neural Networks, focusing mainly on horse racing markets and yielding a significant ROI.

In 2017, the 2017 Soccer Prediction Challenge [14] purposed that researchers approached a data set from 52 leagues and seasons from 2000/01 to 2016/17 with more than 200 000 games, known as Open International Soccer Database [13]. Of the best work resulting from this issue, [15] uses Bradley—Terry model and a hierarchical Poisson log-linear model, falling more on the statistical model category. [16] used Bayesian networks that yields a very good results even with constraints of the challenge, since it only allows the usage of the final scores of the matches to build features for the models. It overtakes this obstacle by building dynamic ratings system. The winner, [17], also uses a rating system to generate features used in gradient boosted trees. This challenge has shown that, despite data set restrictions, it was still possible to obtain great results in terms of soccer predictions.

The machine learning algorithm that is of interest is the neural networks. Neural networks are nowadays being used to solve a lot of problems, namely in image data where they dominate over other techniques in terms of published research papers. Neural networks are connection-based models that have a very strong ability to assemble complex models. While neural networks having the ability to generate complex models may look like a plus, it can lead to overfitting, and with that, a bad generalization power, leading to sub-par performance when testing the models in previously unseen instances.

Ensemble models can help to solve this problem. Like the gathering of opinions in the betting markets improves the estimate over a single opinion, ensemble models gather the predictions of several models and by combining them allows the ensemble prediction to be better than any of the individual predictions alone.

Neural network have been used to solve several problems. Neural network ensembles have been extensively explored in the last couple of years. From 2017 to 2019 the number of published papers doubled (Science Direct search engine returns on the query “neural network ensemble”). 2019 has seen 3195 research papers while 2017 had only 1560. Due to its ability to work with image data, the majority comes from the health sector, such as [6] where they attempt to classify skin lesions in order to detect the cancerigenous ones, where they have been able to improve the score of a single network in 8%. Health anomaly detection seems to be the area of most success when using neural networks on image data. Even though image data seems to dominate the research in the area there is some research on quantitative data such the energy sector where [12] attempts to predict the load on the grid to improve systems planning.

What is missing from the academic perspective is a review of the available ensemble methods for neural networks. Neural networks have an extensive problem of overfitting. However, ensemble models have the ability to solve this problem. It is necessary to study what algorithms fit better with this technique, along with the development of ensemble techniques that enhance the advantages of the neural networks while hiding their flaws.

The reason for soccer data to be the ideal environment to test ensemble methods is that there is available a baseline in the bookmakers odds that is also calculated using an ensemble, the wisdom of the crowd.

The paper is organized as following. Section 2 describes the data used in the experiments. Section 3 talks about our experimental setup, focusing on the metrics, feature generation and models used. In Sect. 4 we describe the experiments made and discuss the results. Finally, in Sect. 5, we make our final conclusions and discuss possible future work in the area.

2 Describing the Data

The experiments described in this paper uses data from two sources: fivethirtyeight.com soccer-spi data set [4] and football-data.co.uk [5]. From the first, we retrieve the expected goals metric for every match from the season 2016/2017 to 2018/2019. On the second, we acquire the odds from the matches that we retrieved from the soccer-spi data set. The resulting data set is described in Table 1.

The data set has games from 6 leagues: English Premier League, French Ligue 1, Spanish La Liga, Italian Serie A, German Bundesliga, and Portuguese Primeira Liga. On this last, the 2016/2017 season data is not available.

The training set will be composed of the 2016/2017 and 2017/2018 season data, in a total of 3178 games. For the test set, the full 2018/2019 season is used, amounting to 1656 games.

Table 1. Variables present in the assembled data set.

Variable	Data type	Description
game_id	Symbolic	Internal unique id for integrating the databases
season	Symbolic	Season in which the game occurred
season_day	Numeric	Day of the season (season start set to July 1st)
home_team	Symbolic	Home team identifier
away_team	Symbolic	Away team identifier
ftw	Symbolic	Full time winner (H, D, A)
h_expected_goals	Numeric	Performance of the home team measured in expected goals
a_expected_goals	Numeric	Performance of the away team measured in expected goals
h_odd	Numeric	Average odd from the bookmakers for the home team to win
d_odd	Numeric	Average odd from the bookmakers for the draw
a_odd	Numeric	Average odd from the bookmakers for the away team to win

2.1 Expected Goals

The expected goals metric [10] is a measure of shot quality. It is calculated from the likelihood of a shot ending in a goal, taking into account factors such as distance to the goal, angle of the shot, body part used to make the shot and whether it was a first touch shot or not. The mathematical formulation can be seen in Eq. 1.

$$team\ X\ expected\ goals = \sum_{for\ all\ team\ X\ shots} P(shot\ leading\ to\ a\ goal) \quad (1)$$

This metric allows us to abstract from the binary goal metric. By incorporating probabilities we can obtain a better representation of how the game was played between both teams, reducing the randomness associated with soccer. This is the reason why this metric is gathering the attention of fans and the media.

3 Experimental Setup

3.1 Performance Metrics

The goal of the experiments is to obtain results that allow us to compare the performance of different ensemble algorithms. We are going to define a set of metrics that will be used, each one focusing on a part of the problem.

The first metric that will be used is the accuracy (Eq. 2). This is a metric that is of standard use in classification problems.

$$accuracy = \frac{\text{correct predictions}}{\text{number of predictions}} \quad (2)$$

The probabilities generated by the classification algorithms can be evaluated from two points of view: the betting and the regression point of view.

On the betting point of view, the defined metric was the rentability (Eq. 3), that tells us how much money the model made in relation to the stake. For this calculation, the algorithms will be always betting in the predicted favourite and the stake of each bet will be one unit.

$$rentability = \sum_{\text{correct predictions}} (\text{odd} - 1) - \text{number of incorrect predictions} \quad (3)$$

From the regression point of view, two metrics were used: bias and estimated standard deviation (referred simply as *estimated stdev* in the rest of the paper). Both bias and estimated stdev are defined in Eqs. 4 and 5, where M is the number of predictions made, γ is the predicted value and y is the real value. The bias is the error caused by the model's simplified assumptions that cause a constant error across different choices of training data. Estimated stdev means that for the same instance, the same model trained in slightly different data will yield different results.

$$bias^2 = \frac{1}{M} \sum_i^M y_i - \gamma_i \quad (4)$$

$$estimated\ stdev = \frac{1}{M} \sum_i^M stdev(\gamma_i) \quad (5)$$

The last metric measured is the average training time of the models. No optimization was made in any of the algorithms.

3.2 Feature Set

The feature set will be generated on top of the expected goals metric. For that, we will assemble the expected goals scored and conceded in each of the last 7 games for both teams facing up. This will leave us with 14 features for each team and a total of 28 features for our model.

3.3 Base Learners

There is a wide spectrum of choices when tuning a neural network, and these choices have a high impact in the final results of the neural network. Therefore

it is necessary to establish what will be the base learners used in the ensemble models in order to keep them comparable.

We are going to use the Keras API on top of the TensorFlow 1.13 to run these experiments. We have chosen Python 3.6 as the programming language. The parameters where no reference is made in this section revert to the default values that can be found in the Keras documentation.

Through an iteration process we found the base learners described next to yield good results as singular models.

The base learners are sequential models with dense layers. Two architectures were tested. The first uses a single hidden layer with 15 nodes (15,), the second uses two hidden layers of 10 and 5 nodes (10,5). In both cases, the layers use the *softmax* activation function, in order to have predictions in the form of probabilities. These two architectures were the ones that had the better performance when considering also time consumed.

A parameter that will be tested is the early stop. Since it naturally increases the variability of the models it might lead to better performance than the no early stop variant when ensembled.

The other non-default parameters are constant through the models. The learning rate is 0.025 and the batch size is 200. The loss function used is the categorical cross-entropy and the optimizer used is *Adam*. Dropout is used or not depending on the ensemble.

Table 2 summarizes the base learners used in the experiences.

Table 2. Presentation of the base learners.

Model	Architecture	Epochs	Early Stop	Patience
1	(15,)	100	No	
2	(15,)	500	Yes	10
3	(10,5)	100	No	
4	(10,5)	500	Yes	10

3.4 Proposed Algorithms

Bagging. Bagging [1] is perhaps the most common ensemble approach used. The idea is to generate new training data sets from a single data set by sampling, which can be performed with or without replacement.

In our implementation, similarly to what is done in random forest with the random subspace sampling, there is a parameter that allows us to modify the feature subset in which the models are trained. This parameter will be called *feature_ratio*, and it indicates a percentage of the features that will be used at each split. The number of samples from the base data that will be used is indicated by the parameter *sample_ratio*, which is a percentage of the samples that will be used. Samples are drawn without replacement.

Simple Average Dropout Networks (SADN). The SADN is an ensemble in which the goal of each model is to have low estimated stdev. The dropout parameter acts as regularization for the networks, not allowing them to become too complex and overfit. The predicted probabilities from the ensembles' models are then averaged and re-normalized in order to produce the ensemble predictions. On the experiments, the hidden layers will have a 0.3 dropout rate.

Negative Correlation Learning (NCL). In negative correlation learning [8], the approach is to train individual networks in an ensemble and combining them in the same process. All the neural networks in the ensemble are trained simultaneously and interactively through a correlation penalty term in their error function.

The difference from regular neural network training is in the loss function. Subtracted to the regular loss function (a function of the predicted value and the real value) is a percentage (λ parameter) of the loss function calculated between the value predicted by the model and the ensemble predicted value. This can be seen in Eq. 6, where γ is a neural network prediction, ε the ensemble prediction and y is the real value. This incentives models to go in a different direction from the average value of the ensemble when training, creating diversity in the model's opinions and improving the model classification performance.

$$\text{new loss function} = \text{loss function}(\gamma, y) - \lambda * \text{loss function}(\gamma, \varepsilon) \quad (6)$$

4 Experiments

4.1 Ensemble Hyperparameter Tuning

Before the comparison could be made, the ensemble model parameters need to be tuned. The first parameter defined is that each ensemble will have 50 base learners. While SADN does not need any additional parameter, both bagging and NCL have parameters that need to be tuned. To tune the parameters for the bagging method we do a grid search in order to find the optimal hyperparameters. This can be seen in Table 3.

Table 3. Hyperparameters grid search for bagging in model 1, with results in the format *Accuracy (estimated stdev)*. The results are averages of 10 runs.

		Sample ratio			
		0.25	0.5	0.75	1
Feature ratio	0.25	51.17 (0.0379)	51.07 (0.0455)	51.08 (0.0491)	50.98 (0.0517)
	0.5	51.15 (0.0387)	51.12 (0.0446)	51.04 (0.0484)	50.95 (0.0515)
	0.75	50.97 (0.0536)	50.99 (0.0515)	50.97 (0.0506)	50.92 (0.0513)
	1	50.95 (0.0526)	50.96 (0.0515)	50.95 (0.0509)	50.93 (0.0514)

Table 4. Hyperparameters grid search for NCL. The tests were done on the model 1. The results are averages of 10 runs.

Lambda parameter	0.05	0.10	0.20	0.30	0.40
Accuracy	50.1	50.16	49.6	49.71	47.2
Estimated stdev	0.0252	0.0291	0.0362	0.0494	0.0757

Table 5. Results from the tests with optimal parameters. Note that the expected rentability (calculated by betting in every outcome of every game) is -77.22 . The negative number demonstrates the earlier mentioned fact that bookmakers are making consistent profit.

Evaluation metric	Single model		Bagging		SADN		NCL
	Yes	No	Yes	No	Yes	No	No
Architecture	(15,)						
Accuracy	48.95	50.18	50.83	51.22	51.70	51.24	50.01
Rentability	-96.23	-77.92	-73.46	-65.85	-43.24	-60.83	-71.39
Bias	0.6284	0.6287	0.6288	0.6287	0.6297	0.6293	0.6293
Estimated stdev	0.2199	0.1049	0.0766	0.0397	0.0104	0.0112	0.0229
Average execution time	4.18	1.65	89.65	24.72	82.19	140.47	102.9
Architecture	(10,5)						
Accuracy	50.21	50.61	51.08	51.35	50.91	51.00	51.31
Rentability	-62.75	-70.44	-52.16	-61.29	-49.66	-52.40	-53.27
Bias	0.6291	0.6286	0.6334	0.6294	0.6326	0.6311	0.6302
Estimated stdev	0.1710	0.0881	0.0528	0.0304	0.0133	0.0095	0.0136
Average execution time	2.56	1.71	43.00	25.51	107.39	170.22	126.46

From Table 3 we can conclude that both `feature_ratio` and `sample_ratio` improve the performance when lowered. This can be verified from both accuracy and estimated stdev perspective. The best performing hyperparameters (both parameters equal to 0.25) will be used.

For the NCL we need to set the parameter λ . In the tuning phase, low λ values seemed to perform better, as seen in Table 4. The chosen λ is 0.1.

4.2 Ensemble Methods Comparison

Table 5 shows the results of the experiments. These results are the average of 50 runs of each algorithm.

Accuracy wise, the best performing model was the SADN (15,) with early stop with 51,7%. This is also the only instance where early stopping leads to better accuracy results since neither bagging or single models were able to improve when using early stop. In the (10,5) architecture the SADN with early stopping did not replicate this success. The accuracy performance of the SADN (15,) enabled the rentability to also be the best overall with -43.24, a value nearly 34 units above the expected value.

Table 6. A sample of the odds generated by the best model of each type in comparison with the bookmaker’s odd.

Season	Res.	Home	Away	Home					Draw					Away				
				Bookie	NN	BagNN	SADN	NCL	Bookie	NN	BagNN	SADN	NCL	Bookie	NN	BagNN	SADN	NCL
18/19	H	Napoli	Cagliari	1.26	1.83	1.58	1.48	1.72	6	4.45	4.94	5.92	3.99	11.14	4.39	6.12	6.4	5.9
18/19	H	Liverpool	Huddersfield	1.07	1.25	1.24	1.28	1.37	13.61	7.49	7.3	6.22	5.39	36.37	14.73	17.38	17.69	12.02
18/19	A	Nacional	Porto	13.51	10.48	9.18	12	7.02	6.65	8.17	6.92	6.86	5.48	1.19	1.28	1.34	1.3	1.48
18/19	H	Atalanta	Genoa	1.51	1.11	1.18	1.15	1.22	4.33	12.03	8.42	9.03	6.9	6.5	50.25	28.52	62.12	30.93
18/19	H	Paris SG	Monaco	1.51	1.48	1.4	1.53	1.38	4.74	4.11	5.78	5.02	5.44	5.69	12.19	8.86	6.81	11.02
18/19	A	Tondela	Santa Clara	1.76	3.78	6.49	4.4	5.5	3.63	2.55	3.62	3.89	4.93	4.57	2.92	1.76	1.94	1.63
18/19	D	Leipzig	Bayern M	4.61	5.02	2.82	2.42	2.18	4.22	2.99	3.19	3.66	3.66	1.69	2.15	3.01	3.19	3.74
18/19	H	Roma	Juventus	2.52	3.58	4.62	3.86	4.8	3.43	4.95	2.85	3.44	4.6	2.79	1.93	2.31	2.22	1.74
18/19	D	Leicester	Chelsea	2.4	4.02	2.58	2.67	2.36	3.57	3.83	3.59	3.43	3.7	2.89	2.04	3	3	3.27

Since these models were focused on estimated stdev reduction, it was expected that the bias did not change considerably. While improvements in bias are scarce, the cost of using ensembles was low, with none of the biases increasing by over 1% over the single model.

On the other side, the estimated stdev was immensely reduced, with some models achieving approximately 95% reduction. The most notable performance here was also obtained by the SADN algorithm, with the NCL being a close contender.

The more complex architecture (10,5) found it harder to improve results. While the performance jumped almost 3% in the best scenario for the (15,) architecture, the (10,5) failed to improve even 1%. However, with the exception of the SADN, all the algorithms managed to perform better on the (10,5) architecture.

Since the (15,) is a less complex model, ensembling with methods that do not induce more variability in predictions (SADN) yielded better results. On the other side, a more complex model, (10,5), needed algorithms that introduced variability in the models (bagging/NCL) to improve the predictions. We know that the most important factor when ensembling is to find the right balance of variability in the ensemble’s models. Either too much or too little variance will worsen the results. For this, the combination of simpler models trained with dropout and early stopping seem to be the best solution.

In terms of quickness, bagging is the better option. Without early stopping, bagging is able to only take approximately 15x more time to train than the simple network, while training 50x more models, due to the reduced number of instances and features used.

From the business point of view, all the ensemble methods were able to have a better performance than the single neural network. While the results are still far away from being profitable, we have to take into account the fact that these models are already able to beat the expected rentability of -77.22 units.

In Table 6 we are able to see that the models’ predictions are, in some instances, very similar to the bookmakers. This values could potentially be improved since the models do not take into account a lot of information about the games: player availability, how many rest days since last games, among others.

5 Conclusions

Ensembling neural networks improves the results of single models in terms of accuracy up to 3 %. This accuracy improvement can have a massive impact from the business perspective, as it can be seen in the rentability, with the best performing ensemble cutting the losses in half over the single model variant.

In general, ensemble proves itself to be a reliable way to reduce variance in the neural network context. However, in problems that are already very demanding in terms of computation time, this might not be optimal. An increase of 3% in accuracy lead to at least a 600% percent increase in computational time. Therefore, we conclude that the usage of ensemble techniques with neural networks is situational.

5.1 Future Work

While the results look promising, especially for SADN, the tests are only done in one data set. The algorithms need to be tested across multiple data sets to verify if these conclusions are consistent or if they only hold true in the soccer prediction scenario. Even in the same data set, tests with different parameters can be done.

References

1. Aggarwal, C.C.: *Neural Networks and Deep Learning*: eBook. Springer, Switzerland (2018)
2. Constantinou, A.C., Fenton, N.E., Neil, M.: Profiting from an inefficient association football gambling market: prediction, risk and uncertainty using Bayesian networks. *Knowl. Based Syst.* **50**, 60–86 (2013)
3. Dzalbs, I., Kalganova, T.: Forecasting price movements in betting exchanges using cartesian genetic programming and ANN. *Big Data Res.* **14**, 112–120 (2018)
4. FiveThirtyEight. fivethirtyeight.com. Accessed 21st June 2019
5. Football-Data.co.uk. football-data.co.uk. Accessed 21st June 2019
6. Harangi, B.: Skin lesion classification with ensembles of deep convolutional neural networks. *J. Biomed. Inform.* **86**, 25–32 (2018)
7. Joseph, A., Fenton, N.E., Neil, M.: Predicting football results using Bayesian nets and other machine learning techniques. *Knowl.-Based Syst.* **19**(7), 544–553 (2006)
8. Liu, Y., Yao, X.: Ensemble learning via negative correlation. *Neural Networks* **12**(10), 1399–1404 (1999)
9. McCabe, A., Trevathan, J: Artificial intelligence in sports prediction. In: *Fifth International Conference on Information Technology: New Generations*, pp. 1194–1197. IEEE Computer Society, Las Vegas (2008)
10. Opta. optasports.com. Accessed 22nd June 2019
11. Rotshtein, A.P., Posner, M., Rakityanskaya, A.B.: Football predictions based on a fuzzy model with genetic and neural tuning. *Cybern. Syst. Anal.* **41**(4), 619–630 (2005)
12. Ribeiro, G., Mariani, V., Coelho, L.: Enhanced ensemble structures using wavelet neural networks applied to short-term load forecasting. *Eng. Appl. Artif. Intell.* **82**, 272–281 (2019)

13. Dubitzky, W., Lopes, P., Davis, J., Berrar, D.: The open international soccer database. *Mach. Learn.* **108**, 9–28 (2019)
14. Berrar, D., Lopes, P., Davis, J., Dubitzky, W.: Guest editorial: special issue on machine learning for soccer. *Mach. Learn.* **108**(1), 1–7 (2018). <https://doi.org/10.1007/s10994-018-5763-8>
15. Tsokos, A., Narayanan, S., Kosmidis, I., Baio, G., Cucuringu, M., Whitaker, G., Király, F.: Modeling outcomes of soccer matches. *Mach. Learn.* **108**(1), 77–95 (2018). <https://doi.org/10.1007/s10994-018-5741-1>
16. Constantinou, A.C.: Dolores: a model that predicts football match outcomes from all over the world. *Mach. Learn.* **108**(1), 49–75 (2018). <https://doi.org/10.1007/s10994-018-5703-7>
17. Hubáček, O., Šourek, G., Železný, F.: Learning to predict soccer results from relational data with gradient boosted trees. *Mach. Learn.* **108**(1), 29–47 (2018). <https://doi.org/10.1007/s10994-018-5704-6>



Static Music Emotion Recognition Using Recurrent Neural Networks

Jacek Grekow^(✉) 

Faculty of Computer Science, Bialystok University of Technology,
Wiejska 45A, 15-351 Bialystok, Poland
j.grekow@pb.edu.pl

Abstract. The article presents experiments using recurrent neural networks for emotion detection for musical segments using Russell’s circumplex model. A process of feature extraction and creating sequential data for learning networks with long short-term memory (LSTM) units is presented. Models were implemented using the WekaDeeplearning4j package and a number of experiments were carried out with data with different sets of features and varying segmentation. The usefulness of dividing data into sequences as well as the sense of using recurrent networks to recognize emotions in music, whose results have even exceeded the SVM algorithm for regression, were demonstrated. The author analyzed the effect of the network structure and the set of used features on the results of regressors recognizing values on two axes of the emotion model: arousal and valence.

Keywords: Emotion detection · Audio features · Sequential data · Recurrent Neural Networks

1 Introduction

Music is an organization of sounds over time, and one of its more important functions is the transmission of emotions. The music created by a composer is ultimately listened to by a listener. The carriers of emotions are sounds distributed over time, their quantity, pitch, loudness, and their mutual relations. These sounds in music terminology are described by melody, timbre, dynamics, rhythm, and harmony. Before a person notices the emotions in music, he/she must have some time to analyze the listened to fragment [2]; depending on the changes in melody, timbre, dynamics, rhythm, or harmony, we can notice different emotions, such as happy, angry, sad, or relaxed.

The aim of this paper was to imitate the time-related perception of emotions in music by humans through the construction of an automatic emotion detection system using recurrent neural networks (RNN). Just as the human brain is “fed” with subsequent sound information over time, on the basis of which it perceives the emotions in music, similarly, the neural network downloads subsequent information vectors in subsequent time steps to predict the emotion value of the analyzed musical fragment.

Division into categorical and dimensional approach can be found in papers devoted to music emotion recognition (MER). In the categorical approach, a number of emotional categories (adjectives) are used for labeling music excerpts [8, 12]. In the dimensional approach, emotion is described using dimensional space, like the 2D model proposed by Russell [13], where the dimensions are represented by arousal and valence [5, 6, 9, 15].

MER task can also be divided into static or dynamic, where static MER detects emotions in a relatively long section of music of 15–20 s [4, 6, 8], and dynamic MER examines changes in emotions over the course of a composition, for example, every 0.5 or 1 s. Dynamic MER task was conducted by MediaEval Benchmarking Initiative for Multimedia Evaluation, the results of which were presented by Aljanaki et al. [1].

Long-short term memory recurrent neural networks were used in dynamic MER task by Coutinho et al. [5]. Low-level acoustic descriptors extracted using openSMILE and psychoacoustic features extracted with the MIR Toolbox were used as input data. A multi-variate regression using by deep recurrent neural networks was used to model the time-varying emotions (arousal, valence) of a musical piece [15]. In this work, a set of acoustic features extracted from segments of 1 s length were used. Delbouy et al., in [6], used mel-spectrogram from audio and embedded lyrics as input vectors to the convolutional and LSTM networks. Chowdhury et al., in [4], used VGG-style convolutional neural networks to detect 8 emotional characteristics (happy, sad, tender, fearful, angry, valence, energy, tension). For network training perceptual mid-level features (melodiousness, articulation, rhythmic stability, rhythmic complexity, dissonance, tonal stability, modality) were used, and spectrograms from audio signals were used as input vector for neural networks.

What distinguishes this work from others is that it uses a different segment length (6s) than the standard static MER, as well as proposes a method of preparing data for recurrent neural networks, which it tests with various low and mid-level features. Due to the fact that the studied segment is relatively short, a solution of using a sliding window also allows to study changes in emotions throughout the entire composition, i.e. similar to dynamic MER. This paper presents results in relation to previously conducted experiments [9].

The rest of this paper is organized as follows. Section 2 describes the music data set and the emotion model used in the conducted experiments. Section 3 presents the tools used for feature extraction and preparation of data before building the models. Section 4 describes the details of the built recurrent neural networks. Section 5 presents the results obtained while building the models. Finally, Sect. 6 summarizes the main findings.

2 Music Data

A well-prepared database of learning examples affects the results and the correctness of the created models predicting emotions. The advantages of the obtained database are well-distributed examples on the emotion plane as well as congruity between the music experts' annotations.

The data set consisted of 324 six-second fragments of different genres of music: classical, jazz, blues, country, disco, hip-hop, metal, pop, reggae, and rock. The tracks were all 22050 Hz mono 16-bit audio files in .wav format. The training data were taken from the generally accessible data collection project Marsyas¹. After the selection of samples, the author shortened them to the first 6 seconds, which is the shortest possible length at which experts could detect emotions for a given segment. Bachorik et al. [2] investigated the length of time required for participants to initiate emotional responses to musical samples.

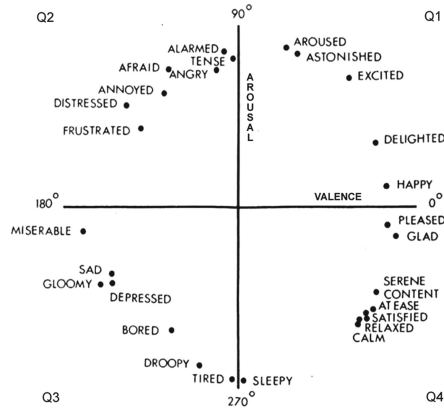


Fig. 1. Russell's circumplex model [13]

Data annotation was done by five music experts with a university musical education. The musical education of the experts, people who deal with the creation and analysis of emotions in music on a daily basis, enables to trust the quality of their annotations. Each annotator annotated all records in the data set - 324 six-second fragments. Each music expert had heard all the examples in the database. As a result during the annotation each annotator was able to see all the shades of emotions in music, which is not always the case in databases with the emotions determined. This had a positive effect on the quality of the received data, which was emphasized by Aljanaki et al. in [1].

During annotation of music samples, we used the two-dimensional arousal-valence Russell's model (Fig. 1) to measure emotions in music, which consists of two independent dimensions of arousal (vertical axis) and valence (horizontal axis). Each music expert making annotations after listening to a music sample had to specify values on the arousal and valence axes in a range from -10 to 10 .

Value determination on the arousal-valence axes (A-V) was clear with a designation of a point on the A-V plane corresponding to the musical fragment. The data collected from the five music experts was averaged. After annotation, the amount of examples obtained in the quarters on the A-V emotion plane were: Q1

¹ <http://marsyas.info/downloads/datasets.html>.

(A:high, V:high): 93; Q2 (A:high, V:low): 70; Q3 (A:low, V:low): 80; Q4 (A:low, V:high): 81.

A well-prepared database, i.e. one suitable for independent regressors predicting valence and arousal, should contain examples where the values of valence and arousal are not correlated. To check if valence and arousal dimensions are correlated in our music data, the Pearson correlation coefficient was used. The obtained value of $r = -0.03$ (i.e. close to zero) indicates that arousal and valence values are not correlated and the music data are a well spread in the quarters on the A-V emotion plane.

All examples in the database were marked by 5 music experts and their annotations had good agreement levels. A good level of mutual consistency was achieved, represented by Cronbach's α calculated for the annotations of arousal ($\alpha = 0.98$) and valence ($\alpha = 0.90$). We can see that the experts' annotations for the arousal value show slightly greater agreement than for the valence value, which is in line with the natural perception of emotions by humans [1]. Details on creating the music data were presented in a previous paper [10].

3 Feature Extraction

3.1 Tools for Feature Extraction

For feature extraction, tools for audio analysis and audio-based music information retrieval, Essentia [3] and Marsyas [14], were used. Marsyas software, written by George Tzanetakis, has the ability to analyze music files and to output the extracted features. The tool enables the extraction of the following features: Zero Crossings, Spectral Centroid, Spectral Flux, Spectral Rolloff, Mel-Frequency Cepstral Coefficients (mfcc), and chroma features - 31 features in total. For each of these basic features, Marsyas calculates four statistic features. The feature vector length obtained from Marsyas was 124.

Essentia is an open-source library, created at the Music Technology Group, Universitat Pompeu Fabra, Barcelona. In the Essentia package, we can find a number of executable extractors computing music descriptors for an audio track: spectral, time-domain, rhythmic, and tonal descriptors. Extracted features by Essentia are divided into three groups: low-level, rhythm, and tonal features. A full list of features is available on the web site². Essentia also calculates many statistic features: the mean, geometric mean, power mean, median of an array, and all its moments up to the 5th-order, its energy, and the root mean square (RMS). The feature vector length obtained from Essentia was 529.

3.2 Preparing Data for RNN

Recurrent neural networks process sequential data and find relationships between the input data sequences and the expected output value. To be able to train the recurrent neural network, it is necessary to enter sequences of the feature vectors.

² http://essentia.upf.edu/documentation/algorithms_reference.html.

In this paper, to extract correlations with time in the studied music fragments, they were segmented into smaller successive sections. The process of dividing a fragment of music (6 s) into smaller segments of a certain length t (1, 2 or 3 s) and overlap (0 or 50%) is shown in Fig. 2. To split the wav file, the *sfplay.exe* tool from Marsyas toolkit was used. From the created smaller segments of music, feature vectors were extracted, which were used to build a sequence of learning vectors for the neural network. A program was written that allows to select the segmentation option for a music fragment, performs feature extraction, and prepares data to be loaded to a neural network.

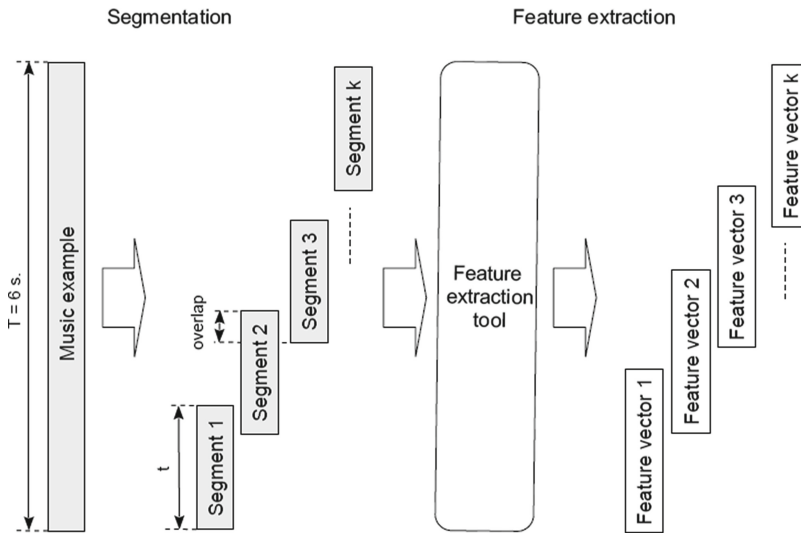


Fig. 2. Creating training data sequences for RNN

4 Recurrent Neural Networks

Long short-term memory (LSTM) units, which were defined in [7], were used to build recurrent networks. LSTM units are special kinds of memory blocks that solve the vanishing gradient problem occurring with simple RNN units. Each LSTM unit consists of a self-connected memory cell and three multiplicative regulators - input, output, and forget gates. Gates provide LSTM cells with write, read, and reset operations, which allows the LSTM unit to store and access information contained in a data sequence that corresponds to data distributed over time. The weights of connections in LSTM units need to be learned during training.

4.1 Implementation of RNN

The WekaDeeplearning4j package [11], which was included with the Weka program [16], was used to conduct the experiments with recurrent neural networks. This package makes deep learning accessible through a graphical user interface. The WekaDeeplearning4j module is based on Deeplearning4j³, which is a widely used open-source machine learning workbench implemented in Java. Weka with WekaDeeplearning4j package enables users to perform experiments by loading data in the Attribute-Relation File Format (ARFF), configuring a neural network, and running the experiment. To predict emotions in music files, a neural network was proposed with the structure shown in Fig. 3. Input data were given to the network in the form of a sequence set of features, and then processed by a layer consisting of LSTM units (LSTM1–LSTMn). The next layer built of densely connected neurons (1–n) converted the signals received from the LSTM layer and created an output signal.

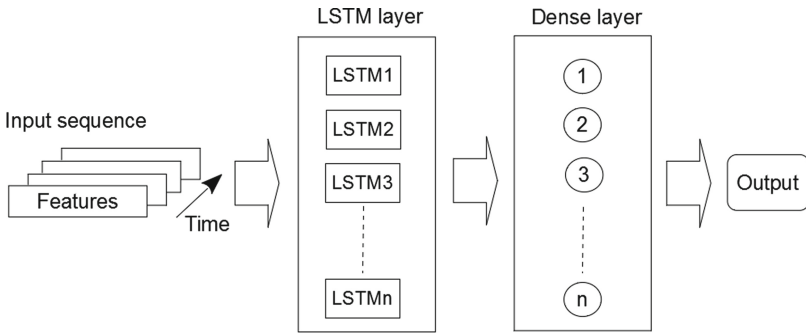


Fig. 3. Recurrent neural network architecture

4.2 Parameters of the RNN

The structure of the neural network was built once with one LSTM layer, once with two layers, and with different amounts of LSTM units (124, 248). A tanh activation function was used for LSTM units. For our regression task (prediction of continuous values of arousal and valence), the identity activation function for a dense layer was used, in conjunction with the mean squared error loss function. For weight initialization, the Xavier method was used. Stochastic gradient descent was used as a learning algorithm with Nesterov updater, which helped to optimize the learning rate. The network was trained with 100 epochs and to avoid overfitting an early stopping strategy was used. The training process was stopped as soon as the loss did not improve anymore for 10 epochs. The loss was evaluated on a validation set (20% of the training data).

³ <https://deeplearning4j.org>.

5 Experiments and Results

During the conducted experiments, regressors for predicting arousal and valence were built. As baseline for comparing the results of the obtained regressors a simple linear regression model (lr) was chosen. The data were also tested with a fairly good SMOreg algorithm with polynomial kernel, which is an implementation of the support vector machine for regression. The author also tested the usefulness of SMOreg in a previous paper [9]. Both algorithms (SMOreg, lr) were tested on the same music fragments as the neural networks but on non-segmented fragments.

The performance of regression algorithms were evaluated using the 10-fold cross validation technique (CV-10). The coefficient of determination (R^2) and mean absolute error (MAE) were used to assess model efficiency. Before constructing regressors arousal and valence annotations were scaled between $[-0.5, 0.5]$, thus the MAE value also corresponds to the average error percentage. Before providing input data to the neural network, the data was standardized to zero mean and unit variance.

Regressors were built using recurrent neural network RNN (RnnSequenceClassifier [11]) and were tested in 4 variants: RNN1 - 1 layer \times 124 LSTM units; RNN2 - 1 layer \times 248 LSTM units; RNN3 - 2 layers \times 124 LSTM units each; RNN4 - 2 layers \times 248 LSTM units each.

5.1 RNN with Marsyas Features

During the testing of RNN efficiency, features obtained from Marsyas tool were divided into 3 sets: (1) all Marsyas features - 124; (2) mfcc features - 13 Mel Frequency Cepstral Coefficients \times 4 statistic - 52; (3) chroma features - 54.

Table 1 presents the coefficient of determination (R^2) and mean absolute error (MAE) obtained during building regressors using mfcc and chroma features. The best results for each regressor type (arousal, valence) are marked in bold. From the obtained results, we can see that the usefulness of the chroma features is small compared with the mfcc features. The results for mfcc features far outweigh those for chroma features.

Table 2 presents the results for all Marsyas features. The simple linear regression model and support vector machine for regression (SMOreg) was outperformed by the RNN models, in two cases RNN3, RNN4 for arousal and valence. The best results were obtained with RNN4 (2 layers \times 248 LSTM): $R^2 = 0.67$ and $MAE = 0.12$ for arousal, $R^2 = 0.17$ and $MAE = 0.15$ for valence. We see that RNN with two LSTM layers gives better results for both arousal as well as valence. As expected, the results show that the sequential modeling capabilities of the RNN are useful for this task.

The use of all features gives the best results; however, in the case of arousal, the set of mfcc features gives quite comparable results, similar to the whole set of features ($R^2 = 0.66$ and $MAE = 0.12$, Table 1). The best results were obtained at a segment length of 2s and without overlap, and those are presented here.

Table 1. Results obtained for mfcc and chroma features

	Mfcc features				Chroma features			
	Arousal		Valence		Arousal		Valence	
	R^2	MEA	R^2	MEA	R^2	MEA	R^2	MEA
Linear regression	0.61	0.13	0.07	0.17	0.40	0.17	0.02	0.18
SMOreg	0.60	0.13	0.10	0.17	0.43	0.16	0.04	0.17
RNN1	0.58	0.14	0.14	0.17	0.43	0.17	0.02	0.18
RNN2	0.61	0.13	0.13	0.16	0.41	0.17	0.02	0.18
RNN3	0.66	0.12	0.11	0.16	0.48	0.15	0.03	0.17
RNN4	0.64	0.12	0.14	0.15	0.46	0.16	0.02	0.18

Table 2. Results obtained for all Marsyas features

	Arousal		Valence	
	R^2	MAE	R^2	MAE
Linear regression	0.56	0.14	0.13	0.17
SMOreg	0.62	0.13	0.15	0.16
RNN1	0.58	0.14	0.12	0.17
RNN2	0.62	0.13	0.12	0.17
RNN3	0.66	0.12	0.16	0.15
RNN4	0.67	0.12	0.17	0.15

5.2 RNN with Essentia Features

Experiments with the features obtained from the Essentia package were also conducted. These features include the mfcc and chroma features, which are also in the Marsyas tool, but also contain many higher-level features such as rhythm or harmony. Table 3 shows the results of the experiments. The experiments were expanded by two networks with an increased number of LSTM units, similar to the number of features in the sequence: RNN5 - 1 layer \times 529 LSTM units, RNN6 - 2 layers \times 529 LSTM units each.

From the obtained results (Table 3) for the Essentia features set, we can see a significant improvement of the results compared with the database algorithms (RNN1-RNN6 for arousal, RNN2-RNN6 for valence). Better features from the Essentia toolkit give better neural network results. The best results were obtained with RNN4: $R^2 = 0.69$ and $MAE = 0.11$ for arousal, $R^2 = 0.40$ and $MAE = 0.13$ for valence. The improvement is also significant for regressors for valence, compared with the results from Marsyas features (Table 2), where the best result was: $R^2 = 0.17$ and $MAE = 0.15$.

In regard to the different numbers of layers and LSTM units, the best results were obtained using the RNN4 network (2 layers \times 248 LSTM) for both arousal and valence. Two-layer networks recognized emotions better than one-layer

Table 3. Results obtained for Essentia features

	Arousal		Valence	
	R^2	MAE	R^2	MAE
Linear regression	0.07	0.25	0.06	0.19
SMOreg	0.48	0.18	0.27	0.17
RNN1	0.54	0.14	0.21	0.16
RNN2	0.58	0.14	0.32	0.14
RNN3	0.67	0.12	0.32	0.13
RNN4	0.69	0.11	0.40	0.13
RNN5	0.61	0.13	0.29	0.15
RNN6	0.68	0.12	0.36	0.14

networks. What is quite interesting in the case of arousal ($R^2 = 0.69$, $MAE = 0.11$), the results are comparable with the results obtained from the Marsyas package ($R^2 = 0.67$, $MAE = 0.12$, Table 2). Mfcc features are quite good for detecting arousal, and adding new features improved the results only slightly. A significant result of these experiments is that features from the Essentia package, like rhythm and tonal features, significantly improved the detection of valence. In the case of arousal, it is not necessary to use such a rich set of features, which is why the model for arousal is not so complex.

6 Conclusions

This article presents experiments using recurrent neural networks for emotion detection for musical segments. The sequential possibilities of the models turned out to be very useful for this type of task as the obtained results exceeded such algorithms as support vector machine for regression, not to mention the weaker linear regression. In all the built models, the accuracy of arousal prediction exceeded the accuracy of valence prediction. There was more difficulty detecting emotions on the valence axis than arousal. Similar difficulties were noted when music experts were annotating files, which was confirmed during annotation compliance testing.

It is significant that the use of higher-level features (features from Essentia tool) had a very positive effect on the models, especially the accuracy of valence regressors. Interestingly, to predict arousal, even a small set of features (mfcc from Marsyas tool) provided quite good results, similar to those of the large features set from Essentia. Low-level features, like mfcc, are generally sufficient for predicting arousal. In the future, feature selection for the Essentia set could be made and the most useful features chosen, despite that testing them on recurrent neural networks can be very time-consuming.

Acknowledgments. This research was realized as part of study no. KSIiSK in the Bialystok University of Technology and financed with funds from the Ministry of Science and Higher Education.

References

1. Aljanaki, A., Yang, Y.H., Soleymani, M.: Developing a benchmark for emotional analysis of music. *PLoS One* **12**(3), e0173392 (2017)
2. Bachorik, J., Bangert, M., Loui, P., Larke, K., Berger, J., Rowe, R., Schlaug, G.: Emotion in motion: investigating the time-course of emotional judgments of musical stimuli. *Music Percept.* **26**, 355–364 (2009)
3. Bogdanov, D., Wack, N., Gómez, E., Gulati, S., Herrera, P., Mayor, O., Roma, G., Salamon, J., Zapata, J., Serra, X.: ESSENTIA: an audio analysis library for music information retrieval. In: *Proceedings of the 14th International Society for Music Information Retrieval Conference*, pp. 493–498 (2013)
4. Chowdhury, S., Portabella, A.V., Haunschmid, V., Widmer, G.: Towards explainable music emotion recognition: the route via mid-level features. In: *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, Delft, The Netherlands*, pp. 237–243 (2019)
5. Coutinho, E., Trigeorgis, G., Zafeiriou, S., Schuller, B.: Automatically estimating emotion in music with deep long-short term memory recurrent neural networks. In: *Working Notes Proceedings of the MediaEval 2015 Workshop, Wurzen, Germany (2015)*
6. Delbouys, R., Hennequin, R., Piccoli, F., Royo-Letelier, J., Moussallam, M.: Music mood detection based on audio and lyrics with deep neural net. In: *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR 2018), Paris, France*, pp. 370–375 (2018)
7. Gers, F.A., Schmidhuber, J., Cummins, F.A.: Learning to forget: continual prediction with LSTM. *Neural Comput.* **12**, 2451–2471 (2000)
8. Grekow, J.: Audio features dedicated to the detection of four basic emotions. In: Saeed, K., Homenda, W. (eds.) *CISIM 2015. LNCS*, vol. 9339, pp. 583–591. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24369-6_49
9. Grekow, J.: Music emotion maps in arousal-valence space. In: Saeed, K., Homenda, W. (eds.) *CISIM 2016. LNCS*, vol. 9842, pp. 697–706. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45378-1_60
10. Grekow, J.: Human annotation. In: *From Content-Based Music Emotion Recognition to Emotion Maps of Musical Pieces. SCI*, vol. 747, pp. 13–24. Springer, Cham (2018). <https://doi.org/10.1007/978-3-319-70609-2>
11. Lang, S., Bravo-Marquez, F., Beckham, C., Hall, M., Frank, E.: WekaDeeplearning4j: a deep learning package for Weka based on Deeplearning4j. *Knowl.-Based Syst.* **178**, 48–50 (2019)
12. Lu, L., Liu, D., Zhang, H.J.: Automatic mood detection and tracking of music audio signals. *Trans. Audio Speech Lang. Proc.* **14**(1), 5–18 (2006)
13. Russell, J.A.: A circumplex model of affect. *J. Pers. Soc. Psychol.* **39**(6), 1161–1178 (1980)
14. Tzanetakis, G., Cook, P.: Marsyas: a framework for audio analysis. *Org. Sound* **4**(3), 169–175 (2000)

15. Weninger, F., Eyben, F., Schuller, B.: On-line continuous-time music mood regression with deep recurrent neural networks. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5412–5416 (2014)
16. Witten, I.H., Frank, E., Hall, M.A., Pal, C.J.: Data Mining: Practical Machine Learning Tools and Techniques, 4th edn. Morgan Kaufmann Publishers Inc., San Francisco (2016)



Saliency Detection in Hyperspectral Images Using Autoencoder-Based Data Reconstruction

Annalisa Appice^{1,3}✉, Francesco Lomuscio¹, Antonella Falini¹,
Cristiano Tamborrino², Francesca Mazzia¹, and Donato Malerba^{1,3}

¹ Dipartimento di Informatica, Università degli Studi di Bari Aldo Moro,
via Orabona, 4, 70126 Bari, Italy

{[annalisa.appice](mailto:annalisa.appice@uniba.it),[antonella.falini](mailto:antonella.falini@uniba.it),[ncesca.mazzia](mailto:ncesca.mazzia@uniba.it),
[donato.malerba](mailto:donato.malerba@uniba.it)}@uniba.it,
f.lomuscio7@studenti.uniba.it

² Dipartimento di Matematica, Università degli Studi di Bari Aldo Moro,
via Orabona, 4, 70126 Bari, Italy

cristiano.tamborrino@uniba.it

³ Consorzio Interuniversitario Nazionale per l'Informatica - CINI, Bari, Italy

Abstract. Saliency detection extracts objects attractive to a human vision system from an image. Although saliency detection methodologies were originally investigated on RGB color images, recent developments in imaging technologies have aroused the interest in saliency detection methodologies for data captured with high spectral resolution using multispectral and hyperspectral imaging (MSI/HSI) sensors. In this paper, we propose a saliency detection methodology that elaborates HSI data reconstructed through an autoencoder architecture. It resorts to (spectral-spatial) distance measures to quantify the salience degree in the data represented through the autoencoder. Finally, it performs a clustering stage in order to separate the salient information from the background. The effectiveness of the proposed methodology is evaluated with benchmark HSI and MSI data.

1 Introduction

Saliency detection has been inspired by the natural visual attention mechanism that identifies an object to be salient when it attracts visual attention more than anything else, i.e., the background. It has been demonstrated that the human vision system is prone to pay more attention to objects having higher contrast with their surroundings [9]. This consideration has paved the way for one of the earliest methodology, the Itti's method [10], that is based on the construction of center-surround contrast models on color images. These models are constructed by computing the Euclidean distance between the considered pixels and their surrounding ones in the color space. While simple, Itti's method has inspired several approaches investigated in the saliency investigation field for RGB images (see [3] for a recent survey). Recent models have also combined Itti's model

with a clustering stage [11] proving that clustering may help in the extraction of salient information. On the other side, in recent years, it has definitely emerged that many applications in various fields (e.g. weather forecasting, military intelligence) may take advantage of the elaboration of MultiSpectral Imaging (MSI) and HyperSpectral Imaging (HSI) data. Using multispectral/hyperspectral sensors, imagery data can be captured with a wide spectrum of light instead of just assigning primary colors (red, green, blue) to each pixel.¹ In general, abundant spectral information, going above and beyond human vision, captures more Earth’s features than standard RGB images [2].

Following the growing amount of HSI data collections, increasing attention has been recently devoted to saliency detection in HSI [8, 9, 13, 19]. However, due to the high dimensionality of HSI data, traditional computer vision methods based on gray-scale or color image analysis cannot be directly applied to HSI problems. On the other hand, witnessing the great success of saliency estimation models that, similarly to the Itti’s method, work in a low dimensional space, a natural research direction consists of combining a saliency detection methodology with a dimensionality reduction technique, in order to handle the abundant spectral information of HSI data [9]. Following this direction, we propose a HSI methodology—AISA (Autoencoding of Hyperspectral Imagery data for Saliency Analysis)—that cascades an autoencoder stage and a clustering stage. The autoencoder stage learns a non-linear encoding and decoding of HSI data. The clustering stage elaborates the autoencoder information, also engineered on surrounding pixels, to separate the salient object from the background, working in a lower dimensional space.

The remainder of this paper is organized as follows. The motivation and contributions are discussed in Sect. 2. The proposed methodology is illustrated in Sect. 3. Section 4 describes the results of the experimental evaluation. Finally, Sect. 5 draws the conclusions and future developments.

2 Motivation and Contributions

The autoencoder is an unsupervised deep neural network that can learn a codification of high dimensional input data so that the decodification resembles the input data as closely as possible [5]. As the codification is commonly used to obtain reduced dimensionality, the encoder representation of HSI data allows us to tackle “the curse of dimensionality” of a high spectrum and to perform data denoising. Recent studies have already proved that autoencoders can learn very interesting HSI data representations compared to other dimensionality reduction techniques [17]. Based on these promising results, autoencoders have been

¹ The main difference between multispectral and hyperspectral is the number of bands and how narrow the bands are. MSI technology commonly refers to a small amount of bands, i.e., from 3 to 10, sensed by a radiometer. HSI technology could have hundreds or thousands bands from a spectrometer. In this paper, we generally refer to HSI data defining a methodology that is then evaluated in both HSI and MSI scenario.

recently considered for the dimensionality reduction in HSI, for tasks of both change detection [1] and classification [20]. In this paper, we explore the use of autoencoders in HSI saliency detection tasks.

The rationale of our proposal is that clustering can be used to separate salient pixels, grouped in one cluster, from background pixels, grouped in another cluster. Clustering is performed on the encoder representation of the HSI data. In fact, the encoder should reduce the dimensionality of the spectrum keeping the hidden interactions and dependencies of the input, without losing relevant information and rejecting noise, as well as unnecessary redundancies [6]. On the other hand, we consider that the classification ability of the cluster algorithm may be improved by making the encoder “aware” on the reconstruction data. To this purpose, we compute the distance between the input HSI data and the data reconstructed in the same input space through the encoding-decoding architecture learned via the autoencoder. The use of the distance is a reasonable choice as the encoding and decoding process should behave similarly on pixels belonging to the same cluster. Consequently, a clustering stage, performed by also accounting for the distance between HSI data and reconstructed data, should contribute to correctly group the salient pixels in the same cluster. We integrate the spatial information into the autoencoder framework by computing a spectral-spatial distance feature that, following the main Itti’s idea, aids the identification of salient objects by contrast.

In short, the contributions of our paper are: (1) the use of autoencoder information, derived with both the encoder and the decoder level, in a HSI saliency detection task; (2) the analysis of spectral and spectral-spatial distances, coupled with autoencoder and clustering, for feature engineering; (3) an empirical evaluation of the effectiveness of the proposed methodology using both HSI and MSI data.

3 Methodology

AISA takes an hyperspectral image \mathbf{Z} as input. This image is a scene acquired at a specific time, represented as a tensor of size $m \times n \times K$. In particular, the scene is described by $m \times n$ pixels and K spectral bands. A pixel denotes an area of about a few square meters of the Earth’s surface—it is a function of the sensor’s spatial resolution—which is unequivocally referenced with spatial coordinates (x, y) with $1 \leq x \leq m$ and $1 \leq y \leq n$, according to the usual matrix representation. Every spectral band is a numeric feature proportional to the surface reflectance. The salience map, given as output by our algorithm, is a binary $m \times n$ matrix, which assigns a binary label (“salient”—1— and “no-salient”—0) to each pixel (x, y) of the scene. Based on these premises, the learning process performed by AISA is, in principle, divided into three stages, (1) training an autoencoder architecture with \mathbf{Z} ; (2) engineering a new feature space from the autoencoder; (3) performing clustering in the engineered feature space, in order to separate pixels in two clusters (salient cluster vs background cluster).

Stage 1 – Autoencoder. The autoencoder is computed by resorting to an encoder-decoder architecture that is used to map each input \mathbf{z} onto the encoding \mathbf{h} via an encoder network, so that $\mathbf{h} = \sigma(\mathbf{W}\mathbf{z} + \mathbf{b})$, where σ is an element-wise activation function, \mathbf{W} is a weight matrix and \mathbf{b} is a bias vector. Weights and biases are initialized randomly and then updated iteratively during training through backpropagation. The encoding is in turn mapped to the reconstruction \mathbf{z}' by means of a decoder network, so that $\mathbf{z}' = \sigma'(\mathbf{W}'\mathbf{h} + \mathbf{b}')$. In particular, the autoencoder is trained to minimize the loss through a feedforward neural network that reproduces the input data on the output layer. Both \mathbf{z} and \mathbf{z}' have the same dimension, while the autoencoder has as many layers as needed, placed symmetrically in the encoder and decoder. Every unit located at any of the hidden layers receives several inputs from the preceding layer. The unit computes the weighted sum of these inputs and applies the activation function to produce the output.

In this study, the input of the autoencoder comprises the $m \times n$ pixels spanned on the K spectral bands. We adopt ADAM, an adaptive learning rate optimization algorithm, in order to produce optimal weights and biases by minimizing the mean square error [12]. The encoder consists of six layers with K , $K/2$, $K/3$, $K/4$, $K/5$ and 1 neuron(s), respectively. The decoder starts in the bottleneck layer (the output layer of the encoder) and maps the bottleneck signals back to the input space through five layers with $K/5$, $K/4$, $K/3$, $K/2$ and K neurons, respectively. In addition, we use the sigmoid activation function that is commonly used in binary classification problems. Finally, we set the maximum number of epochs equal to 150 and the learning rate equal to 0.002.

Stage 2 – Feature Engineering. Let us consider the autoencoder trained in stage 1. For each pixel (x, y) , let \mathbf{z}_{xy} denote the spectral data acquired on pixel (x, y) , \mathbf{h}_{xy} be the representation of \mathbf{z}_{xy} on the bottleneck layer of the autoencoder and \mathbf{z}'_{xy} denote the reconstruction of \mathbf{z}_{xy} computed through the encoder and the decoder. Then we use this information to build a bi-variate representation of the image. In particular, we span every imagery pixel on two features, that is:

1. the encoder feature H that expresses the input data as they are represented at the bottleneck layer of the autoencoder;
2. the decoder feature D that is the distance computed between \mathbf{z}_{xy} and \mathbf{z}'_{xy} .

We evaluate the performance of four distance algorithms to build D , that is, Spectral Angle Mapper (SAM), Z-score (ZID), Z-score correction of Spectral Angle Mapper (SAMZID) and Spectral-Spatial Cross Correlation based Distance (SSCCD). SAM, ZID and SAMZID are spectral distance algorithms often used for material identification. They are simple and fast to compute [7]. SAM is independent of the number of spectral bands and insensitive to sunlight [15]. SAMZID minimizes the effect of noise and atmospheric corrections [7]. SSCCD couples the spectral information to the spatial arrangement of the pixels. As discussed in [18], SSCCD is helpful to overcome possible radiometric and dynamic range differences by accounting for the spatial information.

Formally, $SAM(x, y)$ measures the angle between \mathbf{z}_{xy} and \mathbf{z}'_{xy} , that is:

$$SAM(x, y) = \arccos \left(\frac{\mathbf{z}_{xy} \cdot \mathbf{z}'_{xy}}{\|\mathbf{z}_{xy}\| \|\mathbf{z}'_{xy}\|} \right), \quad (1)$$

where the “ \cdot ” denotes the scalar product.

$ZID(x, y)$ is computed as follows:

$$ZID(x, y) = \sum_{k=1}^K \left(\frac{(\mathbf{z}_{xy}[k] - \mathbf{z}'_{xy}[k]) - \mu_k}{\sigma_k} \right)^2, \quad (2)$$

where k represents the k -th spectral band so that $\mathbf{z}_{xy}[k] - \mathbf{z}'_{xy}[k]$ denotes the spectral divergence computed by applying the difference operator to the k -th band of both \mathbf{z}_{xy} and \mathbf{z}'_{xy} . The symbols μ_k and σ_k represent the mean and standard deviation of the spectral information divergence computed on the k -th spectral band.

The distance $SAMZID$ is computed by combining SAM and ZID through element-wise multiplication and a trigonometric operator. Formally,

$$SAMZID(\mathbf{Z}) = [scale(\sin SAM(\mathbf{Z}))] \times [scale(ZID(\mathbf{Z}))], \quad (3)$$

where $scale()$ is the scale function. In Eq. 3, the combined distance components are scaled to the interval $[0, 1]$.

Finally, the distance algorithm SSCCD is the distance formulation of the spectral-spatial cross correlation measure (SSCC) illustrated in [18]. This correlation metric compares spectral bands of both \mathbf{z} and \mathbf{z}' pairwise by accounting for the spatial arrangement of the spectrum in windows centered at the considered pixel (x, y) . Formally, $SSCC(x, y)$ is computed as follows:

$$SSCC(x, y) = \frac{\sum_{(x', y') \in W(x, y)} [\mathbf{z}_{x', y'} - \bar{\mathbf{z}}_{W(x, y)}]^T [\mathbf{z}'_{x', y'} - \bar{\mathbf{z}}'_{W(x, y)}]}{\sqrt{\sum_{(x', y') \in W(x, y)} \|\mathbf{z}_{x', y'} - \bar{\mathbf{z}}_{W(x, y)}\|^2} \sqrt{\sum_{(x', y') \in W(x, y)} \|\mathbf{z}'_{x', y'} - \bar{\mathbf{z}}'_{W(x, y)}\|^2}}, \quad (4)$$

where $\bar{\mathbf{z}}_{W(x, y)}$ and $\bar{\mathbf{z}}'_{W(x, y)}$ denote the mean vectors of the spectral vectors determined band-by-band on all the pixels of $W(x, y)$ in \mathbf{Z} and \mathbf{Z}' , respectively. In this study, $W(x, y) = \{(x + I, y + J) | I = \pm 2, J = \pm 2\}$. The distance measure $SSCCD(x, y)$ can be derived from $SSCC(x, y)$ as:

$$SSCCD(x, y) = 1 - scale(SSCC(x, y)), \quad (5)$$

where $scale()$ scales SSCC values in the range $[0, 1]$.

Stage 3 – Clustering. As a clustering algorithm, we consider the popular Gaussian Mixture Model (GMM) algorithm in the version described in [21]. Specifically, we train the estimation network, described in [21], on the bi-variate input constructed at stage 2 (i.e. the input that comprises the imagery data spanned on

both the encoder feature H and the distance feature D). This network estimates a probability density function under the GMM framework and predicts to which Gaussian distribution each pixel belongs to. In particular, the estimation network estimates the parameters of the GMM (mixture-component distribution, mixture means and mixture covariance) and evaluates the likelihood/energy of samples by utilizing a multi-layer neural network, in order to predict the mixture membership for each sample. We consider the estimation network as it has been implemented at <https://github.com/danieltan07/dagmm>. Similarly to [21], we use a threshold to label the samples of high energy as salient pixels (we consider the salient pixels as anomalies with respect to the background). However, differently from [21], where a pre-chosen threshold is considered, we use the Otsu’s method [14], in order to automatically determine this threshold. The Otsu’s method is a simple algorithm that returns a single intensity threshold to separate pixels into two classes—foreground and background. In particular, this threshold is determined by minimizing the intra-class intensity variance or, equivalently, by maximizing the inter-class variance. In this paper, we use the implementation of `threshold_otsu` from `skimage.filters`.

Final considerations concern the fact that the brute application of the described clustering stage may occasionally yield spurious, isolated assignments of pixels to clusters. To avoid this issue, we apply the principle of local autocorrelation congruence of objects [16]—detected clusters are generally expanding over contiguous areas. Based on this principle, we change the assignment of pixels that strongly disagree with the surrounding assignments. This corresponds to perform a spatial-aware correction of the original cluster assignments. Based upon this correction, each pixel may be re-assigned to the cluster (and to its label) that originally groups the most part of its neighboring pixels.

4 Experimental Study

AISA is written in Python 3.7 with Pytorch to build autoencoders. To evaluate AISA, we consider HS-SOD dataset [9]—a collection of 60 hyperspectral images with their respective ground-truth binary saliency images.² Several competitors have been already evaluated using these data. We also consider two multispectral images of Madrid collected with Sentinel-2A satellite.

4.1 HS-SOD Data

The images have been collected by accounting for several aspects during the data collection (e.g. variation in object size, number of objects, foreground-background contrast, object position on the image). Data have been acquired with a NH-AIK model hyperspectral camera from fifty scenes at the public parks of Tokyo Waterfront City in Japan in several days between August and September 2017. Spectral bands are collected in the visible spectrum (380–780 nm)

² The dataset is available at <https://github.com/gistaicr/HS-SOD>.

Table 1. AISA: mean \pm standard deviation of AUC-Borji computed on 60 images of HS-SOD by varying the distance among SAM, ZID, SAMZID and SSCCD (column 1); number of images of HS-SOD where AUC-Borji is the highest per distance (column 2).

Distance	AUC-Borji (mean \pm stdev)	Count
SAM	0.7152 \pm 0.1258	9
ZID	0.7057 \pm 0.1267	12
SAMZID	0.6781 \pm 0.1377	6
SSCCD	0.7540 \pm 0.1244	33
Best distance	0.7727 \pm 0.1083	–

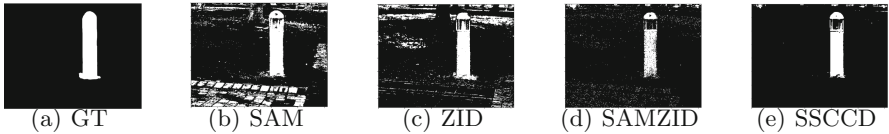


Fig. 1. AISA output on image id=26 of HS-SOD dataset: the binary ground truth-GT (Fig. 1(a)), the saliency maps computed by AISA with SAM (AUC-Borji = 0.8414, Fig. 1(b)), ZID (AUC-Borji = 0.8399, Fig. 1(c)), SAMZID (AUC-Borji = 0.7716, Fig. 1(d)), SSCCD (, AUC-Borji = 0.8509, Fig. 1(e)).

leaving 81 spectral bands. Each image consists of 1024×768 pixels. The saliency ground truth is available for these images. For the quantitative evaluation of the saliency detection performances, we compute the Area Under Curve (AUC) metric. We consider the AUC implementation described in [4] (AUC-Borji) that is commonly used in saliency detection method evaluations [9].

We start this analysis by evaluating the performance of AISA with respect to the distance measure computed in the feature engineering stage. For each distance, the mean and standard deviation of AUC-Borji computed on 60 images is reported in column 1 of Table 1, while the number of times each distance achieves the highest AUC-Borji is reported in column 2 of Table 1. Figure 1 reports an example of saliency maps built by varying the distance. These results highlight that the highest overall performance is achieved by using the SSCCD distance that accounts for the spatial arrangement of the data besides the spectral information. In any case, the accuracy analysis, performed image per image, also shows that although SSCCD achieves the highest AUC-Borji in 33 out of 60 images, SAM, ZID or SAMZID outperform SSCCD in the remaining images. This result suggests that a future development of this research consists in exploring if any dependence exists between the best distance and the image characteristics (e.g. camera settings such as exposure time and gain values, weather condition or salient object material, which may change in the considered images).

We complete the analysis by comparing AISA to competitors reported in [9]. Competitors include: Itti’s method computed on spectral data and its variants checking the spectral distances between each spatial region for saliency

Table 2. Competitor analysis. Results of competitors are collected in [9]

Method	AUC-Borji
AISA	0.7760
Itti et al.	0.7694
SED	0.6415
SAD	0.7521
GS	0.7597
SED-OCM-GS	0.7863
SED-OCM-SAD	0.8008
SGC	0.8205

computation by using spectral Euclidean distance (SED) and spectral Angle distances (SAD); the method computing the saliency on the spectral bands divided in groups and calculating the Euclidean distance as color opponency between groups (GS); the methods using the orientation based salient features (OCM) in the combinations SED-OCM-GS and SED-OCM-SAD; the method using the spectral gradient contrast (SGC) in combination with super-pixels extracted by considering both spatial and spectral gradients. Results, reported in Table 2, show that AISA outperforms Itti et al. method, SED, SAD and GS, while it is outperformed by methods including orientation features (SED-OCM-GS and SED-OCM-SAM) and/or super-pixels (SGC). This result suggests that new accuracy can be gained in AISA by augmenting the feature space of the clustering stage with orientation features and/or adding super-pixel in the learning stage. In particular, super-pixels, automatically determined based on spatial-spectral arrangement of data, may aid in revealing higher-level saliency patterns that may significantly help in denoising the data and neglecting details.

4.2 Madrid Data

The Sentinels are a fleet of satellites designed to deliver the wealth of data and imagery that are central to the European Commission’s Copernicus programme. Sentinel-2 carries an innovative wide swath high-resolution multispectral images with a swath width of 290 km and 13 spectral bands. We have considered two images 1000×1000 in the area of Madrid (Spain). No ground truth is available for these data. RGB rendering of these images has been reported in Figs. 2(a) and 2(c), while the saliency maps computed by AISA with SSCC have been plotted in Figs. 2(b) and 2(d). The algorithm is able to delineate salient areas in the landscape, which roughly correspond to the river, roads and buildings in Madrid 1. The algorithm is also able to track the river path in Madrid 2.

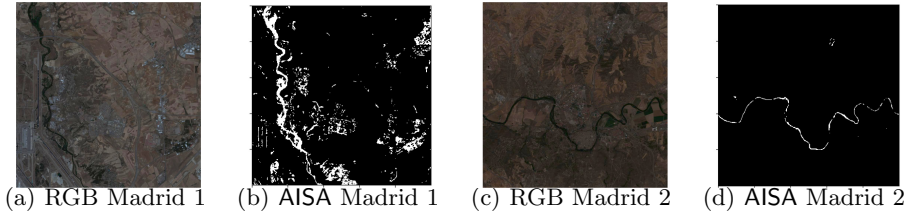


Fig. 2. RGB rendering and saliency map computed by AISA on Madrid 1 and Madrid2.

5 Conclusion

This paper illustrates AISA—an unsupervised machine learning methodology that addresses saliency detection problems in HSI by resorting to salient information extracted through autoencoders. Spatial data arrangement is also taken into account to improve the accuracy. The experimental study, performed using 60 hyperspectral images and 2 multispectral images, reveal that AISA achieves competitive accuracy compared to recent state-of-the-art techniques disclosing relevant salient regions. Based on the validation results, some directions for further work are still to be explored. Orientation features may be integrated in the elaboration, in order to better delineate the shape of the salient objects detected. On the other hand, super-pixel segmentation may be performed, in order to introduce spectral information aggregated at spatial level and perform salient object detection possibly abstracting on details within the objects.

Acknowledgments. This work fulfills the research objectives of the PON “Ricerca e Innovazione” 2014–2020 project RPASInAir “Integrazione dei Sistemi Aeromobili a Pilotaggio Remoto nello spazio aereo non segregato per servizi” (ARS01.00820), funded by the Italian Ministry for Universities and Research (MIUR). The research of Antonella Falini is founded by PON Project AIM 1852414 CUP H95G18000120006 ATT1. The research of Cristiano Tamborrino is funded by PON Project “Change Detection in Remote Sensing” CUP H94F18000270006. We thank Planetek Italia srl for Madrid data.

References

1. Appice, A., Di Mauro, N., Lomuscio, F., Malerba, D.: Empowering change vector analysis with autoencoding in bi-temporal hyperspectral images. In: CEUR Workshop Proceedings, vol. 2466(2019)
2. Appice, A., Malerba, D.: Segmentation-aided classification of hyperspectral data using spatial dependency of spectral bands. *ISPRS J. Photogrammetry Remote Sens.* **147**, 215–231 (2019)
3. Borji, A., Cheng, M.-M., Hou, Q., Jiang, H., Li, J.: Salient object detection: a survey. *Comput. Vis. Media* **5**(2), 117–150 (2019). <https://doi.org/10.1007/s41095-019-0149-9>

4. Borji, A., Tavakoli, H.R., Sihite, D.N., Itti, L.: Analysis of scores, datasets, and models in visual saliency prediction. In: 2013 IEEE International Conference on Computer Vision, pp. 921–928 (2013)
5. 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)
6. Han, W., Wang, G., Tu, K.: Latent variable autoencoder. *IEEE Access* **7**, 48514–48523 (2019)
7. Hussain, M., Chen, D., Cheng, A., Wei, H., Stanley, D.: Change detection from remotely sensed images: from pixel-based to object-based approaches. *ISPRS J. Photogrammetry Remote Sens.* **80**, 91–106 (2013)
8. Imamoglu, N., Ding, G., Fang, Y., Kanezaki, A., Kouyama, T., Nakamura, R.: Salient object detection on hyperspectral images using features learned from unsupervised segmentation task. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2019), pp. 2192–2196. IEEE (2019)
9. Imamoglu, N.: Hyperspectral image dataset for benchmarking on salient object detection. In: 2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX), pp. 1–3 (2018)
10. Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* **20**(11), 1254–1259 (1998)
11. Jia, Y., Hao, C., Wang, K.: A new saliency object extraction algorithm based on Itti’s model and region growing. In: 2019 IEEE International Conference on Mechatronics and Automation (ICMA), pp. 224–228 (2019)
12. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: 3rd International Conference on Learning Representations (ICLR 2015), Conference Track Proceedings (2014). [arXiv:1412.6980](https://arxiv.org/abs/1412.6980)
13. Liang, J., Zhou, J., Bai, X., Qian, Y.: Salient object detection in hyperspectral imagery. In: 2013 IEEE International Conference on Image Processing, pp. 2393–2397 (2013)
14. Otsu, N.: A threshold selection method from gray-level histograms. *IEEE Trans. Geosci. Remote Sens.* **9**(1), 62–66 (1972)
15. Seydi, S.T., Hasanlou, M.: A new land-cover match-based change detection for hyperspectral imagery. *Eur. J. Remote Sens.* **50**(1), 517–533 (2017)
16. Wang, J., Liu, S., Zhang, S.: A novel saliency-based object segmentation method for seriously degenerated images. In: 2015 IEEE International Conference on Information and Automation, pp. 1172–1177 (2015)
17. Windrim, L., Ramakrishnan, R., Melkumyan, A., Murphy, R.J., Chlingaryan, A.: Unsupervised feature-learning for hyperspectral data with autoencoders. *Remote Sens.* **11**(7), 1–19 (2019)
18. Yang, Z., Mueller, R.: Spatial-spectral cross-correlation for change detection : a case study for citrus coverage change detection. In: ASPRS 2007 Annual Conference, vol. 2, pp. 767–777, January 2007
19. Zhang, L., Zhang, Y., Yan, H., Gao, Y., Wei, W.: Salient object detection in hyperspectral imagery using multi-scale spectral-spatial gradient. *Neurocomputing* **291**, 215–225 (2018)
20. Zhou, P., Han, J., Cheng, G., Zhang, B.: Learning compact and discriminative stacked autoencoder for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **57**(7), 4823–4833 (2019)
21. Zong, B., et al.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: International Conference on Learning Representations (2018)



Mesoscale Anisotropically-Connected Learning

Qi Tan, Yang Liu, and Jiming Liu^(✉)

Department of Computer Science, Hong Kong Baptist University,
Hong Kong, China
{csqtan, csygliu, jiming}@comp.hkbu.edu.hk

Abstract. Predictive spatio-temporal analytics aims to analyze and model the data with both spatial and temporal attributes for future forecasting. Among various models proposed for predictive spatio-temporal analytics, the recurrent neural network (RNN) has been widely adopted. However, the training of RNN models becomes slow when the number of spatial locations is large. Moreover, the structure of RNN is unable to dynamically adapt to incorporate new covariates or to predict the target variables with varying spatial dimensions. In this paper, we propose a novel method, named Mesoscale Anisotropically-Connected Learning (MACL), to address the aforementioned limitations in RNN. For efficient training, we group the dataset into clusters (which refers to the mesoscale) along the spatial dimension according to the spatial adjacency and develop individual prediction module for each cluster. Then we design an anisotropic information exchange mechanism (i.e., the information exchange is not symmetric), to allow the prediction modules leveraging state information from nearby clusters for enhancing the prediction accuracy. Furthermore, for timely adaptation, we develop a local updating strategy for adapting the learning model to incorporate new covariates and the target variables with varying spatial dimensions. Experimental results on a real-world prediction task demonstrate that our method can be trained faster and more accurate than existing methods. Moreover, our method is flexible to incorporate new covariates and target variables of varying spatial dimensions, without sacrificing the prediction accuracy.

1 Introduction

Predictive spatio-temporal analytics, which aims to model and forecast the data collected across space and time, has attracted increasing research interests recently [14, 16]. In predictive spatio-temporal analytics, datasets are generally collected from various spatial locations over a historical period and then fed into the learning model for predicting the target variables in these locations [12]. For example, citywide traffic flow data are collected during the past weeks or months for traffic condition prediction in different district or streets [16]. The nationwide or even global climatic data of the past years or decades are collected for

climate forecast [9]. Moreover, in a dynamic environment, new covariates and new target variables in varying spatial locations arrives sequentially [13]. Given such a large volume of spatio-temporal data in a dynamic environment, on the one hand, it is desired to efficiently train reliable spatio-temporal models. On the other hand, the structure of the designed spatio-temporal models should be flexible to incorporate new covariates and to adapt for making predictions on new dimensions of the target variables (i.e., the new spatial locations), because retraining the entire model for new covariates or the target variables on new spatial locations is computationally expensive or even infeasible.

The recurrent neural network (RNN) based spatio-temporal models achieve the state-of-the-art performance in various predictive spatio-temporal analytics tasks [8, 14]. For capturing the temporal correlation among data, the RNN incorporates the covariates in the previous time steps and then generate a final hidden representation for future forecast. For capturing the dependency among different sequences, e.g., the traffic conditions in the nearby districts, conventionally, the RNN concatenates the multiple sequences as multivariate input.

However, there are two limitations in the current RNN-based models. First, when learning with a large-scale dataset, as the network size of a RNN is quadratically related to the hidden size, it is difficult to train an RNN with large hidden size for incorporating all necessary information when the number of spatial locations is large [2, 7]. Second, when learning in a dynamic environment, the structure of RNN is unable to dynamically adapt to incorporate new covariates or to predict the target variables with varying spatial dimensions. Therefore, in this paper, we aim to develop a novel RNN-based model for efficient training and flexible adaptation.

1.1 Related Work

Many RNN-based spatio-temporal models are proposed in various applications. Yao et al. [14] employed an RNN for the temporal view, which takes the representations from the spatial view and context features as input. Li et al. [8] developed a novel RNN by integrating the spatial graph in the transition matrix. However, these models cannot be easily adapted to the scenarios with varying covariates and new locations.

Some studies investigated learning with varying feature spaces, such as evolvable features [5] and trapezoidal data [17]. However, they were based on the shallow neural networks, which may limit the learning power of the developed models. In this paper, we investigate the problem of online update under the deep model.

For efficient online updating, Doyen et al. [10] proposed a Hedge Back-Propagation method for refining the parameters of DNN effectively. They modified the DNN structure for fast error back-propagation. On the other hand, factorized LSTMs are studied for reducing the number of parameters to be estimated [1, 6], but their input layers are not factorized, which leads to limited computational cost reduction and cannot incorporate new covariate.

1.2 Motivation

As the existing methods cannot address the limitations of RNN in large-scale dynamic predictive spatio-temporal analytics, we attempt to tackle these challenges by exploring the intrinsic property of spatio-temporal data. We observe the fact that the state of one location (i.e., one dimension of the target variable), generally, is closely related to the states of geographically nearby locations, but is less relevant to the states of the faraway locations [11]. Therefore, it may not be necessary to require the temporal sequences at all spatial locations to be used as inputs for predicting the target variables in one location. Similarly, when one new covariate in some locations is collected, it could be the nearby locations that need the new data for the predictive tasks.

In summary, the core question to be answered in this paper is: *How to design an RNN-based learning model for efficient training and timely adaptation in large-scale predictive spatio-temporal analytics, such that new covariates and target variables of varying spatial dimensions in a dynamic environment can be flexibly incorporated into the model, without sacrificing its prediction accuracy?*

1.3 Our Contributions

In this work, we propose a novel method, named Mesoscale Anisotropically-Connected Learning (MACL), to address the above-mentioned question by utilizing the spatial locality and mesoscale connectivity. We first group the dataset into clusters (which refers to the mesoscale) along the spatial dimension according to the spatial adjacency, so that the target variables of the spatial locations in the same cluster are closely related to each other. We develop individual RNN-based prediction module for each cluster. The features extracted from multiple covariates in one cluster are fused as that cluster’s state information. Moreover, the state information in nearby clusters may also benefit the prediction, but the influence of different nearby clusters may not be homogeneous. Therefore, we design an information exchange mechanism, which is anisotropic (i.e., the information exchange is neither symmetric nor homogeneous) to allow the prediction modules leveraging information from nearby clusters to enhance the prediction accuracy. We use different weights to combine information from different nearby clusters. Furthermore, we develop a local updating strategy to adapt the learning model so that new covariates can be incorporated and the target variables with varying spatial dimensions can be predicted. Figure 1 illustrates the procedure of our method. The contributions of this paper are summarized as follows.

1. We investigate an important problem in predictive spatio-temporal analytics, i.e., how to design a deep spatio-temporal recurrent model for efficient training and timely adaptation in a large-scale dynamic environment.
2. We propose a novel learning method MACL to solve the above challenging problem.
3. We perform extensive experiments on a real-world traffic flow dataset, showing the effectiveness and efficiency of the proposed method.

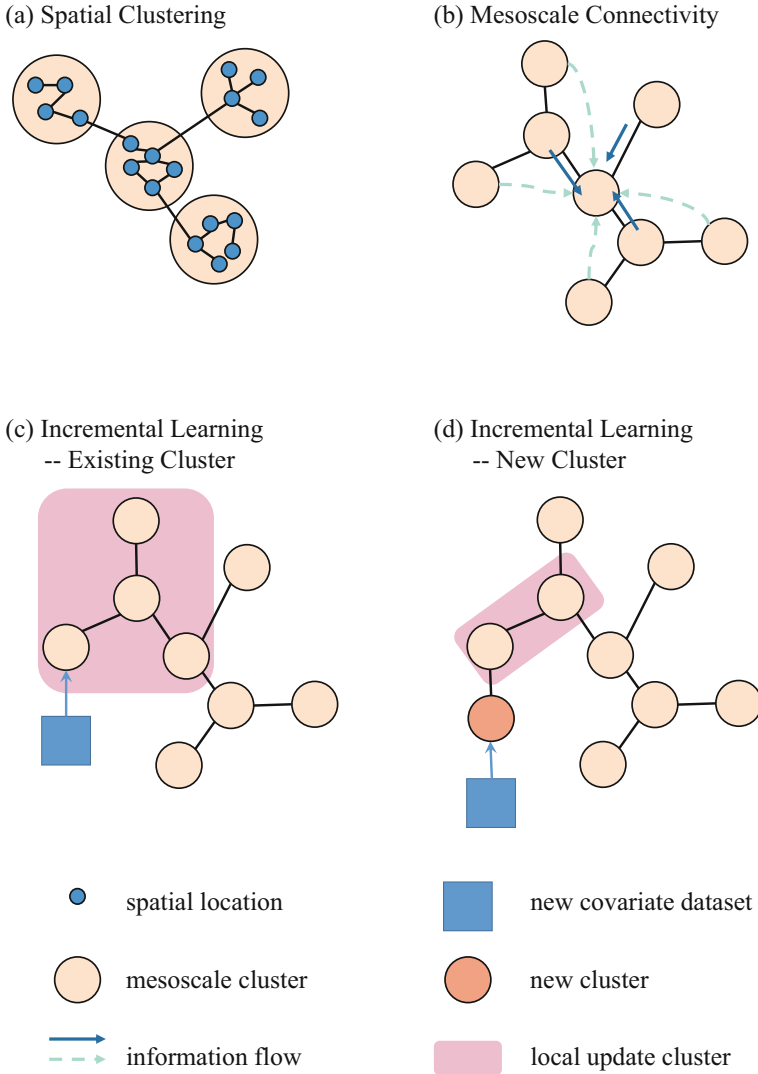


Fig. 1. Illustration of the proposed MACL. (a) First, the locations are grouped into several clusters based on the spatial adjacency. Then the mesoscale connectivity among the clusters is constructed by linking the clusters with their K closest clusters. Section 2.2 shows the details of spatial clustering and connecting procedure. (b) During the learning, the features of the data in the same cluster and the state information from the neighborhood clusters are utilized to update the state representation for prediction. The scenario that the state information is forwarded $L = 2$ times is illustrated. Section 2.3 shows the details of the formulations. (c) In a dynamic environment, when incorporating new covariates in the existing cluster, only prediction modules of the clusters within the distance of 2 need to be updated, i.e., training the modules in the pink box with new covariates via error back-propagation, avoiding the retraining of the entire model. (d) In a dynamic environment, when the new covariates are collected in a new cluster, the mesoscale connections are added and the local update is applied. Section 2.4 shows the details of incremental learning procedure.

2 Approach

In this section, we present the proposed MACL. First, we provide the necessary notations and formally define the problem. Then we introduce the structure of the proposed model and the update strategy for learning with new covariates and new target variables in varying spatial dimension.

2.1 Problem Formulation

In this subsection, we formally define the problem of predictive spatio-temporal analytics with large-scale datasets in a dynamic environment, which requires efficient training and timely adaptation. Let $\mathbf{Y} \in \mathbb{R}^{N \times T}$ be the target data collected in N locations during T time steps, $\mathbf{y}_t \in \mathbb{R}^N$ be the target data at time step $t, t = 1, \dots, T$. Let \mathcal{X} be the set of covariate data and $\mathcal{X}_{[t-t^l:t]}$ be the covariate data from time step $t - t^l$ to t . Predictive spatio-temporal analytics task could be regarded as learning the mapping function parameterized by θ , $f_\theta : \mathcal{X}_{[t-t^l:t]} \rightarrow \mathbf{y}_t$, where t^l is the time length of the features. For simplicity, we denote $\mathcal{X}_{[t-t^l:t]}$ as \mathbf{x}_t and $\mathbf{x}_t^{(d)}$ as the d th dataset, $d = 1, \dots, D$. The predictive spatio-temporal analytics problem becomes more challenging when predicting with large-scale datasets, i.e. N and $|\mathcal{X}|$ are large. Moreover, in a dynamic environment, new data, \mathcal{X}^n and \mathbf{Y}^n , arrive sequentially. We need to timely adapt the model, $f'_\theta : [\mathbf{x}_t, \mathbf{x}_t^n] \rightarrow [\mathbf{y}_t, \mathbf{y}_t^n]$, to incorporate the new covariates and to predict the new target variables.

2.2 Spatial Clustering

In order to achieve efficient training, as illustrated in Fig. 1(a), we first group the locations into several clusters and then build a recurrent prediction module for each cluster: we can group the locations based on the spatial adjacency or mutual information correlation. Assume we group N locations into g clusters and construct the adjacency graph, $G = \langle V, E \rangle, |V| = g$, for these g clusters. There might be multiple covariate datasets collected in one cluster. Thus we use the LSTM [4] to extract the feature for each covariate dataset d :

$$\mathbf{s}_t^{(d)} = \text{LSTM}(\mathbf{x}_t^{(d)}). \quad (1)$$

where $\mathbf{s}_t^{(d)}$ is the extracted feature from one data set $\mathbf{x}_t^{(d)}$ for predicting \mathbf{y}_t . In doing so, we can model the complex interactive among locations within one cluster, as often they are spatially close or semantically similar. We use the hidden state at the last time step as the feature extracted from one dataset. Then we can aggregate the information from the input data for each location:

$$\mathbf{h}_{n,t}^0 = \sum_{d \in \mathcal{S}^n} \mathbf{s}_t^{(d)}, \quad (2)$$

where \mathcal{S}^n is the set of datasets inputted into the n -th cluster.

2.3 Anisotropically-Connected Learning

In order to improve the prediction accuracy of predictive model for each cluster, we further consider the mesoscale correlation between the clusters, i.e., the data in the nearby clusters is also informative for predicting the target variable. As illustrated in Fig. 1(b), we build up a mechanism for exchanging the information among clusters by passing the message to the neighborhoods:

$$\mathbf{m}_{n,t}^l = \sum_{i \in \mathcal{N}^n} \mathcal{G}^{n,i}(\mathbf{h}_{i,t}^{l-1}), \quad (3)$$

where \mathcal{N}^n is the neighborhoods of cluster n and $\mathcal{G}^{n,i}(\cdot)$ is the specific information mapping from cluster i to cluster n . Then we update the state of one cluster as:

$$\mathbf{h}_{n,t}^l = \mathcal{F}^n([\mathbf{m}_{n,t}^l, \mathbf{h}_{n,t}^{l-1}]), \quad (4)$$

where $\mathcal{F}^n(\cdot)$ is the update function of cluster n . Note that if such message exchange could be operated for L times, one cluster could receive the information from other clusters within the distance of L . Finally, the output for target variable estimation is calculated as:

$$\hat{y}_{n,t} = \mathcal{O}^n(\mathbf{h}_{n,t}^L), \quad (5)$$

where $\mathcal{O}^n(\cdot)$ is the output function for cluster n .

2.4 Incremental Learning

For timely adaptation, we develop an incremental learning strategy based on the mesoscale connected structure developed above. The incremental learning strategy consists of the following two steps.

Model Extension. We will introduce the extension strategies in the following two scenarios:

New Covariate From Existing Cluster. In the covariate adaptation, i.e., a new covariate dataset from the existing cluster is collected to facilitate the prediction, we learn a new RNN for the new covariate as shown in Eq. (1). Then we could treat it in the way as other covariates in this cluster and fuse all covariates for prediction, as shown in Eq. (2). Figure 1(c) illustrates the procedure of adding new covariate from the existing cluster.

New Covariate and Target Variable From New Cluster. In this case, as illustrated in Fig. 1(d), we first add this cluster into G and link this new cluster to other existing clusters. Then we add and learn new LSTM model, message exchanging function, update function, and output function for the new cluster.

Local Update. In both scenarios, only the nearest L step neighbourhoods need to be updated, i.e., we fix the parameters in other modules while just train the modules of these neighbourhood, which avoids the retraining of the entire model.

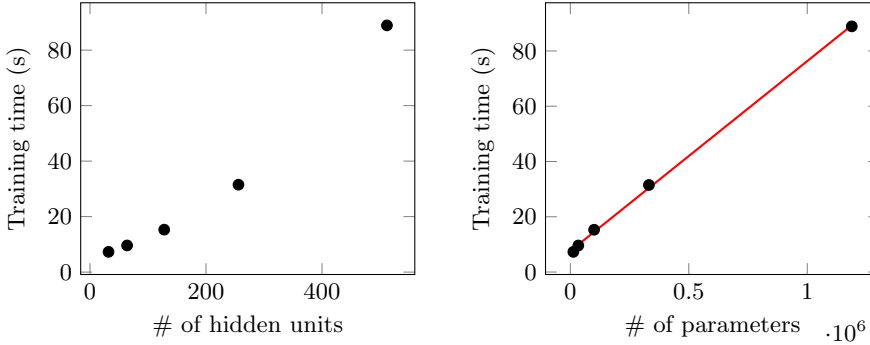


Fig. 2. Running time of LSTM in 1,000 back-propagation iterations with varying numbers of hidden units (left) and parameters (right). The number of parameters in 1-layer LSTM is $4h^2 + 4hd$, where h is the number of hidden units and d is the input size.

Traditional RNN needs to learn $h^2 + hN + hI$ parameters while our model just needs to learn $(2h^2 + hN + hI)/g$ parameters, where g is the number of clusters. In the online update module, we just need to learn $(h^2 + hN + hI)/g^2$ parameters for each influenced module. Thus our approach achieves much faster training speed for new covariate and target variable, compared with traditional RNN.

3 Experiments

In this section, we evaluate the proposed method in efficient training with large-scale datasets and timely adaptation with new covariates and target variables.

We use the traffic crowd flows data¹ in Beijing from Jul. - Oct. 2013 [16] in 1,024 locations. We use the inflow and outflow in the previous time steps to predict the inflow of each location at the current time step. We use the data at the first 70% time steps for training and the remaining 30% for testing.

We group the locations into 20 clusters and construct the connections between clusters based on spatial coordinates. We link each cluster to $K = 2$ closest clusters. We measure the prediction accuracy using the rooted mean square error (RMSE): $RMSE = \frac{1}{TN} \sum_{t=1}^T \sum_{n=1}^N \|y_t^n - \hat{y}_t^n\|$. We implement the models in Pytorch and conduct the experiments on the cluster with CPU Core i7-4771 3.50 GHz, GPU RTX 2070 and 32 GB Memory.

3.1 Evaluation on Efficient Training

We first show the importance of reducing the number of parameters on the fast learning. Figure 2 shows the training time of LSTM model with varying number of hidden units and parameters. We can see that the training time is

¹ Available at <https://github.com/lucktroy/DeepST/tree/master/data>.

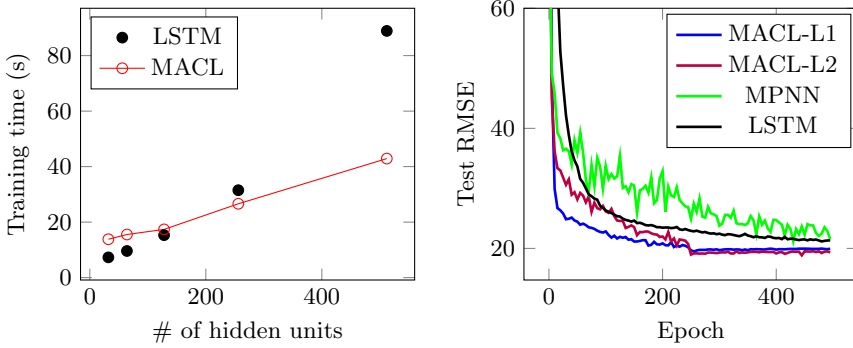


Fig. 3. (Left) Comparison of training time between LSTM and the proposed framework in 1,000 back-propagation iterations with varying number of hidden units. (Right) Comparison of test performance among the proposed framework, MPNN, and LSTM. In our methods, MACL-L1 exchanges the information among the clusters 1 time and MACL-L2 exchanges the information 2 times.

quadratically related to the number of hidden units and linearly correlated with the number of parameters. Thus, reducing the number of parameters and number of hidden units can significantly reduce the computation time of training LSTM.

Then we evaluate the efficiency of the proposed method. Figure 3 shows the training time of our proposed model and LSTM (left) and the testing RMSE of different methods (right). Without loss of generality, all the models are trained with SGD optimizer with the same learning rate. Orthogonal to our work, there are some studies working on using graph convolution neural networks to model spatial correlation [15], among which the message passing network (MPNN) [3] is most similar to our model. In our method, we set the information to be exchanged among clusters for 1 time (MACL-L1) and 2 times (MACL-L2). The results show that our method is trained faster and converges quicker to a better result than the baseline LSTM and the state-of-the-art MPNN. Comparing with LSTM, our method decomposes the large-scale predictive task into several interactive sub-tasks and trains a smaller model for each sub-task, which makes the models can be trained easier. Comparing with MPNN, our method provides more flexibility in modeling the heterogeneous connections by learning an unique weight parameter for each connection.

3.2 Evaluation on Timely Adaptation

In this subsection, we evaluate the adaptation in two scenarios: sequentially arriving new covariates and target variables in new location.

First we test the effectiveness of incorporating new sequentially arriving covariates. We partition the data of each region into 5 groups and input one group of data for each location in certain epoch to simulate the sequentially arriving covariate data. Figure 4 (left) shows the performance of our method

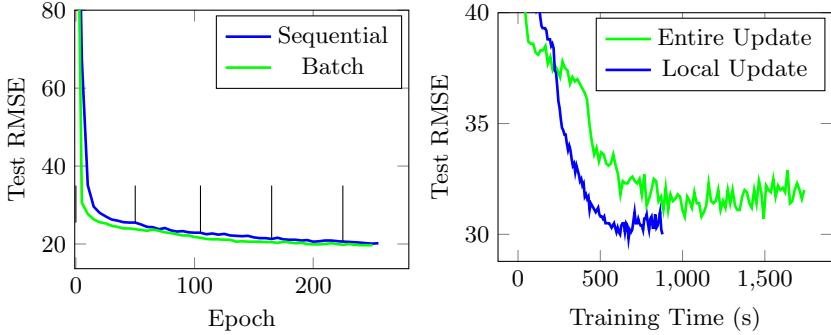


Fig. 4. (Left) The black vertical bars indicate the epochs in which one new covariate dataset is inputted. The performance of the sequential update version, after it converges, is comparable to that of the batch update version. (Right) New locations are added for a trained model. In the entire update version, the entire network is updated after the new location is added; while in the local update version, the partial network is updated. The local update version achieves lower RMSE with shorter training time.

with the sequentially updating way and that with the batch updating. It can be observed from the experimental results that even in the sequentially updating scheme, our model could achieve similar performance as the batch updating version, in which all datasets arrive before the training.

Then we test the efficiency of timely updating when a new target location is explored. We first train the entire network with all clusters except one. Then we adapt the model to incorporate this new cluster and we measure the performance of our method in predicting on this new cluster. Figure 4 (right) shows the performance using the locally updating strategy vs. the entirely updating strategy. The experimental result shows that the locally updating strategy produces better results in a shorter time than the entirely updating strategy.

The above results shows that the spatial mesoscale anisotropically-connected structure is able to achieve fast local adaptation and effective local update, without loss of performance.

4 Conclusion and Future Work

In this paper, we introduced a novel method for efficient training and timely adaptation of RNN model. The results of experimental in the large-scale dynamic environment demonstrated that our model can be trained faster than existing methods while achieving lower prediction error. Moreover, with the local updating and adaptation strategy, our method is flexible to incorporate new covariates and target variables of varying spatial dimensions.

In the future work, we plan to evaluate our framework on larger-scale datasets. Increasing the number of clusters can take the advantage of parallel

computation, while this requires more between-cluster connections to improve accuracy, which increases the computational cost. We will further explore how to determine the number of clusters to achieve good efficiency.

Acknowledgement. The authors would like to thank the reviewers for their valuable comments. This work was supported by the grants from the Research Grant Council of Hong Kong SAR under Project RGC/HKBU12201619 and RGC/HKBU12201318.

References

1. Belletti, F., Beutel, A., Jain, S., Chi, E.: Factorized recurrent neural architectures for longer range dependence. In: International Conference on Artificial Intelligence and Statistics, pp. 1522–1530 (2018)
2. Gao, F., Wu, L., Zhao, L., Qin, T., Cheng, X., Liu, T.Y.: Efficient sequence learning with group recurrent networks. In: Proceedings of the 2018 Conference of the North American Association for Computational Linguistics, vol. 1, pp. 799–808 (2018)
3. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: Proceedings of the 34th ICML, pp. 1263–1272 (2017)
4. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
5. Hou, B.J., Zhang, L., Zhou, Z.H.: Learning with feature evolvable streams. In: Advances in Neural Information Processing Systems, pp. 1417–1427 (2017)
6. Kuchaiev, O., Ginsburg, B.: Factorization tricks for LSTM networks. ICLR Workshop (2017)
7. Lei, T., Zhang, Y., Wang, S.I., Dai, H., Artzi, Y.: Simple recurrent units for highly parallelizable recurrence. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 4470–4481 (2018)
8. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: data-driven traffic forecasting. In: NIPS 2017 Time Series Workshop (2017)
9. Lozano, A.C., et al.: Spatial-temporal causal modeling for climate change attribution. In: Proceedings of the 15th SIGKDD, pp. 587–596. ACM (2009)
10. Sahoo, D., Pham, Q., Lu, J., Hoi, S.C.: Online deep learning: learning deep neural networks on the fly. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence, pp. 2660–2666. AAAI Press (2018)
11. Tobler, W.R.: A computer movie simulating urban growth in the Detroit region. *Econ. Geogr.* **46**(sup1), 234–240 (1970)
12. Wu, X., Zhu, X., Wu, G.Q., Ding, W.: Data mining with big data. *IEEE Trans. Knowl. Data Eng.* **26**(1), 97–107 (2014)
13. Yao, H., Liu, Y., Wei, Y., Tang, X., Li, Z.: Learning from multiple cities: a meta-learning approach for spatial-temporal prediction. In: The World Wide Web Conference, pp. 2181–2191. ACM (2019)
14. Yao, H., et al.: Deep multi-view spatial-temporal network for taxi demand prediction. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
15. Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. arXiv preprint [arXiv:1709.04875](https://arxiv.org/abs/1709.04875) (2017)
16. Zhang, J., Zheng, Y., et al.: Predicting citywide crowd flows using deep spatio-temporal residual networks. *Artif. Intell.* **259**, 147–166 (2018)
17. Zhang, Q., Zhang, P., Long, G., Ding, W., Zhang, C., Wu, X.: Online learning from trapezoidal data streams. *IEEE Trans. Knowl. Data Eng.* **28**(10), 2709–2723 (2016)



Empirical Comparison of Graph Embeddings for Trust-Based Collaborative Filtering

Tomislav Duricic^{1,2}(✉), Hussain Hussain², Emanuel Lacic², Dominik Kowald², Denis Helic¹, and Elisabeth Lex¹

¹ Graz University of Technology, Graz, Austria
{tduricic,dhelic,elisabeth.lex}@tugraz.at

² Know-Center GmbH, Graz, Austria
{hhussain,elacic,dkowald}@know-center.at

Abstract. In this work, we study the utility of graph embeddings to generate latent user representations for trust-based collaborative filtering. In a cold-start setting, on three publicly available datasets, we evaluate approaches from four method families: (i) factorization-based, (ii) random walk-based, (iii) deep learning-based, and (iv) the Large-scale Information Network Embedding (LINE) approach. We find that across the four families, random-walk-based approaches consistently achieve the best accuracy. Besides, they result in highly novel and diverse recommendations. Furthermore, our results show that the use of graph embeddings in trust-based collaborative filtering significantly improves user coverage.

Keywords: Recommender systems · Empirical study · Graph embeddings · Cold-start · Trust networks

1 Introduction

Recommender systems suffer from the well-known cold-start problem [1] that arises when users have rated no or only few items. The cold-start problem is particularly problematic in neighborhood-based recommendation approaches such as collaborative filtering (CF) [2] since the ratings of these users cannot be exploited to find similar users. Trust-based recommender systems (e.g., [3–6]) have been proposed as a potential remedy for the cold-start problem. They alleviate this problem by generating a trust network, i.e., a type of a social network in which nodes usually represent users and edges represent trust connections between users based on their explicitly expressed or implicitly derived trust relationships. Although trust is a complex and ambiguous concept from social sciences, in the context of recommender systems, we use a simple interpretation in which users trust other users in the system if they trust their opinions and ratings on different items [5]. Resulting trust network can be used to find the k -most similar users, whose items are recommended to a target user. Trust networks

are, however, typically sparse [7] since only a fraction of users have trust connections, which makes finding the k -similar users challenging. In the present work, we explore the utility of *graph embeddings* to extract the k -similar users from trust networks. To that end, we conduct experiments on three publicly available benchmark datasets often used in studies on trust-based recommender systems: Epinions [5], Ciao [8], and Filmtrust [9]. We empirically evaluate a range of state-of-the-art graph embedding approaches [10] from four distinct method families, i.e., (i) factorization-based methods, (ii) random-walk-based approaches, (iii) methods based on deep learning, and (iv) the LINE approach [11] that falls in neither of these families, with respect to their ability to deliver accurate, novel, and diverse recommendations [12] for cold-start users.

In our experimental setup, we split each dataset into a validation set (warm-start users) and a test set (cold-start users). For each graph embedding approach, we perform a hyperparameter optimization on each validation set. We then select the hyperparameters which result in highest recommendation accuracy. We generate recommendations for each target user in a CF manner by finding k -similar neighbors using the learned embeddings and ranking their items by similarity scores. Finally, we evaluate the resulting graph embeddings against a corresponding test set with respect to accuracy and beyond accuracy metrics. We compare the graph embedding approaches against five baselines from trust-based recommender systems, commonly used in cold-start settings: (i) Most Popular (MP) recommends the most frequently rated items, (ii) $Trust_{dir}$ extracts trusted users directly from a trust network, (iii) $Trust_{undir}$ ignores edge directions and extracts neighbors from the resulting undirected network, (iv) $Trust_{jac}$ applies the Jaccard coefficient on the explicit trust network, and (v) $Trust_{Katz}$ [13] computes the Katz similarity to infer transitive trust relationships between users. To quantify the algorithmic performance, we evaluate recommendation quality in terms of nDCG, novelty, diversity and user coverage.

We find that as a result of their ability to create a representation of each user in a network, graph embeddings are able to improve user coverage when compared to the baseline approaches. Our experiments also show that random-walk-based approaches, i.e., Node2vec and DeepWalk, consistently outperform other graph embedding methods on all three datasets in terms of recommendation accuracy. Finally, we find a positive correlation between novelty and accuracy in all three datasets suggesting that users in the respective platforms tend to prefer novel content. Summing up, our contributions are three-fold. Firstly, we provide a large-scale empirical study on the efficacy of graph embedding approaches in trust-based recommender systems. Secondly, unlike many previous studies, which evaluated only recommendation accuracy, we compare different approaches with respect to beyond accuracy metrics such as novelty, diversity and user coverage. Lastly, our results provide new insights into user preferences based on correlations between different recommendation quality metrics.

2 Graph Embeddings

In this study, we compare the recommendation performance of graph embedding approaches from four distinct method families [10], i.e., factorization-based methods, random-walk-based approaches, deep-learning-based approaches, and the LINE approach [11] which falls in neither of the first three families.

Factorization-based Approaches. Factorization-based approaches produce node embeddings using matrix factorization. The inner product between the resulting node embedding vectors approximates a deterministic graph proximity measure [14]. In total, we investigate five different factorization approaches:

- *Graph Factorization (GF)* [15] factorizes the adjacency matrix and determines proximity between nodes directly on the adjacency matrix.¹
- *Laplacian Eigenmaps (LE)* [16] factorizes the normalized Laplacian matrix and preserves the 1st-order proximity.¹
- *Locally Linear Embedding (LLE)* [17] minimizes the squared difference between the embedding of a node and a linear combination of its neighbors' embeddings, weighted by the edges connecting to them. The solution of this minimization problem reduces to a factorization problem. (see footnote 1)
- *High-Order Proximity preserved Embedding (HOPE)* is able to preserve higher-order proximities and capture the asymmetric transitivity. [18]. (see footnote 1)
- *Graph Representations with Global Structural Information (GraRep)* [19] can handle higher-order similarity as it considers powers of the adjacency matrix.²

Random Walk-Based Approaches. RW-based approaches first identify the context of a node with a random walk and then learn the embeddings typically using a skip-gram model [10]. In total, we evaluated three different approaches:

- *DeepWalk* [20] extracts node sequences with truncated random walks and applies a skip-gram model [21] with hierarchical softmax on the node pairs.³
- *Node2vec* [22] extends DeepWalk with hyperparameters to configure the depth and breadth of the random walks. In contrast to DeepWalk, Node2vec enables to define flexible random walks, while DeepWalk only allows unbiased random walks over the graph [14].⁴
- *Role2vec* [23] uses attributed random walks to learn embeddings. As Role2vec enables to define functions that map feature vectors to types, it can learn embeddings of *types of nodes*.⁵

¹ Implementation used: <https://github.com/palash1992/GEM-Benchmark>.

² Implementation used: <https://github.com/benedekrozemberczki/role2vec>.

³ Implementation used: <https://github.com/phanein/deepwalk>.

⁴ Implementation used <https://github.com/aditya-grover/node2vec>.

⁵ Implementation used: <https://github.com/benedekrozemberczki/role2vec>.

Deep Learning-Based Approaches. Such approaches use deep neural network models to generate node embeddings. In this research paper, we studied three deep learning-based models in total:

- *Deep Neural Networks for Graph Representations (DNGR)* [24] uses random surfing to build a normalized node co-occurrence matrix and employs a stacked denoising autoencoder to learn node embeddings.⁶
- *Structural Deep Network Embedding (SDNE)* [25] finds neighbors by means of 1st and 2nd order proximity and learns node embeddings via autoencoders.⁷
- *Graph sample and aggregate GraphSAGE* [26] is a multi-layered graph convolutional neural network, which represents nodes internally by aggregating their sampled neighborhoods and utilizes a random-walk-based cost function for unsupervised learning. GraphSAGE performs the convolution in the graph space. It uses either mean-based, GCN-based, LSTM-based, mean pooling or max pooling models for aggregation.⁸

Large-Scale Information Network Embedding. *LINE* [11] creates embeddings that preserve 1st-order and 2nd-order proximity which are represented as joint and conditional probability distributions respectively.⁹

3 Preliminaries

3.1 Datasets

We employ three open datasets commonly used when evaluating trust-based recommender systems, i.e., Epinions [5], Ciao [8], and FilmTrust [9]. For all three datasets, we create an unweighted trust network, in which each node represents a user, and each directed edge denotes a trust relationship between two users. The trust network is then an adjacency matrix \mathbf{A} where $\mathbf{A}_{u,v}$ is 1 in case of a trust link between u and v , and 0 otherwise. As a result of preliminary experiments, we found that most of the approaches achieved better accuracy results with an undirected network. One possible explanation is that removing the edge direction increases the average number of edges for each node and reduces the sparsity of the adjacency matrix. Moreover, some approaches are not able to consider link direction by design. Therefore, we convert the trust network to an undirected network in all of our experiments by removing edge direction, thus making \mathbf{A} symmetric. Furthermore, we create a ratings matrix \mathbf{R} , where each non-zero entry $\mathbf{R}_{u,i}$ represents a rating given by a user u to an item i . Table 1 shows basic statistics for all three datasets.

Dataset Splits. We split each dataset into two sets: warm-start users, i.e., users with >10 ratings and cold-start users, i.e., users with ≤ 10 item ratings. While

⁶ Implementation used: <https://github.com/ShelsonCao/DNGR>.

⁷ Implementation used: <https://github.com/suanrong/SDNE>.

⁸ Implementation used: <https://github.com/williamleif/GraphSAGE>.

⁹ Implementation used: <https://github.com/tangjianpku/LINE>.

Table 1. Dataset statistics.

Dataset	#Users	#Items	#Edges	#Ratings	Graph density
Epinions	49,288	139,738	487,183	664,824	2×10^{-4}
Ciao	19,533	16,121	40,133	72,665	1.85×10^{-3}
Filmtrust	1,642	2,071	1,853	35,497	2.43×10^{-3}

we use the subset of warm-start users as a validation set for hyperparameter optimization concerning recommendation accuracy, the subset of cold-start users is used as a test set for measuring algorithm performance. Table 2 reports the number of cold-start and warm-start users in our datasets.

Table 2. Number of users per dataset split.

Dataset	Users with ratings		Users with ratings & trust	
	Warm-start	Cold-start	Warm-start (Validation set)	Cold-start (Test set)
Epinions	14,769	25,393	14,769	25,393
Ciao	1,020	16,591	571	2,124
Filmtrust	963	545	499	241

3.2 Experimental Setup

The initial directed trust network is converted to an undirected network by removing edge directions. The resulting undirected symmetric \mathbf{A} is then used as an input for the graph embedding methods, which, as a result, create a d -dimensional embedding for each node (i.e., user) in the graph.

Recommendation Strategy. After generating the embedding for each node in the graph, a similarity matrix \mathbf{S} is created based on the pairwise cosine similarity between nodes' embeddings. Recommendations are generated in a kNN manner where we find the k -nearest neighbors N_k (i.e., k most similar users) for the target user u_t using the similarity matrix \mathbf{S} . We use $k = 40$ across all of our experiments as in [13]. Then, we assign a score for all items the users in N_k have interacted with:

$$score(i, u_t) = \sum_{v \in N_k} S_{u_t, v} \cdot R_v(i), \quad (1)$$

where $R_v(i)$ corresponds to the rating assigned by the user v to the item i and $S_{u_t, v}$ corresponds to the similarity score in \mathbf{S} between target user u_t and the neighbor user v from N_k . For each target user u_t with n rated items, we recommend 10 items ranked according to Eq. 1 and compare them with the actual rated items.

Evaluation Metrics. Previous research has shown [27] that accuracy may not always be the only or the best criteria for measuring recommendation quality. Typically, there is a trade-off between accuracy, novelty, and diversity since users also like to explore novel and diverse content depending on the context. Therefore, in our work, we examine both novelty and diversity as well as accuracy. In particular, in our experimental setup, we use the following four accuracy and beyond-accuracy metrics for evaluation.

Normalized Discounted Cumulative Gain (nDCG@n) – a ranking-dependent metric measuring recommendation accuracy based on the Discounted Cumulative Gain (DCG) measure [28].

Novelty@n – corresponds to a recommender’s ability to recommend long-tail items that the target user has probably not yet seen. We compute novelty using the Expected Popularity Complement (EPC) metric [29].

Diversity@n – describes how dissimilar items are in the recommendation list. We calculate it as the average dissimilarity of all pairs of items in the recommendation list [30]. More specifically, we use cosine similarity to measure the dissimilarity of two items based on doc2vec embeddings [31] learned using the item vector from \mathbf{R} where each rating is replaced with the user id.

User Coverage – defined as the number of users for whom at least one item recommendation could have been generated divided by the total number of users in the target set [5].

Baseline Approaches. We evaluate the graph embeddings approaches against five different baselines:

- *Explicit directed trust ($Trust_{dir}$)* is a naive trust-based approach that uses the unweighted, directed trust network’s adjacency matrix for finding user’s nearest neighbors, i.e. $\mathbf{S} = \mathbf{A}$.
- *Explicit undirected trust ($Trust_{undir}$)* is similar to $Trust_{dir}$ but converts the network to an undirected one by ignoring the edge direction, thus making \mathbf{A} symmetric, i.e. $\mathbf{S} = \mathbf{A}_{undir}$.
- *Explicit trust with Jaccard ($Trust_{jac}$)* uses the Jaccard index on the undirected trust network \mathbf{A}_{undir} . \mathbf{S} is a result of calculating the pairwise Jaccard index on \mathbf{A}_{undir} . The intuition behind this algorithm is that two users are treated as similar if they have adjacent users in common, i.e., trustors and trustees.
- *Explicit trust with Katz similarity ($Trust_{Katz}$)* [13] exploits regular equivalence, a concept from network science by using Katz similarity in order to model transitive trust relationships between users.
- *Most Popular (MP)* is a non-personalized approach in recommender systems, which recommends the most frequently rated items.

4 Results

Table 3 shows our results in terms of nDCG, novelty, diversity and user coverage for $n = 10$ recommendations on cold-start users (test set). The reported

results depict those hyperparameter configurations¹⁰, which achieve the highest recommendation accuracy on warm users (validation set).

4.1 Accuracy Results

To ease the interpretation of the evaluation results across all three datasets, we rank the results by nDCG and compute the average of these ranks. Correspondingly, in the Rank_{nDCG} column, we show the resulting average rank for the three datasets for recommendation accuracy.

We can observe that RW-based approaches, especially Node2vec and DeepWalk, are the best performing approaches on all three datasets. In most cases, approaches based on graph embeddings outperform the baselines, except for Trust_{jac} on Epinions. Contrary to a study conducted in [13], Trust_{jac} achieves higher accuracy in comparison with Trust_{Katz}. The reason is that in the present work, we convert the trust network to an undirected network, i.e., do not consider the direction of the trust edge. HOPE and Laplacian Eigenmaps perform best among the factorization-based approaches; LINE shows a good performance on all three datasets concerning all three metrics, and GraphSAGE is the best deep learning approach. SDNE does not perform well in our experiments, which we attribute to not exploring a sufficiently broad range of hyperparameters.

4.2 Beyond-Accuracy Results

Novelty, Diversity, and User Coverage. Being superior in the case of Rank_{nDCG}, Node2vec also achieves high novelty and diversity scores. Plus, it performs similarly or better than other RW-based methods across all three datasets. Factorization-based approaches show average performance concerning both novelty and diversity, except for GF, which scores very low on novelty and above average on diversity. DL approaches show average to below-average performance on novelty and average performance on diversity. Trust-based baselines achieve high novelty scores in general and, not surprisingly, MP has a high diversity score and the worst novelty score out of all approaches. Since all graph embedding approaches create a latent representation of each user in a trust network using it to generate a set of item recommendations, there are no differences among them in user coverage. Except for MP, all baselines result in lower user coverage than the graph embedding approaches. Since MP provides the same list of recommendations to all users, it always has a maximum user coverage.

Evaluation Metrics and User Preferences. Table 3 reports only mean values for each of the approaches. However, we store individual nDCG, novelty, and diversity values for each target user and each approach and dataset. By computing the Kendall rank correlation coefficient (Bonferroni corrected, $p < 0.01$) on non-zero metrics values for all approaches, we can get an insight into user

¹⁰ Details on the hyperparameter optimization can be found at: <https://github.com/tduricic/trust-recommender/blob/master/docs/hyperparameter-optimization.md>.

Table 3. Evaluation results on cold-start users for different trust-based CF approaches for $n = 10$ recommendations concerning nDCG, novelty, diversity, and user coverage comparing approaches from five different algorithm families across three different datasets. Values marked with * denote that the corresponding approach was significantly better than every other approach with respect to the appropriate metric according to a Wilcoxon signed-rank test (Bonferroni corrected, $p < 0.01$). Rank_{nDCG} is calculated by summing nDCG-based ranks across the datasets and re-ranking the sums.

Cat.	Approach	Rank _{nDCG}	Epinions				Ciao				Filmtrust			
			nDCG	Nov.	Div.	UC	nDCG	Nov.	Div.	UC	nDCG	Nov.	Div.	UC
Baseline	Trust _{dir}	15	.0245	.0060	.6006	59.2%	.0140	.0028	.3700	3.9%	.2655	.0313	.2784	30.3%
	Trust _{undir}	15	.0260	.0063	.5960	97.0%	.0127	.0045	.3632	11.4%	.2739	.0284	.2731	42.0%
	Trust _{jac}	11	.0373	.0056	.6548	99.9%	.0176	.0027	.3996	12.8%	.3387	.0369	.2266	36.1%
	Trust _{Katz}	12	.0290	.0046	.6979		.0158	.0026	.3842	13.0%	.3681	.0322	.2185	42.9%
	MP	17	.0134	.0015	.7621*		.0135	.0012	.5666	100%	.3551	.0137	.1672	100%
Factorization	LLE	7	.0309	.0044	.6977		.0239	.0036	.4013		.3649	.0159	.1926	
	LE	3	.0318	.0045	.6961		.0231	.0034	.3962		.3715	.0161	.1853	
	GF	14	.0138	.0023	.7024		.0154	.0022	.3970		.3686	.0154	.1945	
	HOPE	3	.0331	.0047	.6728		.0220	.0033	.3956		.3718	.0158	.1827	
	GraRep	7	.0298	.0042	.6704		.0209	.0030	.3974		.3694	.0147	.1859	
RW	Node2vec	1	.0413	.0064	.6581	100 %	.0228	.0036	.4042		.3904	.0151	.2235	
	DeepWalk	2	.0435*	.0067*	.6707		.0247*	.0037	.3992	13.1 %	.3654	.0152	.1950	44.2 %
	Role2vec	6	.0363	.0054	.6910		.0149	.0024	.3933		.3695	.0151	.1919	
DL	DNGR	10	.0353	.0051	.6869		.0197	.0031	.4023		.3583	.0142	.1959	
	SDNE	12	.0184	.0022	.7412		.0175	.0028	.3921		.3687	.0152	.2003	
	GS	7	.0325	.0047	.6810		.0216	.0031	.3963		.3678	.0151	.1883	
LINE	LINE	5	.0407	.0063	.6566		.0222	.0033	.3992		.3667	.0150	.1947	

preferences for each dataset. In this manner, we observe a statistically significant positive mean correlation across all three datasets between nDCG and novelty, ranging from 0.43 on Epinions to 0.36 on Filmtrust. This suggests that users on all three platforms prefer recommendations with higher novelty, especially on Epinions. We also observe a statistically significant mean negative correlation between diversity and novelty on Epinions (-0.15), which suggests that on this platform, more novel content seems to be less diverse.

5 Conclusions and Future Work

In this work, we explored the utility of graph embedding approaches from four method families to generate latent user representations for trust-based recommender systems in a cold-start setting. We found that random-walk-based approaches, (i.e., Node2vec and DeepWalk), consistently achieve the best accuracy. We additionally compared the methods concerning novelty, diversity, and user coverage. Our results showed that again, Node2vec and DeepWalk scored high on novelty and diversity. Thus, they can provide a balanced trade-off between the three evaluation metrics. Moreover, our experiments showed that we can increase the user coverage of recommendations when we utilize graph embeddings in a k -nearest neighbor manner. Finally, a correlation analysis between the

nDCG, novelty, diversity scores revealed that in all three datasets, users tend to prefer novel recommendations. Hence, on these datasets, recommender systems should offer a good tradeoff between accuracy and novelty. This could also explain the superior performance of the random-walk based approaches and we plan to investigate this in more detail in follow-up work.

Limitations and Future Work. One limitation of this study is that we treated the trust networks as undirected while, in reality, they are directed. This may have resulted in loss of information, and as such, we aim to further explore how to preserve different properties of trust networks (e.g., asymmetry). Moreover, it is possible that we did not examine an ample enough space of hyperparameters, which might have resulted in a more reduced performance of some of the approaches, e.g., SDNE. Furthermore, we aim to explore node properties of k -nearest neighbors for all methods to find and interpret the critical node properties preserved by the graph embeddings, which impact the recommendation accuracy, thus providing a better understanding of the complex notion of trust. Finally, we aim to incorporate user features obtained from the rating matrix into graph embeddings learned on the trust network.

References

1. Schein, A.I., Popescul, A., Ungar, L.H., Pennock, D.M.: Methods and metrics for cold-start recommendations. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 253–260. ACM (2002)
2. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) The Adaptive Web. LNCS, vol. 4321, pp. 291–324. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72079-9_9
3. Fazeli, S., Loni, B., Bellogin, A., Drachsler, H., Sloep, P.: Implicit vs. explicit trust in social matrix factorization. In: Proceedings of the 8th ACM Conference on Recommender Systems, RecSys 2014, pp. 317–320. ACM (2014)
4. Lathia, N., Hailles, S., Capra, L.: Trust-based collaborative filtering. In: Karabulut, Y., Mitchell, J., Herrmann, P., Jensen, C.D. (eds.) IFIPTM 2008. ITIFIP, vol. 263, pp. 119–134. Springer, Boston, MA (2008). https://doi.org/10.1007/978-0-387-09428-1_8
5. Massa, P., Avesani, P.: Trust-aware collaborative filtering for recommender systems. In: Meersman, R., Tari, Z. (eds.) OTM 2004. LNCS, vol. 3290, pp. 492–508. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30468-5_31
6. O’Donovan, J., Smyth, B.: Trust in recommender systems. In: Proceedings of the 10th International Conference on Intelligent User Interfaces, IUI 2005, pp. 167–174. ACM (2005)
7. Kim, Y.A.: An enhanced trust propagation approach with expertise and homophily-based trust networks. *Knowl. Based Syst.* **82**, 20–28 (2015)
8. Tang, J., Gao, H., Liu, H.: mTrust: discerning multi-faceted trust in a connected world. In: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, pp. 93–102. ACM (2012)

9. Guo, G., Zhang, J., Yorke-Smith, N.: A novel Bayesian similarity measure for recommender systems (2013)
10. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: a survey. *Knowl.-Based Syst.* **151**, 78–94 (2018)
11. 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*, pp. 1067–1077. *International World Wide Web Conferences Steering Committee* (2015)
12. Zhang, Y.C., Ó Séaghdha, D., Quercia, D., Jambor, T.: Auralist: introducing serendipity into music recommendation. In: *Proceedings of the 5th ACM Conference on Web Search and Data Mining (WSDM-12)* (2012)
13. Duricic, T., Lacic, E., Kowald, D., Lex, E.: Trust-based collaborative filtering: tackling the cold start problem using regular equivalence. In: *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018*, pp. 446–450. *ACM* (2018)
14. Hamilton, W.L., Ying, R., Leskovec, J.: Representation learning on graphs: methods and applications. *IEEE Data Eng. Bull.* **40**(3), 52–74 (2017)
15. Ahmed, A., Shervashidze, N., Narayanamurthy, S., Josifovski, V., Smola, A.J.: Distributed large-scale natural graph factorization. In: *Proceedings of the 22nd International Conference on World Wide Web*, pp. 37–48. *ACM* (2013)
16. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: *Advances in Neural Information Processing Systems*, pp. 585–591 (2002)
17. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. *Science* **290**(5500), 2323–2326 (2000)
18. Ou, M., Cui, P., Pei, J., Zhang, Z., Zhu, W.: Asymmetric transitivity preserving graph embedding. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1105–1114. *ACM* (2016)
19. Cao, S., Lu, W., Xu, Q.: Grarep: learning graph representations with global structural information. In: *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 891–900. *ACM* (2015)
20. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: online learning of social representations. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2014*, pp. 701–710. *ACM* (2014)
21. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *Advances in Neural Information Processing Systems 26*, pp. 3111–3119. *Curran Associates Inc.* (2013)
22. 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* (2016)
23. Ahmed, N.K., et al.: Learning role-based graph embeddings. *arXiv e-prints. arXiv:1802.02896*, February 2018
24. Cao, S., Lu, W., Xu, Q.: Deep neural networks for learning graph representations (2016)
25. 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*, pp. 1225–1234. *ACM* (2016)
26. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: *Advances in Neural Information Processing Systems*, pp. 1024–1034 (2017)

27. Kaminskias, M., Bridge, D.: Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems. *ACM Trans. Interact. Intell. Syst.* **7**, 2:1–2:42 (2016)
28. Järvelin, K., Price, S.L., Delcambre, L.M.L., Nielsen, M.L.: Discounted cumulated gain based evaluation of multiple-query IR sessions. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) *ECIR 2008*. LNCS, vol. 4956, pp. 4–15. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78646-7_4
29. Vargas, S., Castells, P.: Rank and relevance in novelty and diversity metrics for recommender systems. In: *Proceedings of the Fifth ACM Conference on Recommender Systems*, pp. 109–116 (2011)
30. Smyth, B., McClave, P.: Similarity vs. diversity. In: Aha, D.W., Watson, I. (eds.) *ICCBR 2001*. LNCS (LNAI), vol. 2080, pp. 347–361. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44593-5_25
31. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *Advances in Neural Information Processing Systems*, pp. 3111–3119 (2013)



Neural Spike Sorting Using Unsupervised Adversarial Learning

Konrad A. Ciecierski^(✉) 

Research and Academic Computer Network, Warsaw, Poland
konrad.ciecierski@gmail.com

Abstract. During the analysis of neurobiological signals acquired during neurosurgical procedures such as Deep Brain Stimulation, one focuses mainly on the signal's background and spiking activity. Spikes, the electrical impulses generated by neuron cells, are events lasting about 1.5 ms with voltage typically well below 200 μV . As the shape of spike derives from the physical construction of the neuron's cell membrane and its distance from the recording point, one can infer that spikes of different shapes were generated by different neuron cells. This information can be beneficial during the analysis of the acquired signals, and the procedure of grouping spikes according to their shapes, is called spike sorting. Typically in this procedure, there are defined various attributes characterizing spike shape, and subsequently, the standard clustering method is used to produce groups of spikes of different shapes. This may cause some difficulties that arise both from the definition of good discriminating attributes as well as from the fact that one does not know beforehand how many clusters of shapes to expect. The approach presented in this paper is different, using the fully unsupervised and explainable AI, it generates groups of spike shapes in a fully automatic way. No attributes have to be defined nor the expected number of shape clusters provided. Spike waveforms are fed into the autoencoder that has defined additional constraints on its latent layers. After training, data obtained from those layers can be used for the purpose of spike shape clustering. Beside shape class, one also receives information on how much given spike shape differs from the mean shape of its class.

Keywords: Spike sorting · Explainable AI · Autoencoder · Unsupervised adversarial learning · Distribution enforcement · Weighted MSE loss

Introduction

In the process of many Deep Brain Stimulation (DBS) surgeries, the brain activity is recorded using thin microelectrodes [1]. Signals acquired by those electrodes help neurosurgeons in precise localization of the target of the surgery. Such signals are mainly analyzed in two ways: first focusing on signal's background and second dealing with neural spikes detected in the signal [2]. As the recording

microelectrode can register electrical activity in a radius of about $50 \mu\text{ m}$ from its lead [1], the activity of nearest neurons is registered as distinct spikes while a summary of the activity of farther neurons is registered as background noise.

For each neuron cell, the shape of spikes it generates is generally constant as it comes from the concentration and distribution of ion channels in its cell membrane [3]. Spikes registered from neurons being farther from the recording lead have also amplitude lower than those being near [1]. From this comes the assumption that spikes of different shapes do originate in different neuron cells [1]. Information about the number of active cells in the vicinity of the electrode might be beneficial in the identification of the area in which the electrode is located. To provide such information i.e., to cluster spike shapes into classes, the spike sorting is used. There are many papers regarding spike sorting approaches [4, 5]. The main difficulty in the classical spike sorting is the definition of attributes that discriminate spike shapes in a good manner without the prior knowledge about the number of spike shape clusters expected in a given recording.

One of the approaches to classification or clustering of data for which there are no known attributes is an autoencoder neural network [6] with an adequately small latent layer. In the purest form, such a network consists of two neural networks. First network – encoder – transforms input data into latent space that has dimension much lower than the dimension of the original data. The second network reconstructs the input data from the latent space. Network in which the output of the encoder is the input for the decoder forms autoencoder. During the training of such networks, the goal is to minimize the difference between input and output. In most cases, this difference, the loss, is calculated as MSE [6]. The latent layer acts as a bottleneck for the information that passes through the network. After training, one might attempt to use data from the latent layer as a set of attributes characterizing data fed into the encoding part of the autoencoder. The difficulty lies in the fact that there are no constraints put on the latent layer. Latent data is self-organized, and only for specific input data, one can find here any naturally occurring clusters.

To improve this situation the variational autoencoders (VAE) were defined [7]. In those autoencoders, the loss is augmented in such way as to reflect how much distribution of the latent layer differs from the normal distribution. During the training process, as the loss is minimized, the distribution of the latent layer shifts towards the normal distribution. This is feasible for data of one kind, where one attempts to represent it by normally distributed latent layer. It shows which data is typical and which one lies on tails of the distribution curve, i.e., which is typical and which one is not. Another limitation of VAE is that it enforces only the normal distribution.

The ability to enforce any chosen distribution upon the latent layer has come with the concept of adversarial autoencoders [8]. Here beside the encoder and decoder, there is another network. This network is trained to discriminate data having specified distribution. The discriminating network takes as input, data of size identical to this given by encoder's output. Discriminator's output pro-

duces, as a result, one if its input data has desired distribution, 0 otherwise. For simplicity disregarding the minibatch [6] dimension, the adversarial training works in the following way. a) data with desired distribution is fed into discriminating network which is trained to give one as an output b) input data is fed into encoder; subsequently the encoder's output is fed into discriminator, which is trained to give 0 as an output c) encoder is trained in such way to provide output for which discriminator gives one as output. Summarizing, in step a) discriminator learns to produce 1 for data with desired distribution and in step b) discriminator learns to treat encoder's output as not having desired distribution. In crucial step c), encoder learns to produce output than can fool discriminator, i.e., output that is so close to desired distribution that discriminator outputs 1. Of course, besides described adversarial training, autoencoder consisting of encoder paired with a decoder is still trained to minimize the difference between encoder's input and decoder's output.

Using adversarial autoencoder, one can approach the problem of spike sorting in an unsupervised way. Let us assume that encoder for single input produces output composed of two parts. First part has dimension dim_c while dimension of second part is dim_g . For each of those parts, there is a defined separate discriminator network. The first discriminator enforces the first part of the encoder's output to have categorical distribution [9]. The second discriminator enforces the remaining part of the encoder's output to have the normal distribution. In such a way, it forces the adversarial autoencoder to represent input data in the latent layer in no more than dim_c classes where variability in each class is defined as a normally distributed vector of size dim_g .

1 Construction of the Autoencoder

1.1 Input Data

Waveforms of spikes were obtained from recordings done during DBS surgeries for Parkinson's Disease and Dystonia [1]. Sampling frequency during data acquisition was 24 kHz; each spike waveform was recorded in the form of 48 samples, i.e., as 2 ms of data. The full dataset consists of 330963 spikes. Spikes were divided into training, validation, and test sets in proportions 80:10:10, resulting in training data having 264770 spikes, validation data having 33096 spikes, and finally test data with 33097 spikes. All detected spikes were aligned to have their maximal value at sample 12 so relatively to spike maximal point the registered time span ranges from $-0.5 \mu\text{s}$ to $1.5 \mu\text{s}$. Originally the recorded data is in micro-volts and fits in the range $[-500 \mu\text{V}, 500 \mu\text{V}]$. Before being fed into the network, the data is normalized to fit in the range $[-1, 1]$.

1.2 Architecture

In the construction of the autoencoder, it has been assumed that spikes are to be separated into no more than ten classes, and that content of each class is to

be described as a normally distributed vector from \mathbb{R}^3 . Assuming that minibatch size is N , the full autoencoder consists of the following parts.

The encoder is a neural network taking as input tensor of dimensions $(N, 48)$ and producing as an output pair of tensors with dimensions $(N, 10)$ and $(N, 3)$. Let us define first output tensor as categorical head and second as Gaussian head of the encoder.

$$enc(input) = (categorical_head, gaussian_head)$$

$$where\ input \in \mathbb{R}^{(N,48)},\ categorical_head \in \mathbb{R}^{(N,10)},\ gaussian_head \in \mathbb{R}^{(N,3)}$$

The encoder takes input tensor of shape $(N, 48)$ and puts it through two dense layers [6]. Each of the two dense layers, has 100 output neurons and uses ELU [6] activation function. After the second dense layer, there are two dense layers connected to it. Those two layers use the linear activation function. The first layer has ten output neurons and produces tensor of shape $(N, 10)$, which is the categorical head of the encoder. The second one has only three output neurons and produces tensor of shape $(N, 3)$, which is the Gaussian head of the encoder. The architecture of the encoder is shown in Fig. 1.

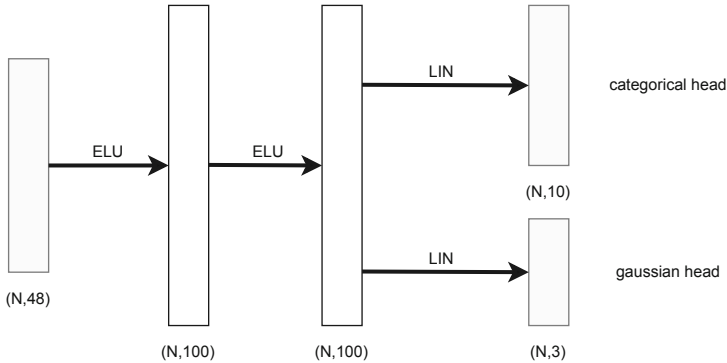


Fig. 1. Architecture of the encoder network

The decoder acts in a way reverse to the encoder. It takes the encoder output in the form of two tensors of shapes $(N, 10)$ and $(N, 3)$ to produce the output of shape identical to encoders input, i.e. $(N, 48)$.

$$dec(categorical_input, gaussian_input) = output$$

$$where\ categorical_input \in \mathbb{R}^{(N,10)},\ gaussian_input \in \mathbb{R}^{(N,3)},\ output \in \mathbb{R}^{(N,48)}$$

The architecture of the decoder (see Fig. 2) is slightly more complicated. There are two dense input networks, one for each of the inputs. The first network

takes tensor of shape $(N, 10)$ as an input and outputs tensor of shape $(N, 100)$. The second input network takes tensor of shape $(N, 3)$ as an input and also outputs tensor of shape $(N, 100)$. In the subsequent step, two resulting tensors are added to form a single tensor of shape $(N, 100)$. This tensor is fed into a dense network with 100 output neurons and ELU activation function. Finally, the last layer is also the dense layer but with 48 output neurons and TANH [6] activation function. The choice of the TANH activation function instead of often used Sigmoid [6] function is caused by the fact that original normalized data is in the range $[-1, 1]$ which is identical to the output range of the Tanh function.

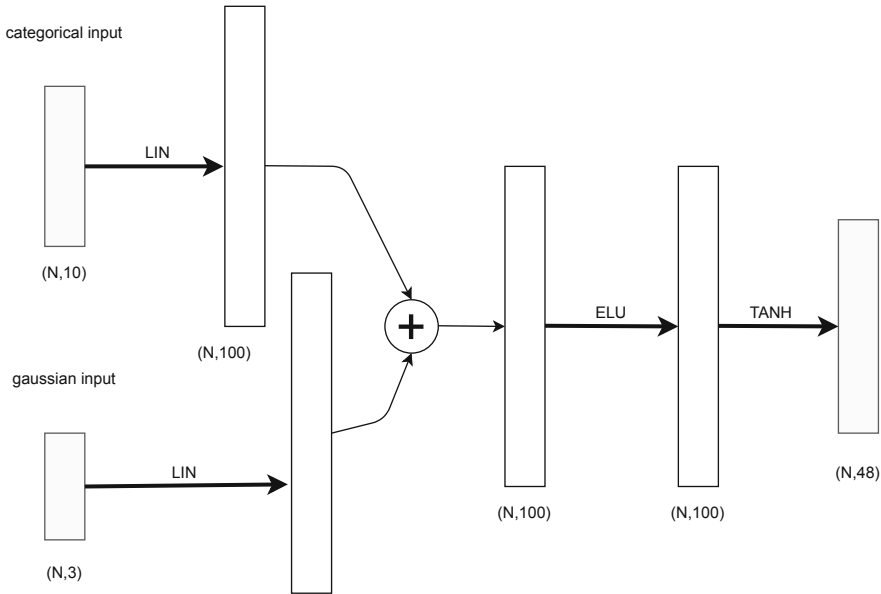


Fig. 2. Layer composition of the decoder

The categorical and Gaussian discriminators are very similar; the only difference between them is the shape of the input tensor. In the case of the categorical discriminator, the shape of the input tensor is identical as the categorical head of the encoder, i.e. $(N, 10)$. Analogously the shape of the input tensor for Gaussian discriminator matches the Gaussian head and is $(N, 3)$. The discriminator consists of three dense layers. The first two layers have 100 output neurons and use ELU activation function. The last layer has two output neurons and is used for one-hot encoding of the discriminator output.

$$dsc_c(\text{categorical_input}) = \text{output}$$

$$dsc_g(\text{gaussian_input}) = \text{output}$$

$$\text{where } \text{categorical_input} \in \mathbb{R}^{(N,10)}, \text{ gaussian_input} \in \mathbb{R}^{(N,3)}, \text{ output} \in \mathbb{R}^{(N,2)}$$

The architecture of the categorical discriminator is shown in Fig. 3.

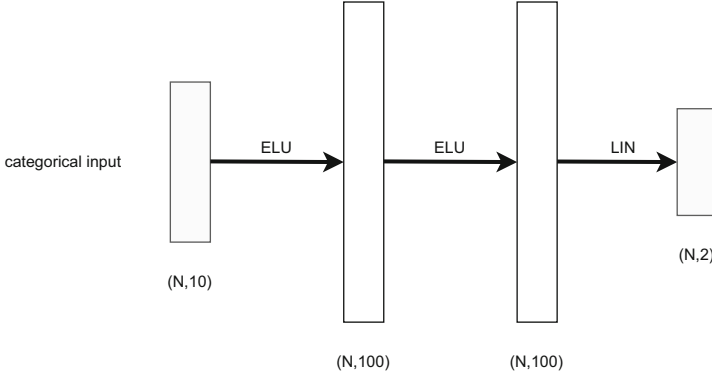


Fig. 3. Layer composition of the categorical discriminator

2 Training

2.1 Training Phases for Single Epoch

For each minibatch, the training consists of three distinct steps.

The Reconstruction Phase. In this phase, autoencoder is trained in a way that is similar to classical autoencoder. For the given input, the loss function is defined as $MSE_w(input, dec(enc(input)))$. The MSE_w being a custom loss function is based upon the standard MSE function. Each spike waveform is 48 samples long, the sampling frequency is 24 kHz, and so the spike waveform span is 2 ms. Not all parts of the spike are equally important for the sorting process [5] let us so define MSE_w for minibatch as follows:

$$in = [x_{i,j}], i = 0 \dots N - 1, j = 0 \dots 47; \quad out = dec(enc(in))$$

$$SE_p(in, out, b, e) = \sum_{i=0}^{N-1} \sum_{j=b}^e (in_{i,j} - out_{i,j})^2$$

$$SE_w(in, out) = SE_p(in, out, 0, 5) + 50 SE_p(in, out, 6, 17) + SE_p(in, out, 18, 47)$$

$$MSE_w(in, out) = \frac{SE_w(in, out)}{48 N}$$

The coefficient of 50 used in the definition of $SE_w(in, out)$ has been found experimentally. The MSE_w focuses on time span between $-250 \mu s$ and $+250 \mu s$ as shown in Fig. 4a. The result that is achieved by using such weighted loss function can be seen in Fig. 4b where the main spike shape is recreated very

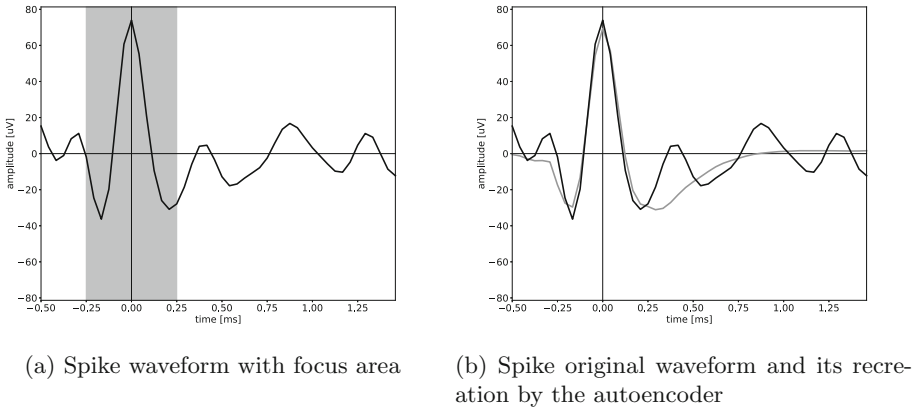


Fig. 4. Role of spike focus area

closely while the distortions present after time 0.5 ms are replaced by general waveform learned from spikes during the training process.

The Adversarial Phase. In this phase, the discriminators are trained. Firstly for the given *input* of size $(N, 48)$ the *categorical.head* and *gaussian.head* are calculated. In the next four steps, the total discriminatory loss is calculated. The $loss_{HC}$ is calculated as cross-entropy loss between the output of the $dsc_c(\text{categorical.head})$ and zero filled vector of size N . The $loss_{HG}$ is calculated as cross-entropy loss between the output of the $dsc_g(\text{gaussian.head})$ and zero filled vector of size N . At this step, two tensors are generated, first tensor tc is of size identical to *categorical.head* and is filled with randomly generated numbers sampled with categorical distribution along horizontal the axis. The second tensor tg is of size identical to *gaussian.head* and is filled with random numbers sampled with the normal distribution. The $loss_{SC}$ is calculated as cross-entropy loss between the output of the $dsc_c(tc)$ and one filled vector of size N . The $loss_{SG}$ is calculated as cross-entropy loss between the output of the $dsc_g(tg)$ and one filled vector of size N . Finally the total discriminatory loss $loss_{HC} + loss_{HG} + loss_{SC} + loss_{SG}$ is used for training of the categorical and Gaussian discriminators. It is expected that discriminators will return class 0 for data produced by the encoder while returning class 1 for the data sampled with desired distributions. cross-entropy loss was calculated using `CrossEntropyLoss` implementation provided by PyTorch [10].

The Generation Phase. The purpose of this phase is to train encoder to enforce categorical distribution on *categorical.head* and normal distribution on *gaussian.head*. This step is similar to the previous one in this that in each epoch, it uses the same calculated *categorical.head* and *gaussian.head* tensors. The $loss_{HC}$ and $loss_{HG}$ are however calculated differently. The $loss_{HC}$ is calculated as cross-entropy loss between the output of the $dsc_c(\text{categorical.head})$ and one filled vector of size N . The $loss_{HG}$ is calculated as cross-entropy loss between

the output of the $dsc_g(\text{gaussian_head})$ and one filled vector of size N. Finally, the total generative loss $loss_{HC} + loss_{HG}$ is used for the training of the encoder. The encoder is trained to produce heads that discriminators would recognize as having proper distributions.

2.2 Training of the Autoencoder

For all four networks i.e., encoder, decoder, and both discriminators, for the weight optimization, the Adam [11] algorithm has been chosen. The initial learning rate for encoder and decoder was set to $1e-4$, while for discriminators, the $1e-2$ value was set. Additionally, for generalization, for the encoder and decoder, the weight decay was set to 0.0005. The stop condition in case of training of adversarial autoencoders is not as clearly defined as in many other applications where the lowest loss for the validation data is the golden standard [6]. Here one of the goals is, of course, the low validation loss on reconstruction phase but the main goal is the achievement of a stable equilibrium between adversarial and generative loss. All those conditions have been achieved during training as can be seen in Fig. 5a and especially in Fig. 5c and d. At epoch 3000, the recreation loss was 0.0020818 validation data (V) and 0.0020843 for test data (T). The respective adversarial losses were 2.3282955 (V) and 2.3300469 (T). Finally, the generative losses were 1.5815901 (V) and 1.5819962 (T). Important information can also be found in Fig. 5b where one can find how many of the classes represented by the categorical head, were actually used by the autoencoder.

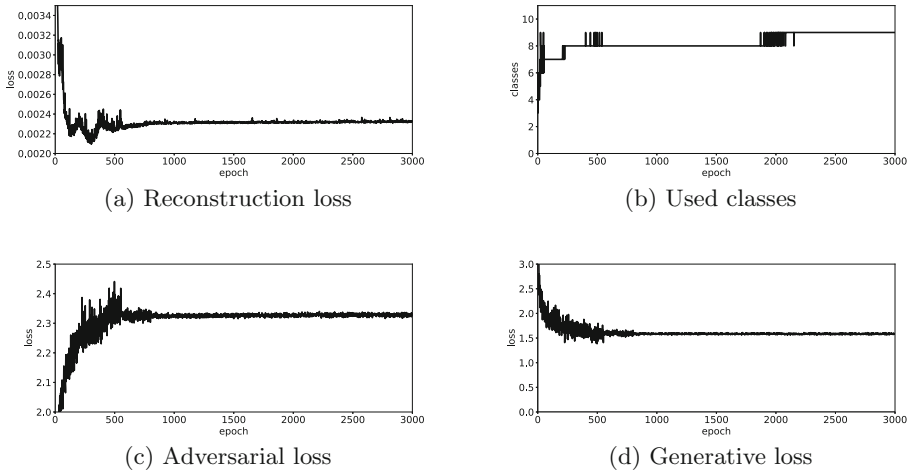


Fig. 5. Losses and used class count for validation data

3 Evaluation and Summary

During training, only nine out of 10 possible classes were used (see Fig. 5b). Spikes of all those classes were also found in validation and test data. Below, shown are three out of nine found classes. For each class shape, its center is shown with a thick black line; this is the shape for a particular class when all values in the Gaussian input of the decoder are set to 0. Gray shapes were generated by putting random numbers from $\mathcal{N}(0, 0.75)$ into the Gaussian input of the decoder. In fact, the decoder alone has become a GAN network [12] generating various spike shapes.

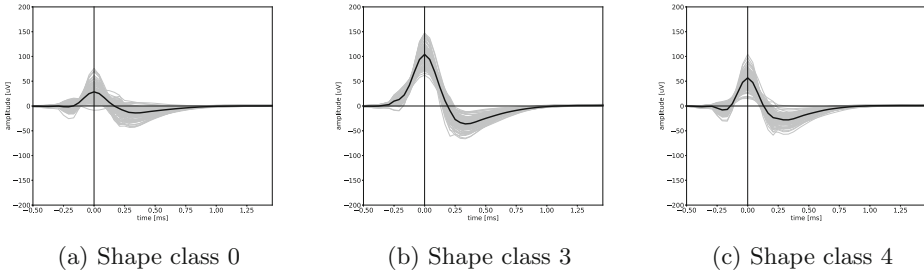


Fig. 6. Selected shape classes found in test data

The question that now has to be answered is if the Gaussian head for the test data has a normal distribution. Table 1 shows the p-values for the D’Agostino and Pearson’s test [13].

Table 1. p-values for data in Gaussian head for classes found in test data. Gaussian head is represented here as $gh \in \mathbb{R}^3$

Class ID	Count	p-value for $gh[0]$	p-value for $gh[1]$	p-value for $gh[2]$
0	6133	2.042e-07	2.296e-08	3.290e-25
1	4648	9.557e-25	2.173e-09	9.937e-31
2	2965	1.223e-05	5.464e-13	2.833e-22
3	1895	9.965e-07	<u>3.931e-01</u>	2.052e-03
4	5198	1.624e-12	7.758e-11	<u>5.241e-02</u>
5	3581	7.687e-46	5.608e-07	2.595e-05
6	2	–	–	–
7	4221	2.646e-41	3.097e-04	4.114e-32
8	4454	3.731e-55	9.075e-03	8.396e-20

Clearly, for all classes where the calculation of p-value was feasible, in all but two underlined cases, the p-value is well below 0.005, i.e., the data is normally

distributed. Additionally, one case where the p-value is above 0.05 is a narrow miss being 0.052.

It has been shown that adversarial autoencoder can be used for spike shape classification. Additionally, when analyzing spikes detected in a single recording, where their variety is much smaller than in the training set, obtained results might be used as a basis for further research. For example, one might try to find smaller, more specific clusters of spike shapes within classes found by autoencoder.

It must be remarked here that in the architecture of the adversarial autoencoder, setting the sizes of heads sets only the upper limits for the network. As the learning is in this particular case unsupervised, there is no guarantee that the number of classes found will be equal to the size of the categorical head. For example, in this paper, the categorical head has size 10, and only 9 classes were found.

Similarly, setting a certain size of the Gaussian head does not guarantee that all its dimensions will be used. During tests, when the size of the Gaussian head was set to 5, two of those dimensions were found to be very close to zero for all training, validation, and test data. They were left unused by the encoder and what is remarkable, further tests showed that their values had almost no influence on the decoder output.

References

1. Israel, Z., Burchiel, K.J.: *Microelectrode Recording in Movement Disorder Surgery*. Thieme Medical Publishers (2004)
2. Ciecierski, K., Raś, Z.W., Przybyszewski, A.W.: Foundations of recommender system for STN localization during DBS surgery in Parkinson's patients. In: Chen, L., Felfernig, A., Liu, J., Raś, Z.W. (eds.) *ISMIS 2012. LNCS (LNAI)*, vol. 7661, pp. 234–243. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34624-8_28
3. Koch, C.: *Biophysics of Computation: Information Processing in Single Neurons*. Oxford University Press (2004)
4. Lewicki, M.S.: A review of methods for spike sorting: the detection and classification of neural action potentials. *Netw. Comput. Neural Syst.* **9**(4), R53–R78 (1998)
5. Quiroga, R.Q., Nadasdy, Z., Ben-Shaul, Y.: Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. *Neural Comput.* **16**(8), 1661–1687 (2004)
6. Goodfellow, I., Bengio, Y., Courville, A.: *Deep Learning*. MIT press, New York (2016)
7. Chen, X., et al.: Variational lossy autoencoder. arXiv preprint [arXiv:1611.02731](https://arxiv.org/abs/1611.02731) (2016)
8. Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., Frey, B.: Adversarial autoencoders. arXiv preprint [arXiv:1511.05644](https://arxiv.org/abs/1511.05644) (2015)
9. Agresti, A., Kateri, M.: *Categorical Data Analysis*. Springer, Heidelberg (2011). https://doi.org/10.1007/978-94-007-0753-5_291
10. Paszke, A., et al.: Automatic differentiation in PyTorch (2017)

11. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
12. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
13. Pearson, E.S., D’Agostino, R.B., Bowman, K.O.: Tests for departure from normality: comparison of powers. *Biometrika* **64**(2), 231–246 (1977)

Digital Signal Processing



Poriferal Vision: Classifying Benthic Sponge Spicules to Assess Historical Impacts of Marine Climate Change

Saketh Saxena¹, Philip Heller¹ (✉), Amanda S. Kahn², and Ivano Aiello²

¹ San Jose State University, San Jose, CA 95192, USA
philip.heller@sjsu.edu

² Moss Landing Marine Laboratories, Moss Landing, CA 95039, USA

Abstract. Sponges and corals are ecologically important members of the marine community. Climate change, while harmful to corals, has historically been favorable to sponges. Sponge population dynamics are studied by analyzing core samples of marine sediment. To date this analysis has been performed by microscopic visual inspection of core cross sections to distinguish spicules (the rigid silica components of sponge skeletons) from the residue of other silica-using organisms. Since this analysis is both slow and error prone, complete analysis of multiple cross sections is impossible.

FlowCam® technology can produce tens of thousands of microphotographs of individual core sample particles in a few minutes. Individual photos must then be classified *in silico*. We have developed a Deep Learning classifier, called Poriferal Vision, that distinguishes sponge spicules from non-spicule particles. Small training sets were enhanced using image augmentation to achieve accuracy of at least 95%. A Support Vector Machine trained on the same data achieved accuracy of at most 86%. Our results demonstrate the efficacy of Deep Learning for analyzing core samples, and show that our classifier will be an effective tool for large-scale analysis.

Keywords: Deep Learning · Sponges · Porifera · Climate

1 Introduction

Sponges (Phylum Porifera) are prolific reef builders [1] and benthic grazers [2, 3]. Many sponges (Phylum Porifera, especially Classes Demospongiae and Hexactinellida) form spicules – the rigid components of their skeletons – composed of silica. Since silica can resist breakdown for millions of years, spicules are part of the geological record, providing insight into the presence and ecological significance of sponges under a variety of climatic conditions. Sponge populations have increased significantly during planetary warm periods, including the Upper Carboniferous period (300 mya) [4], some stages of the Paleozoic (540–250 mya) and Mesozoic (250–65 mya) eras [5], the Triassic/Jurassic transition (200 mya) [6], and the Pliocene warm period (3 mya) [7].

Shallow water reefs are hotspots of marine biodiversity, fringing 1/6 of the world's coastlines [8] and supporting hundreds of thousands of species [9]. Reefs are predominantly coral rather than sponge; the hard carbonate exoskeletons of dead coral polyps accrete to form non-living support structures for living reef components. However, coral reefs are particularly vulnerable to current climate change conditions. Rising temperatures cause bleaching by forcing expulsion of heat-intolerant symbionts [10, 11]. Ocean acidification, driven by increased CO₂ in the atmosphere, slows production of calcium carbonate and accelerates removal of calcium carbonate by dissolution and bioerosion [12–14] in shallow and deep waters; moreover there is mounting evidence that increasing acidity disrupts the life cycle of corals at multiple stages [15]. Goreau et al. have estimated that climate change had destroyed or degraded 25% of all reefs by the year 2000 [16]. The loss of coral reefs can be expected to continue through the century, since atmospheric CO₂ levels are expected to rise to 750 ppm by 2100 [17].

The ongoing degradation of coral reefs, together with the historical success of sponges during warm periods, suggest that sponges may be due for a resurgence in ecological importance, and may replace corals as major reef builders [18]. The ecological and economic importance of reefs motivate research into the mechanics of the historical success of sponges during past warm periods. Although glass sponges do not build significant shallow water reefs, they are similar to reef-building corals in that they form the foundation for unique communities [19], provide three-dimensional habitat [20], are a nursery habitat for many species [21], and enhance local biodiversity [19, 22]. Their historical response to climate change can therefore elucidate present-day coral and sponge population dynamics. Much information can be gained by counting and classifying spicules from smear slides of spicule-bearing core samples (Fig. 1).

Unfortunately, while particle size distribution can be computed mechanically, identification of particles is labor intensive. Core sections must be inspected microscopically, and spicules must be distinguished from the organic and inorganic constituents of the cores, plus refractory detritus of other silica-using organisms such as diatoms and radiolarians (Phylum Retaria). Comprehensive analysis of large numbers of sections is not practical. FlowCam® technology (Fluid Imaging Technologies, Inc.) offers a partial solution to this problem of scale. In a FlowCam device, particles in suspension pass one by one in front of a camera; this approach can generate tens of thousands of photographs from a sample in a matter of minutes. To our knowledge this technology has not yet been systematically applied to core samples, although recent success with cyanobacterial blooms [23] suggests that the approach is promising. However, a FlowCam approach to large-scale spicule analysis requires software classification of large numbers of individual photographs. To facilitate development of a classifier, Fluid Imaging Technologies generously performed a small-scale run on a core sample from the Bering Sea from the Pliocene warm period. 110 photographs of sponge spicules and 113 photographs of non-spicule particles (diatoms and radiolarians) were generated (Fig. 1).

Our experience with computer vision technology suggested that an accurate photograph classifier could be achieved using Artificial Neural Networks (ANNs), which have been shown to be effective in handling large scale image classification problems and are the de-facto standard in image recognition problems [24, 25]. In particular we hypothesized that a deep convolutional neural network (ConvNet) [26] could achieve

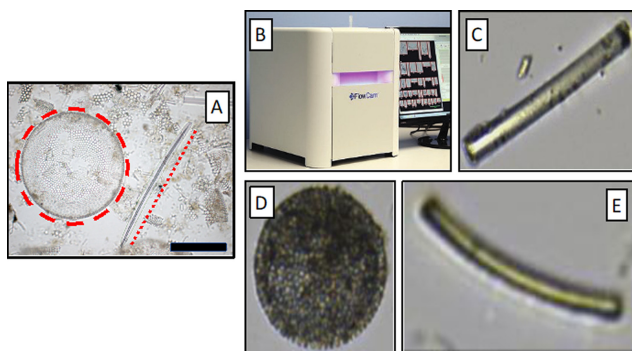


Fig. 1. (A) Smear slide microphotograph showing a diatom (in red dashed circle) and a glass sponge spicule (along dashed red line). (B) FlowCam series 8000 flow imaging microscope. Reprinted with permission from Fluid Imaging Technologies, Inc. (C) Glass sponge spicule. (D) Diatom. (E) Radiolarian. The similarity between (C) and (E) necessitates a sophisticated software classifier. (Color figure online)

acceptable accuracy despite the obvious similarity between glass sponge spicules and radiolarians. However, ConvNets require extensive positive training data, and the available training data consisted of just 110 positive examples and 113 negative examples from the few FlowCam images provided by Fluid Imaging Technologies. Fortunately, training data can be enhanced *in silico*, and we further hypothesized that ConvNet accuracy could benefit from data augmentation [27] via image manipulation of the original photographs. We have trained a deep ConvNet, called “Poriferal Vision”, that achieves 95% accuracy with available data. A Support Vector Machine (SVM) classifier [28, 29] was trained using the same data to provide a basis for comparison; the Poriferal Vision model consistently achieved higher accuracy.

2 Methods

To enable rapid *in silico* identification of FlowCam images from marine core samples, a deep convolutional neural network (ConvNet) called “Poriferal Vision” was trained on data from the Bering Sea. Data augmentation was applied to training instances. 20% of the augmented data was withheld from training and used for testing. Withheld data was tested on the Poriferal Vision ConvNet and, for comparison, on a Support Vector Machine.

2.1 Data Collection

Sediments from a section of a Bering Sea core sample (Fig. 3) were provided by the Aiello research group at Moss Landing Marine Laboratories. The Integrated Ocean Drilling Program Expedition 323 to the Bering Sea (IODP Exp 323) discovered that, in the central part of this marginal sea at Bowers Ridge (Site U1340, ~600 m; water depth 1295 m; Fig. 3), sponge spicules comprise a very important component of hemipelagic sedimentation, together with diatom frustules and ice-rafted debris (Fig. 2) [7, 30].

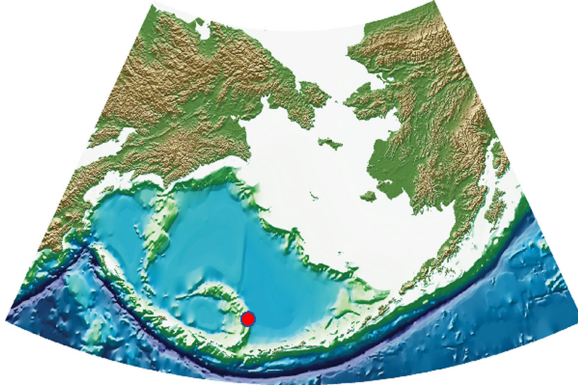


Fig. 2. Bering Sea and adjacent landmasses. The red dot marks sampling site U1340. (Color figure online)

To create positive training and testing sets, 110 sponge spicule photographs were manually selected. Negative example photographs of 80 diatoms and 33 radiolarians were collected. All photographs were randomly split into training and testing sets. To compensate for the small number of training examples, the positive and negative training sets were enhanced by transforming images using the OpenCV software package [31]. Individual images were read into OpenCV in grayscale. In order to preserve the structure of the single objects in the photos and to get more realistic transformations, the polygonal chain approximation method [32] was used to trace highly accurate contours around each object. A rectangular bounding box was drawn around the contours, isolating the object and preserving its local background context. The bounding box was flipped horizontally, vertically, and horizontally + vertically to generate 3 new images from each original. All images were rotated in increments of 15° . Lastly, in order to reduce the probability of misidentification of poorly centered images, 6 perspective transformations were applied to each original image by translating the object by a small distance. To ensure that perspective transformation did not create any blank images, a blurring Gaussian filter [33] was applied to the transformed images; any image with a majority of black pixels or a majority of white pixels was removed from the data set. All image backgrounds were cleaned by removing noise and deleting line artifacts of rotation. Images were saved in .jpg format and, as a quality control check, visually inspected. Accepted images were resized to 138×78 pixels (the mean resolution among accepted images).

2.2 ConvNet and Support Vector Machine Design and Training

The Poriferal Vision ConvNet was implemented in TensorFlow [34]; The architecture of the model was conservatively inspired by AlexNet [26], which seems suitable as a pioneering large scale image classification architecture for our specific problem. The motivation behind designing a lean ConvNet was to accommodate for the limited size of the dataset despite data augmentation and learn a generalizable distribution of features from it. It consists of two 2D convolutional layers with a filter size of 64 and kernel size of (3, 3) to convolve over. Considering the relatively small dimension of the transformed images a smaller kernel size would allow learning localized features and differentiate

between spicules and radiolarians, due to the high degree of similarity between them. Both layers use relu [35] activation function and max pooling to down sample the input representation.

These are followed by a flatten and dense layer with relu activation to reduce the spatial dimension of the input, finally followed by an output layer with sigmoid activation [36] (Fig. 3). Training proceeded for 14 epochs.

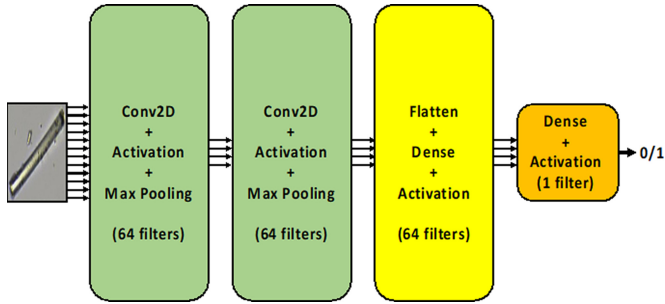


Fig. 3. Poriferal Vision Convolutional Neural Network (CNN) design

To provide a basis for assessing the Poriferal Vision model, a Support Vector Machine classifier was created using the Support Vector Classifier module of SciKit-Learn [37], with a linear kernel. Following the method of R. Sahak et al. [38], training time was reduced by using Principal Component Analysis (PCA) to limit feature dimensionality. The kernel’s c and gamma parameters were optimized using SciKit-Learn’s GridSearchCV module.

2.3 Evaluation

The Poriferal Vision ConvNet was trained first with the original training data and then with the enhanced data. In each category (sponge spicules, diatoms, and radiolarians), 20% of data was withheld from training and later used for testing (Table 1). All training and experiments performed on Poriferal Vision were also performed on the Support Vector Machine. For all 4 experiments (Poriferal Vision and SVM, original and enhanced data), the classifiers were evaluated by computing test accuracy, f-score, precision, and recall.

Table 1. Training and testing sets.

Data Set	Organism	Original photos	Original testing photos	Original training photos	Flipped images	Rotated images	Perspective transformations	Total selected training images
Positive	Sponges	110	30	80	240	5120	376	5816
Negative	Diatoms	90	22	68	204	4352	408	5032
	Radiolarians	33	8	25	75	1600	150	1850

3 Results

Test accuracy, f-score, precision, and recall for the 4 experiments are given in Table 2. Accuracy and loss over 14 training epochs, for both the original and enhanced data sets, are shown in Fig. 4.

Table 2. Results for SVM and Poriferal Vision, for original and enhanced (i.e. original and transformed) images.

Model	Images	Test accuracy	F-score	Precision	Recall
SVM	Original	0.83	0.83	0.83	0.83
	Enhanced	0.80	0.79	0.86	0.80
Poriferal Vision	Original	0.92	0.92	0.93	0.92
	Enhanced	0.95	0.95	0.95	0.95

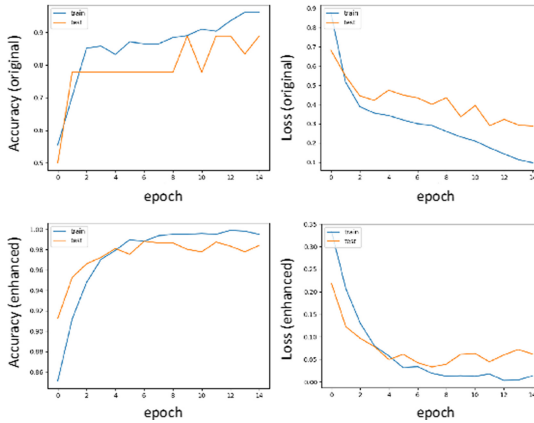


Fig. 4. Poriferal Vision CNN training regimes, showing accuracy (left) and loss (right) with the original (top) and enhanced (bottom) data sets over 14 training epochs. The test set (brown lines) was composed of 20% of each photograph set (spicules, diatoms, radiolarians), selected randomly and withheld from model training. (Color figure online)

4 Discussion

The Poriferal Vision ConvNet, trained with the enhanced data set, achieved 95% accuracy according to all 4 computed statistics (test accuracy, f-score, precision, and recall). Enhancing the positive and negative training data added 2–3% to the statistics. Our hypotheses are therefore confirmed: a ConvNet has been shown to achieve reasonable accuracy, and image enhancement improves accuracy. Significantly, the enhanced-data version of Poriferal Vision is accurate enough to be applied to large-scale sets of FlowCam images when those become available.

The Support Vector Machine classifiers, trained on the same data as the Poriferal Vision models, was less accurate than Poriferal Vision by 5 to 11% points. This may be due to the PCA-based dimension reduction which was applied in order to make model computation time tractable. The time complexity of a linear kernel SVM is $O(n)$, where n is the number of dimensions. Since SVMs are not practical without dimension reduction, they are not appropriate for the kind of classification undertaken here.

The next stage of Poriferal Vision's development will involve taxonomic classification of spicules (and perhaps diatoms and radiolarians). We hope and expect that the insights gained from consequent analysis of the historical record will help the scientific and ocean policy communities to understand the future role of sponges in marine environments.

5 Conclusions

The Poriferal Vision ConvNet, trained with the enhanced data set, achieved at least 95% accuracy according to all 4 computed statistics (test accuracy, f-score, precision, and recall). This result justifies future efforts to expand the training set.

A larger training set, with spicules identified by species or by higher taxonomic level, will enable a more sophisticated classifier that can identify samples by species or by higher taxonomic level. Such a classifier, if deployed on shipboard and receiving FlowCam images in real time, would enable real-time analysis of the local marine environment, thus greatly enhancing the efficiency of sponge analysis at a moment in human history when such analysis is urgent.

References



1. Conway, K.W., Barrie, J.V., Krautter, M.: Geomorphology of unique reefs on the western Canadian shelf: sponge reefs mapped by multibeam bathymetry. *Geo-Mar. Lett.* **25**(4), 205–213 (2005)
2. Yahel, G., Whitney, F., Reiswig, H.M., Eerkes-Medrano, D.I., Leys, S.P.: In situ feeding and metabolism of glass sponges (Hexactinellida, Porifera) studied in a deep temperate fjord with a remotely operated submersible. *Limnol. Oceanogr.* **52**(1), 428–440 (2007)
3. Kahn, A.S., Yahel, G., Chu, J.W.F., Tunnicliffe, V., Leys, S.P.: Benthic grazing and carbon sequestration by deep-water glass sponge reefs. *Limnol. Oceanogr.* **60**(1), 78–88 (2015)
4. West, R.R.: Temporal changes in Carboniferous reef mound communities. *PALAIOS* **3**(2), 152 (1988)
5. Brunton, F.R., Dixon, O.A.: Siliceous sponge-microbe biotic associations and their recurrence through the phanerozoic as reef mound constructors. *PALAIOS* **9**(4), 370 (1994)
6. Kiessling, W., Simpson, C.: On the potential for ocean acidification to be a general cause of ancient reef crises: ancient reef crises. *Glob. Change Biol.* **17**(1), 56–67 (2011)
7. Aiello, I.W., Ravelo, A.C.: Evolution of marine sedimentation in the Bering Sea since the Pliocene. *Geosphere* **8**(6), 1231–1253 (2012)
8. Birkeland, C. (ed.): *Life and Death of Coral Reefs*. Chapman and Hall, New York (1997)
9. Reaka-Kudla, M.L., Wilson, D.E., Wilson, E.O.: *Biodiversity II*. Joseph Henry Press, Washington, D.C. (1997)
10. Glynn, P.W.: Coral reef bleaching: ecological perspectives. *Coral Reefs* **12**(1), 1–17 (1993). <https://doi.org/10.1007/BF00303779>

11. Hughes, T.P.: Climate change, human impacts, and the resilience of coral reefs. *Science* **301**(5635), 929–933 (2003)
12. De'ath, G., Fabricius, K.E., Sweatman, H., Puotinen, M.: The 27-year decline of coral cover on the Great Barrier Reef and its causes. *Proc. Natl. Acad. Sci.* **109**(44), 17995–17999 (2012)
13. Hoegh-Guldberg, O., et al.: Coral reefs under rapid climate change and ocean acidification. *Science* **318**(5857), 1737–1742 (2007)
14. Wisshak, M., Schönberg, C.H.L., Form, A., Freiwald, A.: Ocean acidification accelerates reef bioerosion. *PLoS ONE* **7**(9), e45124 (2012)
15. Albright, R.: Reviewing the effects of ocean acidification on sexual reproduction and early life history stages of reef-building corals. *J. Mar. Biol.* **2011**, 1–14 (2011)
16. Goreau, T., McClanahan, T., Hayes, R., Strong, A.: Conservation of coral reefs after the 1998 global bleaching event. *Conserv. Biol.* **14**(1), 5–15 (2000)
17. Nakićenović, N., Intergovernmental Panel on Climate Change (eds.): *Special Report on Emissions Scenarios: A Special Report of Working Group III of the Intergovernmental Panel on Climate Change*. Cambridge University Press, Cambridge, New York (2000)
18. Bell, J.J., Davy, S.K., Jones, T., Taylor, M.W., Webster, N.S.: Could some coral reefs become sponge reefs as our climate changes? *Glob. Change Biol.* **19**(9), 2613–2624 (2013)
19. Chu, J., Leys, S.: High resolution mapping of community structure in three glass sponge reefs (Porifera, Hexactinellida). *Mar. Ecol. Prog. Ser.* **417**, 97–113 (2010)
20. Beaulieu, S.E.: Life on glass houses: sponge stalk communities in the deep sea. *Mar. Biol.* **138**(4), 803–817 (2001)
21. Marliave, J.B., Conway, K.W., Gibbs, D.M., Lamb, A., Gibbs, C.: Biodiversity and rockfish recruitment in sponge gardens and bioherms of southern British Columbia, Canada. *Mar. Biol.* **156**(11), 2247–2254 (2009)
22. Guillas, K.C., Kahn, A.S., Grant, N., Archer, S.K., Dunham, A., Leys, S.P.: Settlement of juvenile glass sponges and other invertebrate cryptofauna on the Hecate Strait glass sponge reefs. *Invertebr. Biol.* **138**(4), e12266 (2019)
23. Graham, M.D., et al.: High-resolution imaging particle analysis of freshwater cyanobacterial blooms: FlowCam analysis of cyanobacteria. *Limnol. Oceanogr. Methods* **16**(10), 669–679 (2018)
24. Widrow, B., Rumelhart, D.E., Lehr, M.A.: Neural networks: applications in industry, business and science. *Commun. ACM* **37**, 93–106 (1994)
25. Rawat, W., Wang, Z.: Deep convolutional neural networks for image classification: a comprehensive review. *Neural Comput.* **29**(9), 2352–2449 (2017)
26. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. *Commun. ACM* **60**(6), 84–90 (2017)
27. Shorten, C., Khoshgoftaar, T.M.: A survey on image data augmentation for deep learning. *J. Big Data* **6**(1), 1–48 (2019). <https://doi.org/10.1186/s40537-019-0197-0>
28. Cortes, C., Vapnik, V.: Support-vector networks. *Mach. Learn.* **20**(3), 273–297 (1995)
29. Chapelle, O., Haffner, P., Vapnik, V.N.: Support vector machines for histogram-based image classification. *IEEE Trans. Neural Netw.* **10**(5), 1055–1064 (1999)
30. Schlung, S.A., et al.: Millennial-scale climate change and intermediate water circulation in the Bering Sea from 90 ka: a high-resolution record from IODP Site U1340: Bering Sea climate change since 90 KA. *Paleoceanography* **28**(1), 54–67 (2013)
31. Bradski, G.R., Kaehler, A.: *Learning OpenCV: Computer Vision with the OpenCV Library*, 1st edn. O'Reilly, Beijing (2011)
32. Ramer, U.: An iterative procedure for the polygonal approximation of plane curves. *Comput. Graph. Image Process.* **1**(3), 244–256 (1972)
33. Deng, G., Cahill, L.W.: An adaptive Gaussian filter for noise reduction and edge detection. In: 1993 IEEE Conference Record Nuclear Science Symposium and Medical Imaging Conference (1993)

34. Abadi, M., et al.: TensorFlow: a system for large-scale machine learning, p. 21 (2016)
35. Hahnloser, R.H.R., Sarpeshkar, R., Mahowald, M.A., Douglas, R.J., Seung, H.S.: Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature* **405**, 947–951 (2000)
36. Kilian, J., Siegelmann, H.T.: The dynamic universality of sigmoidal neural networks. *Inf. Comput.* **128**(1), 48–56 (1996)
37. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
38. Sahak, R., Mansor, W., Lee, Y.K., Yassin, A.I.M., Zabidi, A.: Performance of combined support vector machine and principal component analysis in recognizing infant cry with asphyxia. In: 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology (2010)



Experimental Evaluation of GAN-Based One-Class Anomaly Detection on Office Monitoring

Ning Dong¹, Yusuke Hatae¹, Muhammad Fikko Fadjrimiratno²,
Tetsu Matsukawa¹ , and Einoshin Suzuki¹ 

¹ ISEE, Kyushu University, Fukuoka 819-0395, Japan
dongning.ac@gmail.com, hatae.yusuke.042@s.kyushu-u.ac.jp,
{matsukawa,suzuki}@inf.kyushu-u.ac.jp

² SLS, Kyushu University, Fukuoka 819-0395, Japan
muhammadfikko@gmail.com

Abstract. In this paper, we test two anomaly detection methods based on Generative Adversarial Networks (GAN) on office monitoring including humans. GAN-based methods, especially those equipped with encoders and decoders, have shown impressive results in detecting new anomalies from images. We have been working on human monitoring in office environments with autonomous mobile robots and are motivated to incorporate the impressive, recent progress of GAN-based methods. Lawson et al.'s work tackled a similar problem of anomalous detection in an indoor, patrol trajectory environment with their patrolbot with a GAN-based method, though crucial differences such as the absence of humans exist for our purpose. We test a variant of their method, which we call FA-GAN here, as well as the cutting-edge method of GANomaly on our own robotic dataset. Motivated to employ such a method for a turnable Video Camera Recorder (VCR) placed at a fixed point, we also test the two methods for another dataset. Our experimental evaluation and subsequent analyses revealed interesting tendencies of the two methods including the effect of a missing normal image for GANomaly and their dependencies on the anomaly threshold.

Keywords: One-class anomaly detection · Generative Adversarial Networks · Human monitoring

1 Introduction

Monitoring an office environment, especially the humans inside, represents an interesting problem for intelligent systems from both scientific and industrial viewpoints. Detecting anomalies is one of the most fundamental and yet important subproblems, though collecting and even knowing such anomalies beforehand are at the same time laborious and difficult. One-class anomaly detection,

A part of this work is supported by Grant-in-Aid for Scientific Research JP18H03290 from the Japan Society for the Promotion of Science (JSPS).

which takes only normal data in the training stage to detect anomalies in the test stage, solves these shortcomings. Recently Generative Adversarial Networks (GAN) [1], which are deep neural networks capable of learning the probabilistic distribution of the given, originally unlabeled, examples, have shown impressive results in detecting new anomalies from images. For instance, Lawson et al. report that the false positive rate of 4.72% achieved with their previous method [2] dropped to 0.42% with their GAN-based method in an anomaly detection problem by their patrolbot [3].

We have been working on office monitoring including humans inside with autonomous mobile robots, e.g., skeleton clustering [4], facial expression clustering [5], and fatigue detection [6]. Recently our interests are focused on one-class anomaly detection [7, 8], though these works adopt non-GAN-based approaches. Motivated to incorporate the impressive, recent progress of GAN-based methods, we test two most relevant methods, which we explain in Sect. 2, on our robotic and VCR datasets.

2 Related Work

Recently GAN-based one-class anomaly detection has attracted considerable attention of the machine learning community. Schlegl et al. [9] proposed AnoGAN to detect anomalies on Optical Coherence Tomography (OCT) data. They assumed that the trained latent space represents the true distribution of the training data. However, their method is time-consuming in finding a latent vector that corresponds to an image that is visually most similar to a given query image [10] in the test stage. To cope with the shortcoming that the parameters need to be updated in the test stage of AnoGAN, Zenati et al. proposed an anomaly detection method [11] which is efficient at test time by leveraging BiGAN [12], which simultaneously learns an encoder with a decoder and a discriminator during training. It can avoid the computationally expensive process during the test stage. Sabokrou et al. proposed a framework for one-class novelty detection which consists of a reconstructor and a discriminator [13]. They added noise to the original normal examples to train the reconstructor network to make it more robust and employed the discriminator as a detector to classify whether the input is abnormal. A deep generative model trained on a single class cannot generate examples belonging to other classes. Perera et al. focused on this problem and proposed OCGAN for novelty detection [14]. They restricted the boundary of the latent space and used a latent discriminator and a visual discriminator to ensure the images generated from any latent vector belong to the same class. Different from the traditional GAN-based encoder-decoder approaches, Akcay et al. [10] employed an encoder-decoder-encoder structure to capture the two latent vectors which show significant differences with an abnormal example. The added encoder aids learning the data distribution for the normal examples. We adopt their GANomaly [10] for evaluation in our experiments as we consider it a relevant cutting-edge method.

We have witnessed a number of developments and applications of autonomous mobile robots in anomaly detection. Chakravarty et al. [15] used modified sparse

and dense stereo algorithms to detect anomalies which were never shown during the training stage for a patrolbot. However, light intensity has a great impact on their detection accuracy. Lawson et al. [2] proposed a method with clustering normal features from CNNs in a fixed path with a patrolbot. Abnormal features would produce large distances to these clusters. Later, they extended their work in [3] to find anomalies with an autoencoder-decoder GAN, which achieved much better performance as we stated in the previous section. We extend their method by replacing their autoencoder with a more sophisticated encoder [16], and call the extended method FA-GAN in this paper. We also use FA-GAN in our experiments.

The robotic data and the VCR data we use in our experiments have been introduced in our previous work on detecting anomalous image regions with deep captioning [8]. In the work, our anomaly detector represents each salient region with its image, caption, and position features and uses an incremental clustering method [17] to detect anomalies with these features. The point is to exploit another dataset used in training a combination of Convolutional Neural Network (CNN) [18] and Long Short-Term Memory (LSTM) [19,20] through deep captioning [21]. We will explain the details of our datasets in Sect. 4.1. Since the method [8] uses deep captioning and conducts evaluation on image region level, we leave its comparison with GANomaly and FA-GAN for our future work.

3 Tested Methods

3.1 FA-GAN

Lawson et al. use the DCGAN approach [22], adopting an architecture which is similar to what was proposed in Context Encoders [16]. Unlike DCGAN, they use an autoencoder-style with a bottleneck size of 4096. Considering the more complex nature of our office monitoring problem, we replace their autoencoder with a more sophisticated encoder in [16], as we explained in the previous Section. The resulting FA-GAN has an encoder-decoder architecture, which is shown in Fig. 1(a). The encoder G_E is composed of convolution layers and batch-normalization layers with LeakyReLU activation function. The decoder G_D adopts the structure of DCGAN [22] with deconvolutional layers to generate images from a latent vector. The discriminator D has a similar structure to G_E and uses Sigmoid function to output whether the input is real or generated.

Given a training set X , which consists of N normal images, $X = \{x_1, \dots, x_N\}$, the generator G first reads an input image x as the input to its encoder G_E to downscale x by compressing it to a latent vector $z = G_E(x)$. Then the decoder G_D tries to reconstruct z to an image \hat{x} . To enhance the capability of reconstructing an image, FA-GAN uses the adversarial loss [1] shown in Eq. 1.

$$Loss_{FA} = \mathbb{E}_{x \sim p_x} [\log D(x)] + \mathbb{E}_{x \sim p_x} [\log(1 - D(\hat{x}))] \quad (1)$$

In the test stage, the generator G produces the latent vectors z and \hat{z} from the original input image and its corresponding generated image through the encoders

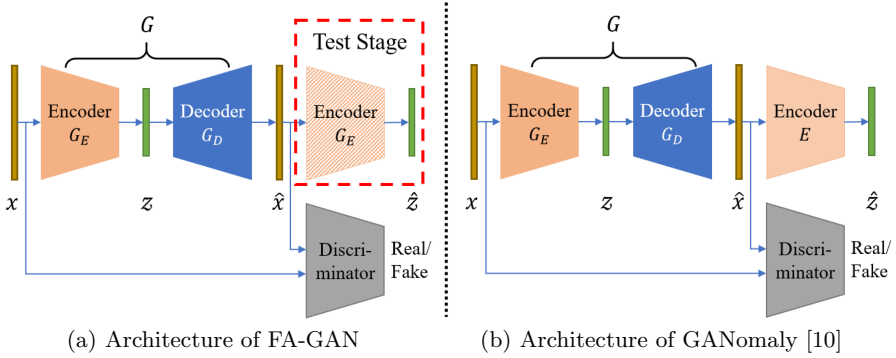


Fig. 1. Architectures of FA-GAN (the left one) and GANomaly (the right one)

in the network, respectively. Note that the encoder in the dashed rectangle is not used during the training stage. It is just a copy of the former one after training. Finally, a test example can be determined as normal or abnormal by comparing its abnormal degree A with a user-given threshold ϕ , where $A = \|z - \hat{z}\|_2$ is computed as the distance between the two latent vectors z and \hat{z} . If $A > \phi$ then the image is predicted as anomalous, otherwise normal. Since the network is only trained with normal data, the generator cannot reconstruct an anomalous image well, which means there will be a large difference between the two latent vectors.

3.2 GANomaly

GANomaly [10] has a similar architecture to FA-GAN. It also uses an encoder-decoder generator in which the latent vector z of the original input is obtained by $G_E(x) = z$. The difference is that there is one more encoder E to produce \hat{z} after the generated image as shown in Fig. 1 (b). The parameters of the additional encoder E are also optimized during the training stage, in which the distance between z and \hat{z} is considered as a loss. After the second encoder E , the generated image is downscaled to a latent vector $\hat{z} = E(\hat{x})$, which has the same size with z .

The loss function $Loss_GANomaly$ of GANomaly consists of an adversarial loss L_{adv} , a contextual loss L_{con} , and an encoder loss L_{enc} as in Eq. 2, where w_{adv} , w_{con} , and w_{enc} represent hyper-parameters. In Eq. 3, $f(x)$ represents the value of the activation function on the last CNN layer in the discriminator given image x [10]. To predict whether a test example is abnormal, GANomaly uses the same flow as FA-GAN, which is to compute the distance between z and \hat{z} , except \hat{z} is produced by the additional encoder E , and not G_E .

$$Loss_GANomaly = w_{adv}L_{adv} + w_{con}L_{con} + w_{enc}L_{enc} \quad (2)$$

$$L_{adv} = \mathbb{E}_{x \sim p_x} \|f(x) - \mathbb{E}_{x \sim p_x} f(G(x))\|_2 \quad (3)$$

$$L_{con} = \mathbb{E}_{x \sim p_x} \|x - G(x)\|_1 \quad (4)$$

$$L_{enc} = \mathbb{E}_{x \sim p_x} \|z - \hat{z}\|_2 \quad (5)$$



Fig. 2. Examples in the robotic dataset (the left two) and the VCR dataset (the right two). From the left, the classes are normal, abnormal, normal, and abnormal.

Table 1. Distributions of the two datasets

	Robotic dataset		VCR dataset	
	Training	Test	Training	Test
Normal	4768	343	16800	684
Abnormal	0	15	0	31

4 Experimental Evaluation

4.1 Experimental Setup

To evaluate the two GAN-based methods on anomaly detection, we conduct experiments with our robotic and VCR datasets, which we introduced in [8]. Figure 2 shows several examples. The robotic dataset is taken in a room by our TurtleBot2 with Kobuki, which is equipped with a Kinect v2. It contains frequent scene changes as the robot moved in an office. The VCR dataset is taken with a VCR placed on a spandrel wall. It has only a few scene changes as the VCR was put on a fixed point and a human occasionally changed its angle. Table 1 shows the distributions of the datasets.

We install GANomaly [10] and implement FA-GAN [3] in PyTorch (v1.3.1 with Python 3.6.9). The networks are optimized by Adam [23] with an initial learning rate of 0.0001, $\beta_1 = 0.5$, and $\beta_2 = 0.999$. We set the batch size to 32 and each network is trained for 70 epochs. The hyper-parameters are set as $w_{adv} = 1$, $w_{con} = 50$, and $w_{enc} = 1$. The size of a latent vector is set to 4096 throughout the experiments. We normalize all the anomaly scores obtained in the test set to the range of [0, 1].

4.2 Results

In this section, we first analyze the dependency of the performance in F1 score on ϕ and then investigate the reasons behind the mistakes by the two methods. The left two plots in Fig. 3 show the results of the dependency. We see that FA-GAN outperforms GANomaly in the robotic dataset but loses to it in the VCR dataset. From Fig. 4 we can see the reasons. In (a), there is a clear boundary to distinguish the normal and abnormal examples with FA-GAN on the robotic dataset. GANomaly succeeds in concentrating the anomaly scores of all normal examples in a small interval on the VCR dataset in (d). The right two plots in Fig. 3 show the results in ROC curve and AUC.

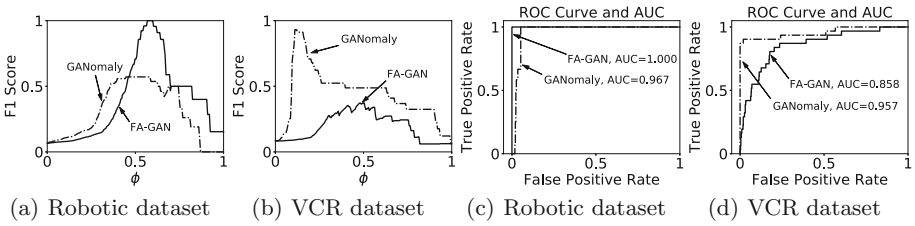


Fig. 3. F1 scores in terms of threshold ϕ (left two plots) and the ROC curve and AUC (right two plots)

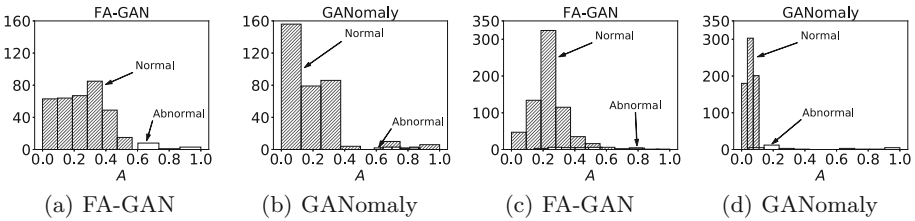


Fig. 4. Histogram of the anomaly scores for the test data on the robotic dataset (the left two plots) and the VCR dataset (the right two plots)

We assume that setting ϕ to its best value is possible as long as the office environment does not change drastically. Based on this assumption we conduct our investigation on the best cases in terms of the value of ϕ . We focus on mistakes committed by the two methods, which are summarized in Table 2. In the Table, FN and FP represent the number of false negatives and the number of false positives, respectively. “Same examples” is the number of images wrongly detected by both methods.

On the robotic dataset, we see from Table 2 that FA-GAN made no mistake while GANomaly 18 false positives. Since the 18 examples look all similar, we

Table 2. Statistics of the wrongly detected examples

	Robotic dataset		VCR dataset	
	FN	FP	FN	FP
FA-GAN	0	0	19	21
GANomaly	0	18	4	0
Same examples	0	0	2	0

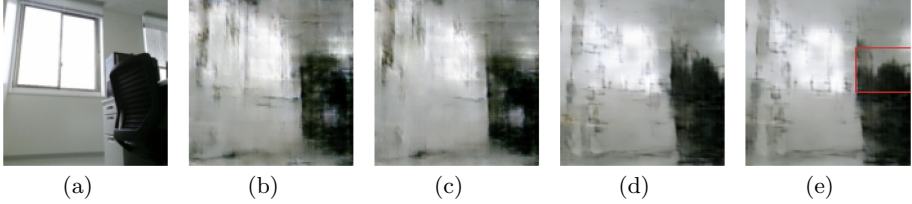


Fig. 5. One of the FP examples with GANomaly in the robotic dataset. (a) Original input. (b) Generated image with z by FA-GAN. (c) Generated image with \hat{z} by FA-GAN. (d) Generated image with z by GANomaly. (e) Generated image with \hat{z} by GANomaly.

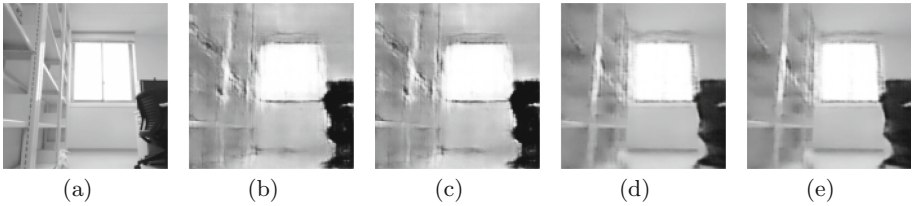


Fig. 6. One of the correctly detected examples by the two methods. See the captions of Fig. 5 for (a)–(e).

pick one of them and show it in Fig. 5(a). The anomaly scores A are 0.968 in GANomaly and 0.506 in FA-GAN. We also show the generated images by z and \hat{z} with both methods in Fig. 5(b)–(e) and see that the red rectangle region in Fig. 5(e) accounts for the large A in GANomaly.

We also pick an example which is similar to Fig. 5(a) but correctly classified by both methods and show it in Fig. 6(a). The anomaly scores are 0.266 and 0.106 for FA-GAN and GANomaly, respectively. We see from Fig. 6(b), (c) and (d), (e) that z and \hat{z} are similar in both methods.

Further inspection revealed that no similar image to Fig. 5(a) exists in the training set while all other images in the test set have similar images in the training set. These analyses show the higher generalization capability of FA-GAN to GANomaly for this dataset. To justify our claim, we added random noise to the 18 FP examples to generate 72 additional examples and added

them in the training set. We trained GANomaly on this dataset and obtained a perfect result, i.e., no mistake committed and hence $AUC = 1.0$.

On the VCR dataset, we see from Table 2 that there are only 4 FN examples with GANomaly but 19 FN examples and 21 FP examples with FA-GAN. This dataset was recorded in one corner by the VCR, so the intuitive complexity of the dataset is relatively reduced compared with the first dataset. Moreover, the training set, which consists of 16800 examples, is large, so the second encoder of GANomaly can well learn the distributions of the normal features in this dataset. Figure 7(a) shows an FN example with FA-GAN, which is correctly detected by GANomaly. Hiding under a table¹ is considered as an anomaly because nobody does it in the training set. The subsequent images in (b)–(e) are generated images with z and \hat{z} by the two methods. Unlike in Fig. 5, the anomaly scores A in both methods are small, which is justified by the small differences between (b) and (c) as well as (d) and (e). The different results can be explained by the best values of ϕ . Figure 4 shows the histograms on the anomaly scores A in the both methods. It can be seen from Fig. 4(b) that GANomaly concentrates the anomaly scores of all normal examples between 0 and 0.12. Note that FA-GAN and GANomaly adopted a relatively large and small values for ϕ , which results in a false negative and a true positive, respectively.

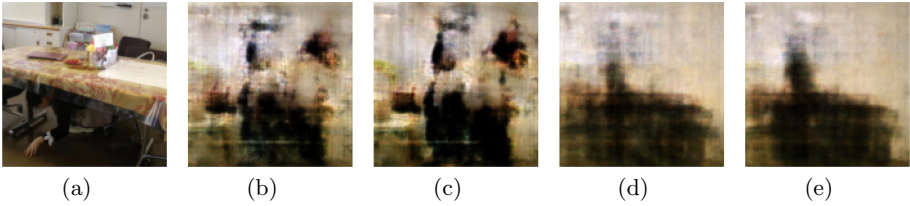


Fig. 7. One of the FN examples with FA-GAN. See the captions of Fig. 5 for (a)–(e).

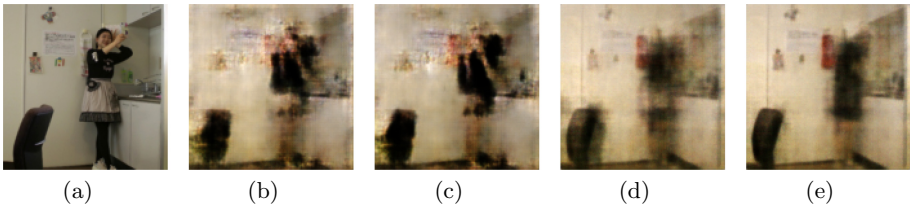


Fig. 8. One of the FN examples with the two methods. See the captions of Fig. 5 for (a)–(e).

¹ Schools in Japan teach students to take this action under strong shakes during an earthquake.

We also show in Fig. 8 one of the FN examples which were wrongly classified by the two methods. Taking a selfie is considered as an anomaly because nobody does it in the training set. Note that the difference between the images generated by z and \hat{z} is small. Figure 9 shows an FP example with FA-GAN. We see that there is a large difference between (b) and (c), especially on the middle part. However, the last two generated images by GANomaly, which achieved an anomaly score of 0.051, are similar.

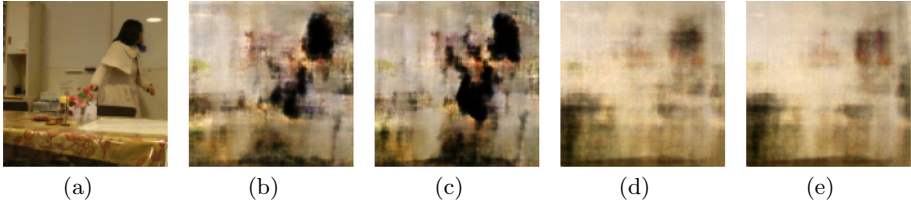


Fig. 9. One of the FP examples with FA-GAN. See the captions of Fig. 5 for (a)–(e).

From the experiments above, we see that in overall the two methods show good performance on the two datasets. We conclude that the two GAN-based methods show their ability to solve the problem of anomaly detection on human monitoring. However, the drawback of the methods is also obvious. For some minor anomalies, e.g., the selfie in Fig. 8(a), the latent vector z after the first encoder does not reflect them, which makes z and \hat{z} be quite similar. It results in a small anomaly score for such examples and thus these abnormal examples will be predicted as normal. We can see these results in Fig. 8.

5 Conclusion

We applied two kinds of GAN-based methods, which are FA-GAN and GANomaly, to the datasets collected by our autonomous robot and with a VCR, so that the anomalies can be detected without any supervision. The results show that FA-GAN performs better on the robotic dataset while GANomaly performs better on the VCR dataset, possibly due to the different frequencies of the scene changes. We analyzed the reason behind the dependency of the performance in F1 score on ϕ through the histograms on the anomaly scores. Also, the methods occasionally make wrong detection with minor anomalies in images due to the scarcity of the scene in the training set or the loss of information in the latent vectors. In general, the two GAN-based methods with encoder-decoder architectures should perform well on our autonomous robot and VCR for human monitoring.

Our future work will take image captions into account to improve the performance of anomaly detection. We think that the captions as weak labels can provide additional useful information on anomaly detection, since we already made some progress with a non-GAN-based approach [8].

References

1. Goodfellow, I.J., et al.: Generative adversarial nets. In: Proceedings of the NIPS, pp. 2672–2680 (2014)
2. Lawson, W., Hiatt, L., Sullivan, K.: Detecting anomalous objects on mobile platforms. In: Proceedings of the CVPR Workshop (2016)
3. Lawson, W., Hiatt, L., Sullivan, K.: Finding anomalies with generative adversarial networks for a Patrolbot. In: Proceedings of the CVPR Workshop (2017)
4. Deguchi, Y., Takayama, D., Takano, S., Scuturici, V.M., Petit, J.M., Suzuki, E.: Skeleton clustering by multi-robot monitoring for fall risk discovery. *J. Intell. Inf. Syst.* **48**(1), 75–115 (2017)
5. Kondo, R., Deguchi, Y., Suzuki, E.: Developing a face monitoring robot for a Deskworker. In: Aarts, E., et al. (eds.) *AmI 2014*. LNCS, vol. 8850, pp. 226–241. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-14112-1_19
6. Deguchi, Y., Suzuki, E.: Hidden fatigue detection for a desk worker using clustering of successive tasks. In: De Ruyter, B., Kameas, A., Chatzimisios, P., Mavrommati, I. (eds.) *AmI 2015*. LNCS, vol. 9425, pp. 268–283. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26005-1_18
7. Fujita, H., Matsukawa, T., Suzuki, E.: Detecting outliers with one-class selective transfer machine. *Knowl. Inf. Syst.* **62**(5), 1781–1818 (2019). <https://doi.org/10.1007/s10115-019-01407-5>
8. Hatae, Y., Yang, Q., Fadjrimiratno, M.F., Li, Y., Matsukawa, T., Suzuki, E.: Detecting anomalous regions from an image based on deep captioning. In: Proceedings of the VISIGRAPP, Subvolume for VISAPP, vol. 5, pp. 326–335 (2020)
9. 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., et al. (eds.) *IPMI 2017*. LNCS, vol. 10265, pp. 146–157. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59050-9_12
10. Akcay, S., Atapour-Abarghouei, A., Breckon, T.P.: GANomaly: semi-supervised anomaly detection via adversarial training. In: Jawahar, C.V., Li, H., Mori, G., Schindler, K. (eds.) *ACCV 2018*. LNCS, vol. 11363, pp. 622–637. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-20893-6_39
11. Zenati, H., Foo, C.S., Lecouat, B., Manek, G., Chandrasekhar, V.R.: Efficient GAN-based anomaly detection. arXiv preprint [arXiv:1802.06222](https://arxiv.org/abs/1802.06222) (2018)
12. Donahue, J., Krähenbühl, P., Darrell, T.: Adversarial feature learning. arXiv preprint [arXiv:1605.09782](https://arxiv.org/abs/1605.09782) (2016)
13. Sabokrou, M., Khalooei, M., Fathy, M., Adeli, E.: Adversarially learned one-class classifier for novelty detection. In: Proceedings of the CVPR, pp. 3379–3388 (2018)
14. Perera, P., Nallapati, R., Xiang, B.: OCGAN: one-class novelty detection using GANs with constrained latent representations. In: Proceedings of the CVPR, pp. 2898–2906 (2019)
15. Chakravarty, P., Zhang, A.M., Jarvis, R., Kleeman, L.: Anomaly detection and tracking for a patrolling robot. In: Proceedings of the Australasian Conference on Robotics and Automation (ACRA) (2007)
16. Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., Efros, A.A.: Context encoders: feature learning by inpainting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2536–2544 (2016)
17. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: a new data clustering algorithm and its applications. *Data Min. Knowl. Discov.* **1**(2), 141–182 (1997)

18. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Proceedings of the NIPS, pp. 1106–1114 (2012)
19. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
20. Zaremba, W., Sutskever, I.: Learning to execute. CoRR abs/1410.4615 (2014)
21. Johnson, J., Karpathy, A., Fei-Fei, L.: DenseCap: fully convolutional localization networks for dense captioning. In: Proceedings of the CVPR, pp. 4565–4574 (2016)
22. 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)
23. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)



Ranking Speech Features for Their Usage in Singing Emotion Classification

Szymon Zaporowski¹  and Bożena Kostek² 

¹ Faculty of Electronics, Telecommunications and Informatics, Multimedia Systems Department, Gdansk University of Technology, 80-233 Gdansk, Poland
smck@multimed.org

² Faculty of Electronics, Telecommunications and Informatics, Audio Acoustics Laboratory, Gdansk University of Technology, 80-233 Gdansk, Poland
bokostek@audioacoustics.org

Abstract. This paper aims to retrieve speech descriptors that may be useful for the classification of emotions in singing. For this purpose, Mel Frequency Cepstral Coefficients (MFCC) and selected Low-Level MPEG 7 descriptors were calculated based on the RAVDESS dataset. The database contains recordings of emotional speech and singing of professional actors presenting six different emotions. Employing the algorithm of Feature Selection based on the Forest of Trees method, descriptors with the best ranking results were determined. Then, the emotions were classified using the Support Vector Machine (SVM). The training was performed several times, and the results were averaged. It was found that descriptors used for emotion detection in speech are not as useful for singing. Also, an approach using Convolutional Neural Network (CNN) employing spectrogram representation of audio signals was tested. Several parameters for singing were determined, which, according to the obtained results, allow for a significant reduction in the dimensionality of feature vectors while increasing the classification efficiency of emotion detection.

Keywords: Mel Frequency Cepstral Coefficients (MFCC) · MPEG 7 low-level audio descriptors · Feature selection · Singing expression classification

1 Introduction

Speech analysis and processing, parametrization as well as automatic classification are the areas being thoroughly researched and developed for the last few decades as their application is of utmost importance in many domains. To name a few [1–4]: telecommunications (VoIP, enhanced IP communication services), automatic speech transcription, automated speech-to-text technologies in video-over IP communications, medical applications such as hearing aids, cochlear implants and speech pathology recognition, language processing for communication services, and more recently human-(intelligent) computer communication based on big data [5, 6]. The last-mentioned application is within the interest of researches as well as commercial usage.

Parametrization is usually the first and often the most crucial block of automatic speech recognition (ASR) in combination with machine learning algorithms. It is only in the last few years that deep learning methodology has forced a different approach to the speech signal processing, in which speech parameters are not retrieved, but the signal in the form of 2D images (i.e., spectrograms, cepstograms, mel-cepstograms, chromagrams, *wavenet*-like, etc.) [7–9] is fed at the net input. On the other hand, automatic evaluation of singing quality in the context of its production (e.g., evaluation of the intonation and timbre of the singing voice) is a relatively poorly studied issue comparing to the ‘pure’ speech area [3, 10]. It should, however, be remembered that singing - like speech - is also a tool for expressing feelings and emotions, thus speech descriptors applied to the singing evaluation should be useful. Also, it is interesting whether deep learning-based methodology may be – in a straightforward way - applied to the singing expression evaluation.

The area of emotion detection in speech is quite well studied, in contrast to the detection of emotion in singing. The article presents issues related to the search for speech signal parameters that may work in the context of automatic evaluation of the quality of expression in singing. For this purpose, a dataset containing recordings of emotional speech and emotionally-singing singing was used, followed by the parametrization of these signals. Some speech descriptors were evaluated for their usage in the feature vector (FV) for singing emotion recognition.

In the next step, the determined parameters were evaluated and reduced using the feature significance algorithm using a Forest of Trees method. Then classification was carried out using the Support Vector Machine (SVM) based on the prepared reduced feature vectors. The final part of the article presents conclusions regarding the development of the proposed methodology to use machine learning methods, including a deep learning approach, to assess the quality of singing expression automatically.

2 Related Work

2.1 Emotion Detection in Speech

The detection of emotions in speech is now very much present in the literature, especially when the possibility of using deep learning for this purpose appeared. Most of the studies describe approaches that use artificial neural networks as classifiers (i.e., convolutional networks, recursive networks, autoencoders), presenting processed spectrograms as input [7, 8, 11]. The use of classical speech signal descriptors (e.g., Mel-Frequency-Cepstral-Coefficients, MFCC) is currently less prevalent in speech research due to the lower accuracy of emotion recognition (approx. 60%) [12, 13]. When 2D image spaces are used as parameters, the classification efficiency can reach over 80% [7, 9]. Such efficiency can also be achieved for some chosen emotions using SVM [8].

2.2 Emotion in Singing

There exist systems that allow automatic assessment of singing and singing quality. The focus of such systems is on assessing the quality of singing individual sounds or a

specific singing technique [14, 15]. Classification accuracy can be up to 80% for these types of systems. Another approach researched is to use the fundamental frequency as a parameter to test whether the person singing a given sound or repeating it after the system prompt is able to sing it correctly [16].

3 Parameter Selection

3.1 Dataset

The RAVDESS database of recordings was used to conduct the experiments presented in this paper. This dataset is often used in research studies, thus it may be treated as a benchmark in the area of speech emotion recognition. The database contains recordings of 24 professional actors (12 women, 12 men) speaking and singing two matched English statements with a neutral North American accent. Speech includes expressions of calm, joy, sadness, anger, fear, surprise and disgust, and singing contains the emotions of calm, joy, sadness, anger, and fear. Each expression is sung and pronounced at two levels of emotional intensity (normal, enhanced). Additionally, an emotionally neutral expression was recorded for each phrase. An example of actors' images presenting a set of emotions available in the database is shown in Fig. 1.

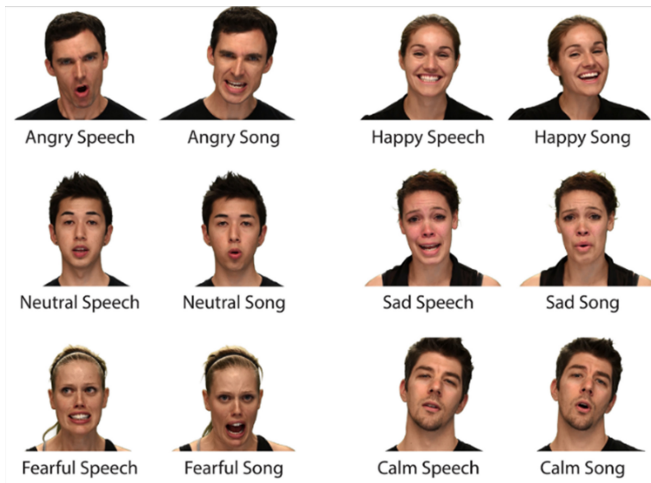


Fig. 1. Example of the RAVDESS emotion expression [17]

All emotion recordings are available in three modalities: audio signal (16-bit resolution, 48 kHz sampling frequency, wave format files), audio-video (720 p resolution, H.264 video coding, AAC audio coding, 48 kHz sampling frequency), mp4 file format) and video signal. The database contains 7356 files (24.8 GB), of which 1440 recordings are of speech alone, and 1012 recordings are of singing. The database is available under a Creative Commons license. Only audio files were used in this study.

3.2 Parameter Selection

Two approaches were utilized to parameterize data from the RAVDESS database. The FV in the first scenario consists of 40 consecutive normalized MFCCs. In the second approach, which uses MPEG 7 descriptors and parameters available in the Librosa library [18], FV contains parameters in both time- and frequency-domains. Time-domain parameters include zero crossings (Zero-Crossing, ZC), and signal energy (Root Mean Square Energy, RMS). The following spectral descriptors were used [18]: the spectral center of gravity (Audio Spectrum Centroid, ASC) and the spectral flatness measure (Audio Spectrum Flatness, ASF). Besides, the spectral roll-off parameter built into the Librosa library was employed [18]. This set of parameters is calculated according to the internal settings of the Librosa library. All the above-mentioned descriptors are present in the literature [19, 20]; thus their definitions will not be recalled here.

For the approach based on Convolutional Neural Networks (CNN), spectrograms were calculated for each of the utterances and songs from the RAVDESS corpora. The audio is sampled at 48000 Hz. Each audio frame is windowed using the Kaiser window of the length of 2048. Fast Fourier Transform (FFT) windows of the length of 2048 are then applied on the windowed audio samples with the STFT hop-length as 512 points. As a result of the aforementioned transformations, the bandwidth of the audio signal was reduced to 8 kHz. In total, there were more than 24200 samples for six classes. That means there were more than 4050 examples for each class.

4 Experiments

4.1 Significance Ranking

To reduce the number of parameters used in the classification and, at the same time, increase the accuracy of the classification by leaving only significant descriptors, the Feature Importance algorithm was employed. The authors have successfully utilized this algorithm in earlier publications related to speech classification [21]. The Feature Importance algorithm is based on another algorithm called Extremely Randomized Trees (ERT) [22]. The concept is derived from Random Forest (RT), which provides a combination of tree predictors so that each tree depends on the value of a random vector sampled independently and has the same distribution for all trees in the forest. The error related to generalization for forests is approaching the limit as the number of trees in the forest increases. ERTs generalization error depends on the correlation between trees in the forest and the strength of individual trees in the entire set [21–23].

The conducted experiments used the implementation of the ERT algorithm contained in the scikit-learn library in Python [25]. The ERT algorithm settings were as follows: `n_estimators = '40'`, `criterion = 'entropy'`, `min_samples_split = 2`, `min_samples_leaf = 1`, `min_weight_fraction_leaf = 0.1`, `max_features = 'auto'`, `min_impurity_decrease = 0.01`, `min_impurity_split = None`, `bootstrap = True`, `random_state = True`, `warm_start = True`, `class_weight = balanced`.

4.2 SVM-Based Classifier

The SVM algorithm, using the scikit-learn package in Python, was employed for the classification. The classifier settings were selected experimentally, ultimately the highest accuracy in the classification for all types of emotions studied was obtained using a degree 3 polynomial kernel with the parameter $C = 0.1$ and ‘balanced’ mode of adjusting weights of individual classes. For comparing classes with each other, one vs. all approach was used.

4.3 CNN Classifier

The CNN classifier used for this experiment was created using the Tensorflow library. The architecture used for this experiment is shown in Fig. 2. The architecture was created in an empirical approach, adding individual layers, and then examining their impact on classification results. Inspiration for this architecture was research presented in the literature [24, 25]. The created neural network was trained for 200 epochs using a batch size of 32 and data split 60/40 for training and validation set, respectively. Titan RTX graphic processor was employed for training.

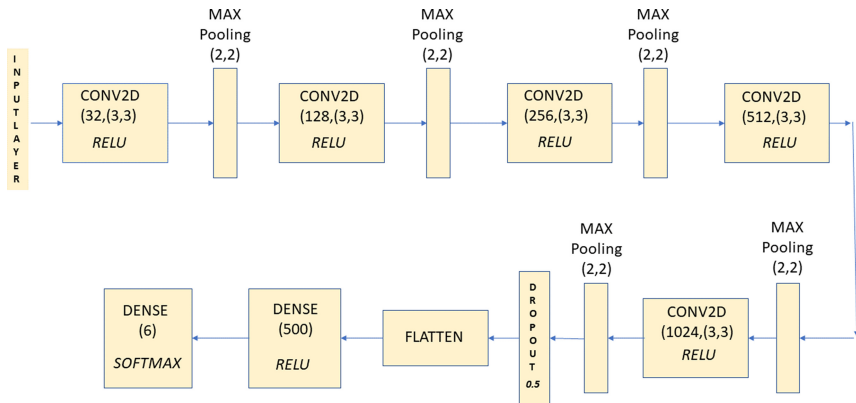


Fig. 2. Architecture of CNN used in the presented experiment

5 Results and Discussions

Below ranking results of the significance of individual parameters and results of emotion classification are shown. Figure 3a) shows the importance of the MFCCs depending on emotions. According to Fig. 3a) coefficient no. 40 is the most versatile. It is the most important feature for several emotions, e.g., joy, calm, sadness, and neutral state. Also, feature no. 1 seems to be most important for anger and fear emotions. Figure 3b) presents the ranking of the parameter importance of speech and singing for all emotions using the MFCC coefficients. As can be seen in Fig. 3b) the most essential parameters for

speech and singing are different. The common one is the coefficient no. 1, but the rest differs. For speech, more important are coefficients with the lower numbers of order, in the case of singing, features with the number higher than 30 were indicated. Tables 1, 2 and 3 show several MFCC parameters, the accuracy of classification as well as the mean square error (MSE) for individual emotions using SVM. The MFCCs shown are derived from the Feature Importance algorithm, which indicated the most important features respectively to values presented in Tables 1, 2 and 3. In most cases, the use of four best coefficients provides the best accuracy results. It is worth noticing that reducing FV to only 10 best features in most cases results in a significant increase in the accuracy score. The emotion of anger is an exception here; accuracy values are oscillating all the time around 70%. Table 4 contains the classification results for the second parameterization scenario employing MPEG-7 low-level descriptors. Table 5 presents results for the CNN-based classification approach. The measure of accuracy is understood as the ratio of the number of correct predictions to the total number of input samples [26].

Based on the presented results, it is possible to distinguish a group of MFCC coefficients that are most important in the process of classifying speech and singing within a given emotion. Classifying emotions in speech and singing using these factors is characterized by high accuracy for most emotions (over 88%). In most cases, the feature vector reduced to two descriptors consisted of MFCCs nos. 29 and 40. For anger, these were coefficients nos. 1 and 39. Among the tested coefficients from the second FV variant, the highest result was obtained using spectral centroid (ASC). The low efficiency of the zero-crossing (ZC) parameter and RMS energy is puzzling. Based on the experiments conducted, it can be observed that MFCC coefficients achieve much better classification results. They seem to be a natural direction in further work on the system for assessing the quality of expression in singing. There was also a decrease in the classification accuracy for anger emotions in all the feature vectors used. This is an interesting phenomenon that should be studied based on another database of recordings. Such a difference may result from a significant change in the volume of speech and possible changes in formant frequencies in the case of this emotion. Articulation associated with emotion can also affect the accuracy of classification. The accuracy of the classification of other emotions is similar. Results for the CNN approach are slightly worse than the results for MFCC parameterization. It could be due to the fact that spectrograms bandwidth were limited to only 8 kHz. It is worth noticing that the categorical cross-entropy values indicate that there is room for improvement, however, presented values and architecture were the best from all tested architectures.

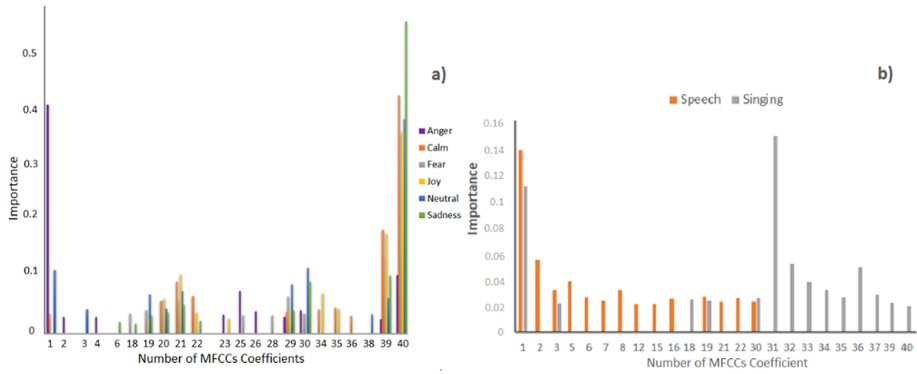


Fig. 3. Classification results for a) The importance of MFCCs in terms of the researched emotion b) for speech and singing for the normalized ranking of MFCCs

Table 1. Classification (speech and singing) results for anger

Quantity of the MFCCs retained	Accuracy [%]	MSE
40	67.7	0.323
20	66.81	0.3319
15	68.58	0.3141
10	68.59	0.3142
5	69.47	0.3053
4	70.35	0.2964
2	68.58	0.3142

Table 2. Classification (speech and singing) results for fear

Quantity of the MFCCs retained	Accuracy [%]	MSE
40	50.66	0.493
20	50.66	0.4933
15	51.33	0.4867
10	82	0.18
5	90	0.1
4	90.67	0.093
2	69	0.31

Table 3. Classification (speech and singing) results for neutral emotion

Quantity of the MFCCs retained	Accuracy [%]	MSE
40	52.7	0.473
20	53.38	0.4662
15	54.05	0.4595
10	70.95	0.29
5	89.19	0.108
4	91.21	0.0878
2	97.973	0.02

Table 4. Emotion classification results for speech and singing based on MPEG 7 descriptors

Emotion [%]	ASC	ASF	Roll-off	ZC	RMS
Neutral	98.52	48.26	34.93	34.53	68.23
Joy	97.87	52.13	38.66	31.81	67.24
Sadness	95.69	47.42	37.84	30.15	62.51
Anger	70.47	27.53	30.92	18.32	47.83
Surprised	93.36	53.77	33.36	24.36	53.27
Fear	96.55	43.29	32.67	21.84	56.68
All	79.39	41.23	35.73	27.27	62.26
Average	90.26	44.80	34.87	26.90	59.72

Table 5. Emotion classification results for speech and singing based on the CNN approach

Emotion	Accuracy [%]	Categorical cross-entropy
Neutral	75.85	0.8693
Joy	51.33	7.8441
Sadness	57.83	2.9245
Anger	76.33	0.9483
Surprised	77.33	1.9979
Fear	77.00	1.3644
All	65.96	1.4873

6 Conclusions

In this paper, an approach to rank speech features based on RAVDESS emotional speech and singing dataset with different approaches to parameterization and classification is presented. Significance of particular MFCC parameters for speech and singing derived from the Feature Importance algorithm is shown. Three different approaches to parameterization using MFCCs, MPEG-7 low-level descriptors, and spectrograms are also demonstrated. The results for each approach are presented and discussed.

In the future, the authors intend to focus on creating parameterization based on all MPEG-7 low-level descriptors and checking their effectiveness in the classification of emotions, both in speech and singing. The next step will also be testing parameterization on sets containing opera singing. The basis of such a system could be the detection of emotions in singing, expanded by a ranking system, using the approach described in the article. However, it seems natural to extend research towards the use of deep learning and 2D representation of signals such as cochleagrams or CQT (Constant-Q) transform.

References

1. Mukesh, K., Shimi, S.: Voice recognition based home automation system for paralyzed people. *Int. J. Adv. Res. Electron. Commun. Eng.* **4**(10), 2508–2515 (2015)
2. Markoff, J.: From Your Mouth to Your Screen, Transcribing Takes the Next Step (2019). <https://www.nytimes.com/2019/10/02/technology/automatic-speech-transcription-ai.html>. Accessed 15 Jan 2020
3. Munir, A., Kashif Ehsan, S., Mohsin Raza, S.M., Mudassir, M.: Face and speech recognition based smart home. In: 2019 International Conference on Engineering and Emerging Technologies ICEET 2019, pp. 1–5 (2019)
4. Delić, V., et al.: Speech technology progress based on new machine learning paradigm. *Comput. Intell. Neurosci.* **2019** (2019). <https://doi.org/10.1155/2019/4368036>
5. Lei, X., Tu, G.-H., Liu, A.X., Ali, K., Li, C.-Y., Xie, T.: The insecurity of home digital voice assistants – Amazon Alexa as a case study (2017)
6. Kannan, K., Selvakumar, J.: Arduino based voice controlled robot. *Int. Res. J. Eng. Technol.* **02**(01), 49–55 (2015)
7. Bertero, D., Fung, P.: A first look into a convolutional neural network for speech emotion detection. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5115–5119 (2017)
8. Kerkeni, L., Serrestou, Y., Raoof, K., Cléder, C., Mahjoub, M., Mbarki, M.: Automatic speech emotion recognition using machine learning (2019). <https://www.intechopen.com/online-first/automatic>
9. Zhao, J., Mao, X., Chen, L.: Speech emotion recognition using deep 1D & 2D CNN LSTM networks. *Biomed. Signal Process. Control* **47**, 312–323 (2019)
10. Scherer, K.R., Sundberg, J., Tamarit, L., Salomão, G.L.: Comparing the acoustic expression of emotion in the speaking and the singing voice. *Comput. Speech Lang.* **29**(1), 218–235 (2015)
11. Cibau, N., Alborno, E., Rufiner, H.: Speech emotion recognition using a deep autoencoder. In: *Anales de la XV Reunion de Procesamiento de la Informacion y Control*, pp. 934–939 (2013)
12. Sezgin, M.C., Günsel, B., Kurt, G.K.: Perceptual audio features for emotion detection. *EURASIP J. Audio Speech Music Process.* **2012**(1), 16 (2012). <https://doi.org/10.1186/1687-4722-2012-16>

13. Poorna, S.S., Jeevitha, C.Y., Nair, S.J., Santhosh, S., Nair, G.J.: Emotion recognition using multi-parameter speech feature classification. In: 2015 International Conference on Computers, Communications, and Systems (ICCCS), pp. 217–222 (2015)
14. Zwan, P.: Expert system for automatic classification and quality assessment of singing voices. In: Audio Engineering Society - 121st Convention Paper 2006, vol. 1, pp. 446–454, January 2006
15. Amir, N., Michaeli, O., Amir, O.: Acoustic and perceptual assessment of vibrato quality of singing students. *Biomed. Signal Process. Control* **1**, 144–150 (2006)
16. Półrończak, E., Łazoryszczak, M.: Quality assessment of intonation of choir singers using F0 and trend lines for singing sequence. *Metod. Inform. Stosow.* **4**, 259–268 (2011)
17. Livingstone, S.R., Russo, F.A.: The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): a dynamic, multimodal set of facial and vocal expressions in North American English. *PLoS ONE* **13**(5), e0196391 (2018)
18. Ellis, D., et al.: librosa: audio and music signal analysis in Python. In: Proceedings of the 14th Python in Science Conference, no. Scipy, pp. 18–24 (2018)
19. Muhammad, G., Melhem, M.: Pathological voice detection and binary classification using MPEG-7 audio features. *Biomed. Signal Process. Control* **11**(1), 1–9 (2014)
20. Dave, N.: Feature extraction methods LPC, PLP and MFCC in speech recognition. *Int. J. Adv. Res. Eng. Technol.* **1**(VI), 1–5 (2013)
21. Zaporowski, S., Czyżewski, A.: Selection of features for multimodal vocalic segments classification. In: Choroś, K., Kopel, M., Kukla, E., Siemiński, A. (eds.) *MISSI 2018. AISC*, vol. 833, pp. 490–500. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-98678-4_49
22. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. *Mach. Learn.* **63**(1), 3–42 (2006)
23. Louppe, G., Wehenkel, L., Suter, A., Geurts, P.: Understanding variable importances in forests of randomized trees. In: *Advances in Neural Information Processing Systems 26*, pp. 431–439 (2013)
24. Svetnik, V., Liaw, A., Tong, C., Christopher Culberson, J., Sheridan, R.P., Feuston, B.P.: Random forest: a classification and regression tool for compound classification and QSAR modeling. *J. Chem. Inf. Comput. Sci.* **43**(6), 1947–1958 (2003)
25. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2012)
26. Goodfellow, I., Bengio, Y., Courville, A.: *Deep Learning*. The MIT Press, Cambridge (2016)



Leveraging Machine Learning in IoT to Predict the Trustworthiness of Mobile Crowd Sensing Data

Corrado Loglisci¹(✉), Marco Zappatore², Antonella Longo^{2,3},
Mario A. Bochicchio³, and Donato Malerba¹

¹ Department of Computer Science, University of Bari, Bari, Italy
{[corrado.loglisci](mailto:corrado.loglisci@uniba.it),[donato.malerba](mailto:donato.malerba@uniba.it)}@uniba.it

² Hesplora srl, Lecce, Italy
marco.zappatore@hesplora.it

³ Department of Engineering Innovation, University of Salento, Lecce, Italy
{[antonella.longo](mailto:antonella.longo@unisalento.it),[mario.bochicchio](mailto:mario.bochicchio@unisalento.it)}@unisalento.it

Abstract. The advances in Internet-of-things (IoT) have fostered the development of new technologies to sense and monitor the urban scenarios. Specifically, Mobile Crowd Sensing (MCS) represents one of the suitable solutions because it easily enables the integration of smartphones collecting massive ubiquitous data at relatively low cost. However, MCS can be affected by wrong data-collection procedures by non-expert practitioners, which can be make useless (or even counter-productive), if contributed data are not trustworthy. Contextualizing monitored data with those coming from phone-embedded sensors and from time/space proximity can improve data trustworthiness. This work focuses on the development of a machine learning method that exploits context awareness to improve the reliability of MCS collected data. It has been validated on a case study concerning urban noise pollution data and promises to improve the trustworthiness of data generated by end users.

1 Introduction

In the recent years, the Mobile Crowd Sensing paradigm (MCS) [1] has become a widely-exploited solution to observe large-scale events and phenomena thanks to the worldwide diffusion of mobiles. MCS applications constantly grow and diversify but they target mainly environmental monitoring in smart cities where non-professionals can be involved to: 1) improve citizenship's environmental awareness and 2) foster the collaboration with scientists to overcome typical limitations to traditional data collection (expensiveness, equipment scarcity).

MCS initiatives very often refer to environmental monitoring as several benefits from wide citizenship participation can be targeted and, at the same time, numerous advantages can be offered to involved users. Thanks to smartphone sensors (embedded or pluggable), the MCS paradigm allows collecting time-/spatial-referenced quantitative observations about multiple parameters (e.g.,

noise, atmospheric pressure, light intensity, etc.). However, some challenges do exist, especially in terms of data quality. Mobile sensors can exhibit limited sensing capabilities, different accuracy and precision, improper usage patterns by their owner, time and space sparsity. These elements determine heterogeneous levels of adequacy to a specific sensing context and produces biased readings that generate outliers in measurement campaigns. Systematic sensing errors can be coped with effectively in advance, by properly training users and calibrating mobile devices by means of professional metering equipment, but data prediction approaches allow to not rely on user and sensor behaviour solely. More specifically, a machine learning approach can rely on pre-categorized mobile crowd-sensed data in terms of their trustworthiness.

This paper proposes an approach that leverages on the contemporary presence of reliable sensor data (about which everything or quite everything is known, *labelled data*) and of unknown (or partially unknown) sensor readings whose trustworthiness is not known a-priori, (*unlabelled data*). Spatial and temporal auto-correlation of these two data typologies can help to assess the trustworthiness of unlabelled data that are spatially and temporally closed to labelled data. To that purpose, a transductive learning approach has been devised to train a classifier able of inferring categories of trustworthiness for the unlabelled data starting from the labelled ones.

The method has been validated for mobile crowd-sensed noise levels collected via smartphone-embedded microphones in the framework of the APOLLON Project¹: a research effort granted by Apulia Region (Italy), focused on urban environmental analyses in terms of noise, air pollution and UV-levels, which offers semantic-based and geo-localized near-/real-time monitoring services to citizens and city decision makers.

The paper is so organized. Section 2 summarizes core features as well as data trustworthiness challenges in MCS. Section 3 details the proposed method, whose empirical evidence is described in Sect. 4. Section 5 provides conclusions.

2 Related Works

The MCS paradigm has proven its effectiveness for individuals, communities of interest and city managers in several scenarios, where mobile sensor readings and user-provided data can be profitably gathered to provide context-based innovative services exhibiting both personal (e.g., improved awareness fostering responsible behaviours) and public applicability (increased fine-grained knowledge of city status in local authorities). By exploiting *participatory* (i.e., user-supervised data collection) or *opportunistic* (i.e., once authorized, the device collects data autonomously) usage patterns, MCS can be relied on to address such typical urban issues as traffic monitoring, first-response management and pollution assessment (air, noise, EM fields, etc.) [7, 13], provided that a large number of mobiles with suitable applications is used. In this way, traditional

¹ <http://web.apollon-project.it/>.

monitoring campaign can be addressed solely where they are needed, thus reducing costs. Mobile apps for MCS require specific architectural capabilities (e.g., local sensing, storage and pre-processing; cloud-based back-ends to process and aggregate data, etc.) and must offer specific data features (e.g., situated data creation, time continuity, high-spatial resolution, etc.) in order to achieve a realistic representation of the scenario where data collection is performed [8]. When participatory MCS solutions are adopted, user-training elements should be added to mobile apps [14] in order to improve the awareness on the correct data collection procedure as well as to motivate users and improve their involvement rate.

When opportunistic MCS strategies are adopted, instead, intelligent data layers targeting a local enhancement of sensor reading quality are needed, featuring either calibration via regression analysis [3, 10] or prediction-based contextualization. Wu et al. [11] investigate a forecasting problem of the trustworthiness by exploiting collaborative filtering. Huang et al. [2] focuses on the device quality and propose a Gompertz function-based classifier to calculate device reputation scores as a reflection of the trustworthiness of data.

However, most part of the methods proposed in the literature can suffer from limited applicability by two main reasons. First, they disregard an intrinsic property of the MCS data, that is, auto-correlation [6], which violates the independence assumption (i.i.d.) on which many machine learning approaches rely on. Second, those methods need user effort in the preparing pre-categorized sensing readings, which requires large collections of labelled data obtained through the manual intervention of authoritative annotators, hence the paradigm *supervised learning*. An approach which inspires the current work, has been studied under the paradigm of *transductive learning* [4] and focuses on the exploitation of unlabelled data in combination with few labelled data, which asks for much less human intervention. This makes unfair any comparison between supervised method and those transductive. In this paper, MCS data refer to a three-level representation [12]. Raw data from physical sensors, virtual sensors (i.e., software applications) or logical sensors (e.g., logs) form the low-level context. The high-level context is inferred through meta-data classified into 1) device context (e.g., connectivity); 2) user context (e.g., user profile); 3) physical context (e.g., temperature); 4) temporal context (i.e., the specific time when the situation occurs). The topmost-level context is the estimation of the user state that can be achieved by combining the two underlying levels.

3 The Method

This section is devoted to the description of the method we design to predict the level (label) of trustworthiness of the MCS data. We first provide basic notions and then explain how the method works.

Let \mathcal{D} be the set of sparsely labelled data which comprises the set \mathcal{L} of labelled data (pre-categorized MCS data) and set of \mathcal{U} of the instances with unknown trustworthiness ($\mathcal{D} = \mathcal{L} \cup \mathcal{U}$). The set \mathcal{D} is spanned on a vector \mathbf{X} of (numeric and discrete) attributes and a discrete attribute Y , which denotes

the trustworthiness level. The attributes \mathbf{X} denote the three-level representation introduced in Sect. 2. For the instances of \mathcal{D} included into the set \mathcal{L} , the labels are known, while for \mathcal{U} , the values of Y are determined by the method.

Following the transductive paradigm, the method inputs both the full information represented by \mathcal{L} and the partially given information represented by \mathcal{U} , it learns a classification model and predicts the trustworthiness for the instances of the unlabelled part. This is done through an iterative convergence approach [9] aiming at improving the accuracy of the classification model through a procedure that converges to a configuration of the predictions on \mathcal{U} as accurate as possible. Before the iteration starts, the method performs a feature augmentation step, which generates an extended set of descriptive properties for every instance. It has the aim to make every instance “aware” about the distribution of the values (for each attribute) over the instances which are more correlated to it. These new attributes are updated during the iterative process, in order to “propagate” accurate predictions over correlated instances. To properly do that, we take only the predictions that could truly improve the classification model, so we do consider only the predictions with high confidence, generated for the current iteration, and feed them the learning process of the next iteration.

To identify data correlating to an instance the most, we introduce the notion of “neighborhoods” of instances. So, for every instance m , we build the spatio-temporal neighborhood $N(m, \delta_s, \delta_t)$ composed of the instances whose spatial distance from m does not exceed δ_s and which have been recorded within a time δ_t from m , formally $N(m, \delta_s, \delta_t) = \{p | p \in \mathcal{D}, \text{distance}(m.c, p.c) \leq \delta_s, m.t - p.t \leq \delta_t, \text{if } m.t \geq p.t, p.t - m.t \leq \delta_t, \text{otherwise}\}$. The terms $m.c$ and $p.c$ denote the spatial coordinates of the instances m and p respectively, while the terms $m.t$ and $p.t$ denote the recording times. This allows us to account for presence of the spatial autocorrelation and temporal auto-correlation [5] by nicely capturing the typical scenario in which the data recorded by the same device in a short time tend to have the same trustworthiness, as well as, the data, spatially close to each other, which have been recorded in a short time tend to have the same trustworthiness level.

Feature Augmentation. The new attributes are defined to compute variation summarization statistics of two classes. *Type 1:* given the base numeric attribute A , we build two new attributes, $AN(\text{mean})$ and $AN(\text{stDev})$, based on A . Both attributes are computed by aggregating A over the neighborhoods $N(m, \delta_s, \delta_t)$ constructed with maximum spatial distance δ_s and maximum temporal contiguity δ_t . Let m be an instance, $AN(u, \text{mean})$ and $AN(u, \text{stDev})$ are computed as the mean and standard deviation of the values of A falling in the neighborhood $N(m, \delta_s, \delta_t)$. Both the new attributes allow us to summarize average and variance, local to the neighborhoods, of the numeric attributes. *Type 2:* given the base discrete attribute A that takes d distinct values, we build d new attributes. These attributes represent the frequency histogram of A , as it is computed on the neighborhoods $N(m, \delta_s, \delta_t)$. In practice, we build one attribute for every distinct value of A . Let m be an instance, val be a distinct value of A , $AN(m, val)$ is computed as the frequency of val over the neighborhood $N(m, \delta_s, \delta_t)$.

Prediction Reliability. We measure the confidence of the labels predicted at each iteration, in order to select those more confident that are then fed back into the learning process for the next iteration. Intuitively, confident predictions should manifest the property of auto-correlation, so that similar labels can be plausibly propagated to

the neighbours. The higher the autocorrelation of the label with neighbour labels, the more confident its prediction. To define the measure, we estimate the presence of the predicted label associated to the instance m over the neighborhood $N(m, \delta_s, \delta_t)$ by quantifying the times in which the prediction on m is identical to the labels of its neighbours included in the set \mathcal{L} (labelled neighbours). The choice of comparing the predicted labels against those original of the set \mathcal{L} is done to provide validity to confidence estimation.

However, the temporal component should have a weight larger, compared to that spatial, because onsets and effects of the urban processes often depend on the timing of human lifestyles and daily periods more than phenomena related to the spatial dislocation. To encode this, we inject the temporal distances between the instance m and its neighbours into the confidence measure and assign weights to the distances dependently on their values. In practice, we build two sorted sets with the neighbours, one set with the instances recorded before the instance m and one set with the instances recorded after m . So, the weights are determined by the number of instances that separate m from the neighbours. The confidence measure for the predicted label done on the instance m is so formulated:

$$\mathcal{R}(m) = \frac{\sum_{p \in \{N(m, \delta_s, \delta_t) \cap \mathcal{L}\}} (\sigma(m.t, p.t) \times (\text{equal}(\bar{y}, p.y)))}{\sum_{p \in \{N(m, \delta_s, \delta_t) \cap \mathcal{L}\}} (\sigma(m.t, p.t))}, \quad (1)$$

where $\sigma(m.t, p.t)$ determines the weight associated to every comparison (that is, temporal distance between m and p), $\text{equal}(\bar{y}, p.y)$ is 1 when the prediction \bar{y} equals $p.y$, 0 otherwise. It has values in the range $[0, 1]$, where 1 denotes the highest number of occurrences of the label \bar{y} in the neighborhood and therefore corresponds to the largest confidence, while 0 indicates poor confidence.

Transductive Classification Process. The transductive classification process essentially carries on learning and prediction along two stages, that is, initialization and iteration, as described in the following.

In the initialization stage, it performs three main operations.

(1) For every instance of $\mathcal{D} = \mathcal{L} \cup \mathcal{U}$, it constructs the respective neighborhood with the instances which have spatial distance less than δ_s and temporal distance less than δ_t . This is done by considering the spatial coordinates and recording time of m , as illustrated in the formulation of $N(m, \delta_s, \delta_t)$. The values of δ_s and δ_t are set by the user. Then, for each attribute X of the attribute vector \mathbf{X} , it generates new attributes of *Type 1* and *Type 2*, dependently on whether X is numeric or discrete. The computation considers both the labelled instances and unlabelled instances of each neighborhood $N_m \in \mathbf{N}_{st}$ previously determined. Finally, it generates new attributes of *Type 2* for the label-attribute Y with the procedure used for the attributes \mathbf{X} . In this case, the computation considers only the labelled instances of each neighborhood $N_m \in \mathbf{N}_{st}$ because the unlabelled instances have no prediction for the attribute Y at the initialization stage. Clearly, all the instances will have the same set of new attributes, while the values are specific per instance m and depend on the data distribution over the respective neighbors N_m .

(2) The algorithm learns a classification model F from the training set \mathcal{L} , which is now represented with an augmented feature space $\mathbf{X} \times \mathbf{X}N \times Y \times \mathbf{Y}N$. This allows us inject the auto-correlation into the learning process since the beginning, without making the subsequent computation burden because the new attributes are built once only.

(3) The model F is finally used to initialize the unknown labels of the instances \mathcal{U} , which are stored as $\hat{\mathcal{U}}$. This way, the predictor F is able to estimate the data trustworthiness by considering additionally the contextual information provided by the nearby MCS data (spatial auto-correlation) and by the readings done by the same devices in the past (temporal auto-correlation), besides of information the devices record in themselves.

In the iteration stage, we aim at improving the predictive accuracy of F and, to this end, we exploit the auto-correlation property from the most confident predictions inferred along the iterations. Basically, the method carries out the following operations.

(1) For every instance m previously labelled and stored in the set $\hat{\mathcal{U}}$, it computes the confidence values by comparing the prediction of m against the originally known labels of the instances of \mathcal{L} included in the neighborhood N_m , as illustrated in the formula 1.

(2) The confidence values $\mathbf{R}_{\mathcal{U}}$ are sorted and then we pick only the first $size_r$ instances with higher rank, being considered as mostly reputable. The assigned labels (stored in B) will be maintained as such because they will contribute to the subsequent operations, since the instances are now “stabilized”. However, these instances will have a role different from the originally labelled instances \mathcal{L} , in accordance with the philosophy of the transductive learning and, in fact, they are removed from the target set (unlabelled instances) \mathcal{U} and moved in the set $\hat{\mathcal{U}}$, which is different from \mathcal{L} .

(3) The new configuration of labels, caused by the reduction of \mathcal{U} and extension of $\hat{\mathcal{U}}$, is propagated over all the instances (\mathcal{L} , \mathcal{U} , $\hat{\mathcal{U}}$) through the update of the new attributes. It should be noted that only the attributes $\mathbf{Y}N$ are influenced by the update, since those of the set $\mathbf{X}N$ remain unchanged, being derived by the attributes \mathbf{X} .

(4) In accordance with the transductive learning, the classification model F is (re-)trained on the originally labelled instances \mathcal{L} , which are now “aware” about the new labeling scenario. So, the predictor F can leverage the *i*) confidence of the predictions ($\hat{\mathcal{U}}$) and *ii*) reinforced configuration of the descriptive attributes, in order to improve the accuracy of the instances left in \mathcal{U} .

This iterative procedure stops when one of the two stopping criteria is satisfied, specifically, either the set \mathcal{U} is empty or the number of iterations completed reaches a user-defined threshold. The depletion of \mathcal{U} is guaranteed as every iteration removes a portion of instances equal to the threshold $size_r$ (user-defined).

4 Empirical Evidence

In order to provide empirical evidence to the proposed solution for trustworthiness assessment, we performed experiments on a MCS-originated dataset of noise level readings recorded by a set of 5 smartphone devices, moving in a 8 km² area in Lecce (Apulia, Italy), during the period 2019/05/27–2019/07/01. Each measurement point is time- and geo-referenced and enriched with contextual data provided from additional smartphone-embedded sensors (e.g., accelerometer, proximity sensor, etc.) and device’s metadata as well.

Specifically, we have 4335 readings uniformly distributed over the categories (991 instances for *not reliable*, 1782 instances for *poorly reliable* and 1562 for *reliable*), so we have no imbalanced concern for the classification task. For instance, noise readings outside sensor’s range are classified as unreliable. Similarly, noise readings with low amplitude (i.e.,]+20;+50] dB(A)), are considered poorly reliable when the proximity

sensor indicates low values (i.e., less than 5 cm), as an obstructed microphone affecting noise readings can be inferred.

We arrange experiments aiming at training and testing the classification models. In particular, we performed a quantitative evaluation on the predictive capabilities of the transductive approach in performing accurate inferences on the trustworthiness level of the unknown noise pollution readings. The accuracy was measured in terms of the F-score and averaged over 5 trials executed according to the inverse 5-fold cross validation. More precisely, for each trial, the learner is trained on one fold (which represents \mathcal{L}) and tested on the set \mathcal{U} composed of the remaining four folds. We guaranteed that the training set \mathcal{L} was balanced. By following the transductive setting, the set \mathcal{L} contains a smaller part of the whole dataset, and, more precisely, it has a percentage of 10% balanced over the three classes. The accuracy was estimated on three variants of the method, which were built by using three base learners to train the predictor F . Specifically, we integrated the classification algorithms of *Decision Tree* (DT), *Random Forest* (RF) and *Logistic Regression* (RF).

The three variants of the method were tested in two main experimental setups, *i*) size of the set of confident predictions ($size_r$), and *ii*) size of the neighborhoods in terms of the values of the thresholds δ_s and δ_t . In particular, we considered three different values of $size_r$, that is, 5%, 10%, 15% of \mathcal{U} and three different neighborhood configurations, that is, $\delta_s = 250$ m and $\delta_t = 10$ min (thereafter, n_10_250), $\delta_s = 500$ m and $\delta_t = 5$ min (thereafter, n_5_500), and $\delta_s = 500$ m and $\delta_t = 10$ min (thereafter, n_10_500), which let us build neighborhoods with different sizes (number of instances contained), that is, 4, 5 and 7, on average, respectively. Neighborhoods which, initially had no labelled instance, were extended with the instance of \mathcal{U} which is closest to the samples already present. For a fair comparison, we fix the maximum number of iterations to 10, which allows the experimental trials terminate under different conditions.

The F-score values computed along the iterations with n_10_250 are reported in the Fig. 1, while those computed with n_5_500 are reported in the Fig. 2, finally, the F-score values computed with n_10_500 are reported in Fig. 3.

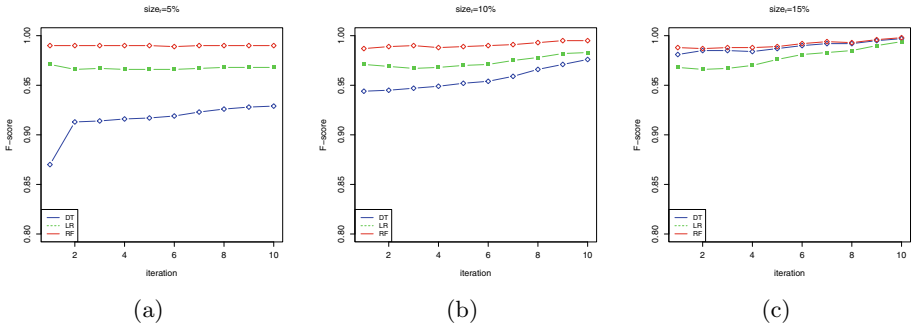


Fig. 1. The values of F-score computed on the unlabelled instances along the iterations when $\delta_s = 250$ m, $\delta_t = 10$ min.

The first consideration we can do is on the number of the iterations. Regardless of the neighborhood configuration, the predictive accuracy in most cases increases as new iterations are performed. The highest gain accuracy is obtained at the initial iterations,

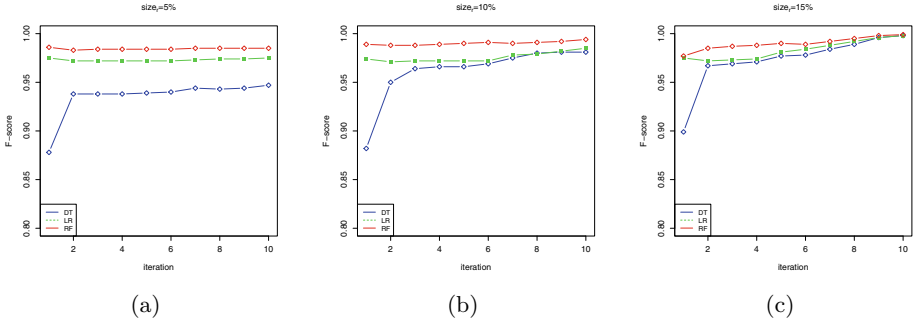


Fig. 2. The values of F-score computed on the unlabelled instances along the iterations when $\delta_s = 500$ m, $\delta_5 = 5$ min.

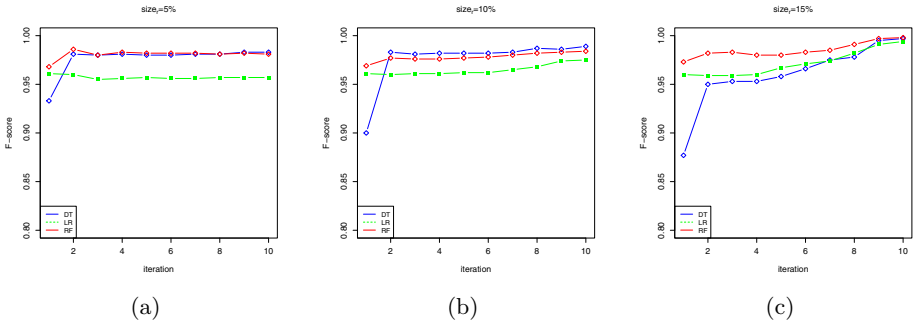


Fig. 3. The values of F-score computed on the unlabelled instances along the iterations when $\delta_s = 500$ m, $\delta_5 = 10$ min.

which indicates the classifier benefits from the best confident predictions since at the early. This confirms the effectiveness of the iterative learning approach. We should also note that acceptable F-score values can be reached even before the execution of 10 iterations. Clearly, this leads benefits from the viewpoint of the running times.

Another consideration deserves the behaviour of the accuracy with respect to the number of confident predictions selected during the iterative process. We see that the lowest value of $size_r$ (5%) guarantees the more stable (less variable) F-score response over the three base learners, meaning that the refinement process of the predictor F allows effectively us to improve the predictions of the instances, which are selected later, instead of removing them from \mathcal{U} at the early. This is confirmed by the higher variance of the F-score when $size_r$ is 15%.

As to the neighborhoods, the indication we can draw is the higher accuracy is obtained with the larger number of neighbours. In fact, we see F-score values greater than 0.95 only for the configuration n_10_500, on the contrary, for n_5_500 and n_10_250, the accuracy is under the threshold of 0.95. This is why the use of greater neighborhoods generally leads to increase the “awareness” of the predictor F about the surrounding instances of a target instance and, consequently, improve the prediction of

the trustworthiness levels. This confirms the advantages of the feature augmentation for “contextual” noise readings.

5 Conclusions

Enriching Mobile Crowd Sensing data with contextual details is essential to maximize the effectiveness of contributed data without explicitly requesting additional information to the end-user. This paper proposes to leverage on machine learning to contextualize the gathered observations in a way that accounts for the properties of the crowdsensors, spanning the device characteristics, the end-user’s behavior and the environment. We have developed a transductive learning method able to learn on both fully known devices and partially labelled data. A validation session on the recurrent scenarios of urban noise pollution has been conducted to refine the accuracy of the trustworthiness prediction and quantitatively measure the influence of the working conditions on the accuracy.

Acknowledgments. This work was supported in part by the research project “APOLLON - environmental POLLution aNalyzer”, within the “Bando INNONETWORK 2017” funded by Regione Puglia (Italy) in the framework of the “FESR - Fondo Europeo di Sviluppo Regionale” - “POR Puglia FESR 2014-2020 - Asse Prioritario 1 - Ricerca, sviluppo tecnologico, innovazione - Azione 1.6”.

References

1. Guo, B., et al.: Mobile crowd sensing and computing: the review of an emerging human-powered sensing paradigm. *ACM Comput. Surv.* **48** (2015). <https://doi.org/10.1145/2794400>
2. Huang, K.L., Kanhere, S.S., Hu, W.: Are you contributing trustworthy data?: the case for a reputation system in participatory sensing. In: 13th International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems, MSWiM 2010, pp. 14–22 (2010). <https://doi.org/10.1145/1868521.1868526>
3. Karkouch, A., Mousannif, H., Moatassime, H.A., Noël, T.: Data quality in internet of things: a state-of-the-art survey. *J. Netw. Comput. Appl.* **73**, 57–81 (2016). <https://doi.org/10.1016/j.jnca.2016.08.002>
4. Loglisci, C., Appice, A., Malerba, D.: Collective regression for handling autocorrelation of network data in a transductive setting. *J. Intell. Inf. Syst.* **46**(3), 447–472 (2016). <https://doi.org/10.1007/s10844-015-0361-8>
5. Loglisci, C., Ceci, M., Malerba, D.: Discovering evolution chains in dynamic networks. In: *New Frontiers in Mining Complex Patterns - Revised Selected Papers*, pp. 185–199 (2012). https://doi.org/10.1007/978-3-642-37382-4_13
6. Loglisci, C., Malerba, D.: Leveraging temporal autocorrelation of historical data for improving accuracy in network regression. *Stat. Anal. Data Min.* **10**(1), 40–53 (2017). <https://doi.org/10.1002/sam.11336>
7. Longo, A., Zappatore, M., Bochicchio, M.A.: Collaborative learning from mobile crowd sensing: a case study in electromagnetic monitoring. In: 2015 IEEE Global Engineering Education Conference (EDUCON), pp. 742–750 (2015)
8. Louta, M., Mpanti, K., Karetos, G., Lagkas, T.: Mobile crowd sensing architectural frameworks: a comprehensive survey. *IISA* **2016** (2016). <https://doi.org/10.1109/IISA.2016.7785385>

9. Neville, J., Jensen, D.: Iterative classification in relational data. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence. AAAI Press (2000)
10. Sailhan, F., Issarny, V., Tavares-Nascimento, O.: Opportunistic multiparty calibration for robust participatory sensing. In: 14th International Conference on Mobile Ad Hoc and Sensor Systems, pp. 435–443, October 2017. <https://doi.org/10.1109/MASS.2017.56>
11. Wu, C., Luo, T., Wu, F., Chen, G.: EndorTrust: an endorsement-based reputation system for trustworthy and heterogeneous crowdsourcing. In: 2015 IEEE Global Communications Conference, pp. 1–6 (2015). <https://doi.org/10.1109/GLOCOM.2014.7417352>
12. Yurur, O., Liu, C.H., Sheng, Z., Leung, V.C., Moreno, W., Leung, K.K.: Context-awareness for mobile sensing: a survey and future directions. *IEEE Commun. Surv. Tutor.* **18**(1), 68–93 (2016). <https://doi.org/10.1109/COMST.2014.2381246>
13. Zappatore, M., Longo, A., Bochicchio, M.A.: Crowd-sensing our smart cities: a platform for noise monitoring and acoustic urban planning. *J. Commun. Softw. Syst.* **13**(2), 53–67 (2017)
14. Zappatore, M., Longo, A., Bochicchio, M., Zappatore, D., Morrone, A., De Mitri, G.: A crowdsensing approach for mobile learning in acoustics and noise monitoring, 04–08 April 2016, pp. 219–224 (2016). <https://doi.org/10.1145/2851613.2851699>



A Hierarchical-Based Web-Platform for Crowdsourcing Distinguishable Image Patches

Ayman Hajja^(✉) and Justin Willis^(✉)

Department of Computer Science, College of Charleston, 66 George Street,
Charleston, SC 29424, USA
hajjaa@cofc.edu, willisjk@cofc.edu

Abstract. In this work, we present an open-source web-platform for crowdsourcing a unique kind of image labels. This is done by introducing image segments that we refer to by the term “distinguishable patches”; as the name implies, a distinguishable patch is a small segment of an image that identifies a particular object. Although these distinguishable patches will naturally be part of the object; more often than not, these distinguishable patches combined will not cover the entire body of the object, which makes the nature of the data collected distinct and rather different than what would be obtained from a traditional image segmentation system. To minimize human labeling efforts while maximizing the amount of labeled data collected, we introduce a novel top-bottom hierarchical approach to automatically determine the size and the location of the patches our system will present to individuals (referred to by “workers”) for labeling, based on previously labeled patches. The three processes of: determining the size and location of the patch, assigning a particular patch to the right worker, and the actual cropping of these patches, all happen in real-time and as the workers are actively using our web-platform. As far as the authors are aware, this unique form of image data has not been collected in the past, and its impact has not been explored, which makes this work highly valuable and important to the research community. One of the many ways that the authors are interested in using these distinguishable patches would be to improve the accuracy of a machine learning image classification system, by providing an enriched dataset of images that not only contain a single label for each image, but rather a spatial distribution of the distinguishability of an object in each image.

Keywords: Crowdsourcing · Labeling · Human computation · Image classification · Machine learning · Web application

1 Introduction and Background

The use of large volumes of labeled data has a direct impact on the accuracy of many machine learning classification algorithms; therefore, researchers and

data science practitioners have been seeking new ways to collect labeled data effectively and accurately. One of the most practical ways of collecting labeled data is through the use of crowdsourcing; one form of crowdsourcing can be defined as the act of engaging many (often low-stake) individuals to collectively collect ground truths for a particular type of data instances (e.g. images, audio segments, video snippets), for the purpose of using these labeled data points to improve the accuracy of a machine learning classifier. It is worth noting here that other forms of crowdsourcing, such as ranking or proposing ideas or solutions, exist; however, crowdsourcing for machine learning purposes is what we are concerned about in this work.

There have been many proposed crowdsourcing systems for labeling images. “Visual Madlibs” [5] is a fill-in-the-blank system that collects labels about people and objects, their appearances, activities, and interactions. “Peekaboom” [4] on the other hand is a game-based image crowdsourcing platform for locating objects in images. Other more specific domains such as medical image analysis have also been interested in creating crowdsourcing platforms to gather ground truth data; one such system is a smartphone app that crowdsources the analysis of bladder cancer TMA core samples [3]. From the domain of audio analysis, Hajja et al. introduced a system to label short audio segments to help improve stutter detection classifiers [2]. Amazon Mechanical Turk (MTurk) [1] is perhaps one of the most known public platforms for crowdsourcing ground truths; it allows “requesters” to create a crowdsourcing study consisting of performing mini-tasks, which need to be undertaken by “workers”. With the exception of MTurk, most of these existing platforms were designed to crowdsource labels of a specific dataset and are not publicly available for other researchers to utilize.

Objects in images are generally identified by either assigning a single label to the entire image, or by segmenting the object within the image and assigning that object a label. Crowdsourcing labeled images using the first image-based approach can be extremely effective due to its simple nature; and for that reason, many existing platforms such as MTurk can be utilized to collect such data. The second more specific segment-based approach requires manual segmentation of the object in the image, which tends to be substantially more tedious and time consuming; hence not easily feasible for collecting large sums of labeled data.

In this research paper, we present an open-source web-platform for crowdsourcing a unique kind of image labels that strikes a delicate balance between the effectiveness benefit of the first approach, and the specificity benefit of the second approach. This is done through the introduction of image segments that we refer to by the term ‘distinguishable patches’; as the name implies, a distinguishable patch is a small segment of an image that identifies a particular object. Although these distinguishable patches will naturally be part of the object; more often than not, these distinguishable patches combined will not cover the entire object, which makes the nature of the data collected distinct and rather different than what would be obtained by a traditional image segmentation system. As will be shown below, our web-platform provides a high level of customization for individuals and teams to create their own unique data collection studies or

campaigns. Some of the parameters that campaign administrators can set include the sizes of image patches, the desired level of specificity and depth of patch generation hierarchy, and the ‘stop threshold’ which will be used to determine the optimal patch-worker assignment.

Instead of using the concept of crowdsourcing to collect a single label for each image, our system is designed to collect a distribution of labels for a set of dynamically-generated segments in each image. This is done through an image segmentation process that takes place in real-time using a cloud-based back-end logic implemented as part of our system. The criteria for which the segmentation process takes place can be set and customized by the campaign administrator. We will provide a detailed description of our system and show screenshots of our web-platform in the following section.

It is worth noting here however, that the system we are presenting here serves an entirely different purpose than a system that segments images. Our system identifies patches that can distinguish an object, as opposed to identifying the object itself. We believe that from a machine learning training perspective, that seemingly subtle difference will have a drastic effect on the effectiveness of the classifier training process. We have conducted a simple experiment to demonstrate that distinguishable patches do not always overlap with the entire object in the image. Figure 1 shows a heatmap displaying the object “distinguishability” in two images. Red regions show patches that workers were able to identify the object (i.e. cat) in; these are the patches that our system is interested in crowdsourcing.

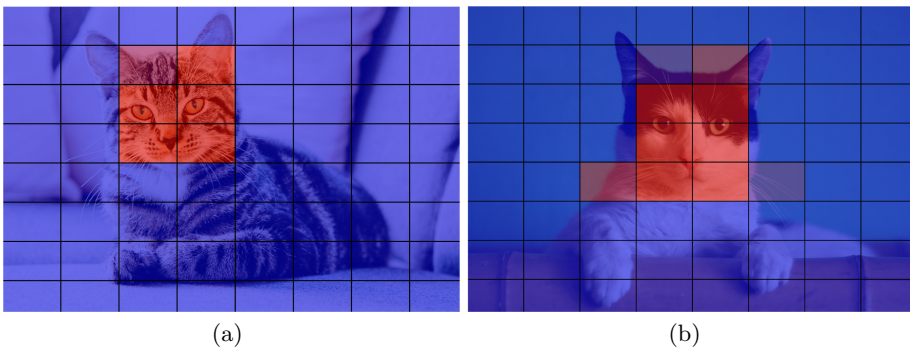


Fig. 1. Two sample outputs of our crowdsourcing platform showing a spatial distribution of object “distinguishability” (Color figure online)

2 Our System

We define patch labeling as the process of segmenting an image into different parts, which we refer to as “patches”, and labeling each patch individually. The segmentation process, which determines the number (and location) of patches

in an image, is established dynamically by our system. The basis behind the dynamic nature of the platform will be guided by the other previously labeled patches for the same image. In this section, we will delve into the logistics of how our system crops and distributes these patches, and we will provide a detailed summary of the parameters and settings that users can specify and tune to create their own unique data collection campaigns.

One of our primary goals for building this platform, is to give researchers the ability to design and administer image crowdsourcing projects according to their needs. Thus, we needed to build a highly-customizable platform that will adapt to the different needs of campaign administrators. In this paper, we will use the term “administrator” or “admin” to refer to an individual (or team) responsible for administering an image labeling project or campaign; although there are no restrictions on who can use the system, we anticipate most of our administrators to be academic researchers and data scientists. The term “worker” on the other hand, will refer to an individual who labels these image segments (referred to by “patches”); again, although anyone can label a public study, we anticipate a good portion of these workers to be research lab members or students.

2.1 The Grid System

Our system starts by dividing the images into an administrator-defined $m \times n$ grid, such that m is the number of rows and n is the number of columns. Each rectangle in the generated grid is referred to by the term “unit”. The administrator also needs to define the size of the largest patch set (referred to by “initial”, or “L1” patches) that will be labeled by workers. As will be shown below, the concept of patch levels (L1, L2, L3, ...) will be one of the primary key concepts in this research work.

Figure 2(a) shows an example of an 8×8 image grid with a 2×2 initial patch size, resulting in 16 initial patches. Figure 2(b) on the other hand shows a 4×4 grid with a 2×2 initial patch size, which yields 4 different non-overlapping initial patches. These grid lines shown in Fig. 2 are displayed in real-time while the admin is experiment with different values, allowing them to visualize the generated patches and determine the most appropriate grid dimension.

In the next subsection, we will elaborate on the different uses for our grid system and show examples of the multi-level patch system.

2.2 Variable Patch Sizes

As mentioned above, one of the primary reasons for the introduction of our system was to generate an enriched image dataset to enhance the performance of a machine learning image classifier. To accomplish that, we needed to identify as many distinguishable patches as possible in each image, which implies that we are particularly interested in finding the set of smallest distinguishable patches. There are two reasons for that, the first being is that extracting smaller patches will yield a higher number of unique patches, which as a result will provide more training data instances for our classifier to utilize. The second reason for our

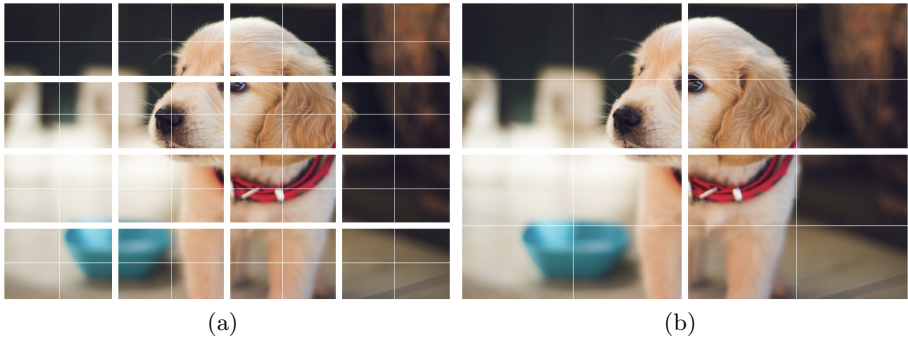


Fig. 2. (a) 8×8 image grid with a 2×2 initial patch size (b) 4×4 image grid with a 2×2 initial patch size.

interest in small patches is due to the information/size ratio; small distinguishable patches occur only when there is a highly distinguishable feature observed in that patch, such as cat whiskers, which can be highly valuable when training an image classification model. Having said that, the size of the smallest distinguishable patches depends on the object in the image, and it cannot be predetermined accurately by campaign administrators. For that reason, we have introduced a new way in which our system can extract these distinguishable patches from workers thorough a top-down hierarchical labeling approach.

Recall that campaign administrators need to specify at least the dimension of the image grid, and the dimension of the initial patch (L1). Each one of the initial patches is displayed to a group of workers (independently), for which they must assign a label. These labels are defined by the campaign administrator for the particular campaign. Figure 3 shows a screenshot of the worker's interface showing one of the four patches from the image shown in Fig. 2(b), along with three labels defined by the campaign administrator.

Campaign administrators also have the option of choosing to divide the set of initial (or L1) patches further. The resulting set of dividing an L1 patch is a set of L2 patches; L2 patches can be further divided into L3, so on and so forth. For example, assuming that the dimension of the image grid is 8×8 , the dimension of L1 patches is 4×4 , L2 patches is 2×2 , and L3 patches is 1×1 . Figure 4 shows the set all patches that could potentially get generated from our image. Note here that although Fig. 4(a) shows a 2×2 grid, the size of each cell in that grid is 4×4 (L1 dimension) provided that the image grid dimension is defined as 8×8 .

As you may have guessed, there are multiple sets of dimensions that will result in the same patches; for example, an L1 division of 9×9 and an L2 division of 3×3 is identical to an L1 division of 3×3 and an L2 division of 1×1 (assuming L2 is the highest level generated patches). Our system allows the administrator to specify any valid set of dimensions; in addition to that, our system will display the number of potentially generated patches for each level.

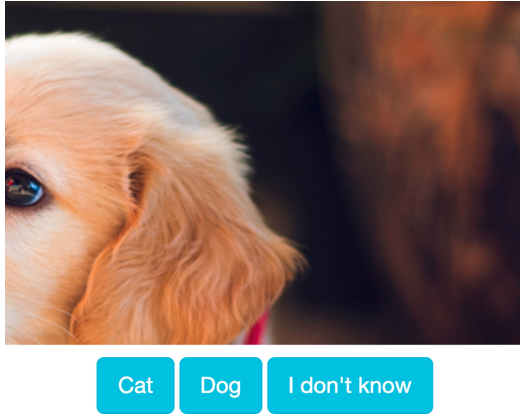


Fig. 3. A screenshot from the worker’s interface showing one of the four patches from the image shown in Fig. 2(b), along with three labels defined by the campaign administrator

Figure 5 shows a screenshot of the page on which the administrators specify the grid size of each level. Note here that these patches are only potentially generated since this will be determined based on the labels provided by previous workers, as will be explained in the next subsection.

2.3 Hierarchical Patch Generation

To minimize human labeling efforts, we have decided to adopt a top-down approach for generating the patches. Our system will start by asking workers to label all the L1 patches first, and based on the obtained labels for each patch, our system will either mark these patches with “unknown” or “distinguishable” label. Next, our system will divide all distinguishable L1 patches into their L2 sub-patches, and present them to workers, so on and so forth. This process will continue until we go through all levels (defined by campaign admin), or until all patches have been labeled with “unknown”, and hence no further division will be required. To provide few examples to demonstrate the idea of hierarchical patch generation, we would have to define three key terms:

1. **Stop threshold:** The stop threshold can be defined as the minimum percentage of agreeability that campaign administrators require for any patch to be marked “distinguishable”. As soon as a particular patch satisfies this agreeability threshold, and as long as the number of responses for that patch is within the minimum and maximum range (defined below), our system will not present this patch for additional labeling.
2. **Minimum number of responses for a patch:** As the name implies, this value is used to define a lower bound for the number of responses obtained from workers for a particular patch. For example, if this value is set to 3,

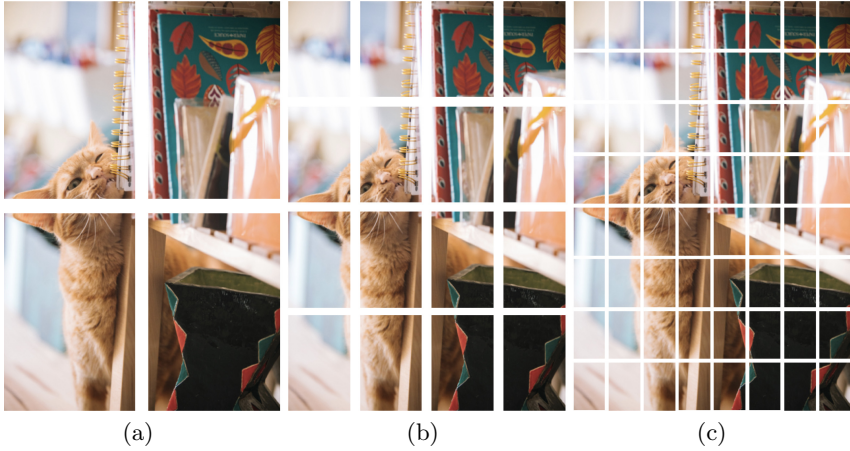


Fig. 4. (a) L1 patches (b) L2 patches (c) L3 patches

Image Grid Dimension by

L1 Dimension by Number of L1 patches will be 4

L2 Dimension by Number of L2 patches will be 16

L3 Dimension by Number of L1 patches will be 64

[Add a new level](#)

Fig. 5. Screenshot of what the admins see when setting the dimensions of the patch levels.

that simply means that every patch needs to be presented to at least three different workers.

3. **Maximum number of responses for a patch:** This value is used to define an upper bound for the number of responses obtained from workers for a particular patch. As we will demonstrate in the next few examples, this value will often not be reached, since the stop threshold and minimum number constraints may be satisfied prior to reaching this value.

Next, we will show an example for demonstrating the hierarchical patch generation in action. In this example, we will use the levels defined in Fig. 4. Let us assume the following:

1. the stop threshold is 70%,
2. the minimum responses is 3 and the maximum responses is 7, and
3. the labels provided to workers are “Cat”, “Dog”, and “I don’t know”.

Let us also agree on a patch numbering system that we will use in this example. If the patch is an L1 patch, then its ID will be L_m such that m is the number of the patch, counting from top left to bottom right; for example, L_1 would be the top left patch in Fig. 3(a), L_2 would be the top right patch in Fig. 3(a), L_3 would refer to the bottom left patch, and L_4 would be the bottom right patch in Fig. 3(a). If the patch is an L2 patch, then we will use two subscripts, the first one referring to the L1 patch that it was generated from, and the second subscript will be used to indicate the number of the patch counting from top left to bottom right; for example, the top right patch in Fig. 3(b) is referred to by $L_{2,2}$ since it is part of patch L_2 and it is the second patch; the bottom right patch in Fig. 3(b) would be identified by $L_{4,4}$ since it is part of patch L_4 , and it is the fourth and last patch (within that sub-patch).

Table 1 below shows a valid patch-worker assignment, with potential labels provided by workers; we will use this table to demonstrate this example. In Table 2, we provide a description of how our system will behave based on the labels provided in Table 1.

Table 1. A hypothetical example of possible labels provided by workers

Time	Patch ID	Worker ID	Label provided
0	L_1	15	Cat
1	L_1	9	Cat
2	L_2	9	I don't know
3	L_1	12	Cat
4	L_2	15	I don't know
5	L_2	5	I don't know
6	L_3	5	Cat
7	L_4	12	I don't know
8	L_4	5	I don't know
9	L_3	9	I don't know
10	L_4	9	I don't know
11	L_3	12	Cat
12	$L_{1,2}$	26	I don't know
13	L_3	29	Dog
14

Table 2. A detailed description of how our system behaves based on the labels provided by workers shown in Table 1

Time	System behavior
3	L_1 satisfies the three criteria (3 is more than or equal to minimum responses, 3 is less than or equal to maximum responses, and 3 workers provided the label “Cat” out of the 3 labels provided, which means that the agreeability is 100% which is higher than or equal to the stop threshold 70%). Since L_1 satisfied the three conditions, we will crop it into its L_2 level patches and repeat
5	L_2 now has been labeled “I don’t know” by three workers. Here, we would stop asking workers to label L_2 and we do not divide it into its Level 2 (L_2) patches. The reason for this decision is because even if the rest of workers (4, since the maximum responses is 7) agree on a label (say “Cat”), the agreeability will be 4/7 which is less than the stop threshold
10	Similar to L_2 , L_4 has been labeled with “I don’t know” by three workers, so we stop asking users to label L_4 , and we do not divide that patch any further
11	Here, L_3 has been labeled “Cat” by two workers and “I don’t know” by one worker. The three criteria have not been satisfied yet since the stop threshold (70%) has not been met. That being said, there is still a possibility that if we ask more workers, we will satisfy all three conditions
12	We can observe two things here: 1. Our system is now asking workers to label patches from the second level (since for patch L_1 , it has been agreed that it exhibits the label “Cat”), and 2. We notice that the worker ID is now different, and that is because the four workers with ID’s 5, 9, 12, and 15 have seen L_1 patch (which contains $L_{1,2}$) and therefore may bias their response
13	After the label provided at Time 13, we have 2 “Cat” labels for L_3 , 1 “I don’t know” label, and 1 “Dog” label. Our system will automatically stop asking workers to label this patch since there is no way that we will reach agreeability higher than or equal to the stop threshold

3 Conclusion and Future Work

In this paper, we described an open-source web-platform designed to provide researchers and data scientists with the ability to efficiently and effectively crowdsource labeled patches from images. The nature of the crowdsourced distinguishable patches presented in this paper is unique, and as far as the authors are aware has not been explored in the past.

The system that we have designed adopts a top-bottom hierarchical-based approach to automatically determine the size and the location of the patches presented to workers; therefore minimizing human labeling efforts while maximizing the amount of labeled data collected, without adding human bias.

One of the areas in which future work can be conducted involves exploring ways to improve the training of image classification models by utilizing these

collected distinguishable patches; either by treating each distinguishable patch as a unique labeled instance, or by augmenting the spatial distribution of workers' responses to existing single-label image datasets.

Another area of future work involves extending our platform to crowdsource labels from sets of many isolated patches within an image. This idea of multi-patch labeling is particularly interesting since it may reveal interesting findings about how combining independent (and indistinguishable) patches could result in meaningful groups of image patches.

References

1. Buhrmester, M., Kwang, T., Gosling, S.D.: Amazon's Mechanical Turk: a new source of inexpensive, yet high-quality data? (2016)
2. Hajja, A., Hiers, G.P., Arbajian, P., Raś, Z.W., Wiczorkowska, A.A.: Multipurpose web-platform for labeling audio segments efficiently and effectively. In: Ceci, M., Japkowicz, N., Liu, J., Papadopoulos, G.A., Raś, Z.W. (eds.) ISMIS 2018. LNCS (LNAI), vol. 11177, pp. 179–188. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01851-1_18
3. Smittenaar, P., et al.: Harnessing citizen science through mobile phone technology to screen for immunohistochemical biomarkers in bladder cancer. *Br. J. Cancer* **119**(2), 220–229 (2018)
4. Von Ahn, L., Liu, R., Blum, M.: Peekaboom: a game for locating objects in images. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 55–64 (2006)
5. Yu, L., Park, E., Berg, A.C., Berg, T.L.: Visual Madlibs: fill in the blank description generation and question answering. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2461–2469 (2015)



Performing Arithmetic Using a Neural Network Trained on Digit Permutation Pairs

Marcus D. Bloice^{1(✉)}, Peter M. Roth², and Andreas Holzinger^{1,3}

¹ Institute for Medical Informatics, Statistics, and Documentation,
Medical University of Graz, Graz, Austria

{marcus.bloice, andreas.holzinger}@medunigraz.at

² Institute of Computer Graphics and Vision, Graz University of Technology,
Graz, Austria

p.m.roth@ieee.org




³ xAI Lab Alberta Machine Intelligence Institute, T6G 2H1 Edmonton, Canada

Abstract. In this paper a neural network is trained to perform simple arithmetic using images of concatenated handwritten digit pairs. A convolutional neural network was trained with images consisting of two side-by-side handwritten digits, where the image's label is the summation of the two digits contained in the combined image. Crucially, the network was tested on permutation pairs that were not present during training in an effort to see if the network could learn the task of addition, as opposed to simply mapping images to labels. A dataset was generated for all possible permutation pairs of length 2 for the digits 0–9 using MNIST as a basis for the images, with one thousand samples generated for each permutation pair. For testing the network, samples generated from previously unseen permutation pairs were fed into the trained network, and its predictions measured. Results were encouraging, with the network achieving an accuracy of over 90% on some permutation train/test splits. This suggests that the network learned at first digit recognition, and subsequently the further task of addition based on the two recognised digits. As far as the authors are aware, no previous work has concentrated on learning a mathematical operation in this way.

1 Introduction

The aim of this study is to attempt to find experimental evidence that would suggest that a network can be trained to perform the task of addition, when supplied with image data containing two digits that should be summed. To ensure that the network has indeed learned this, and is not simply mapping images to labels, a constraint was applied whereby the network is tested with a held back test set of previously unseen permutation pairs. This forces the network to learn more than simply a mapping between individual images and labels as it is tested using digit combination pairs that it has not seen, meaning a direct mapping from an image or shape to a label would not function.

Table 1. Example input images and their corresponding labels. Each image consists of two side-by-side MNIST digits merged into one single image where each image’s label is the summation of the two digits. The label contains no information as to the individual digits contained within the image, nor is there any indication given to the network prior to training that each image consists of two digits or otherwise.

Image	Interpretation	Label
	$5 + 0$	5
	$8 + 5$	13
	$9 + 9$	18

To test this hypothesis, a network was trained with data generated using 90 of the possible 100 combinations of the digits 0–9 up to length 2. Once trained, the network was tested by inputting images based on the remaining 10 permutation pairs, and was required to predict the summations for them. Some examples of the generated samples can be seen in Table 1. 1,000 samples were generated for each permutation pair. This was repeated 10 times for 10 different permutation pair train/test splits.

2 Background Work

There has been previous work relating to this experiment. As opposed to most work, however, the goal of this study was not to recognise digits, extract their numerical values from the images, and then perform (after the network’s character recognition procedure) some mathematical function on the numerical values. The task of this experiment was to learn if the network could learn the logical task of the mathematical operation itself using an end-to-end approach. For example, [3] experimented with computer generated image data, however as output the network was trained to produce images containing the summations. Their work concentrated on the visual learning of arithmetic operations from images of numbers. In contrast, the work presented here outputs its predictions as a real number. Their approach used numbers of longer lengths and were therefore also able to generate many thousands of training samples, despite not using hand written digits. The input consisted of two images, each showing a 7-digit number and the output, also an image, displayed a number showing the result of an arithmetic operation (e.g., addition or subtraction) on the two input numbers. The concepts of a number, or of an operator, are not explicitly introduced. Their work, however, was more akin to the learning of a transformation function, rather than the task of learning a mathematical operation. Other operations, such as multiplication, were not learnable using this architecture. Some tasks were not learnable in an end-to-end manner, for example the addition of Roman numerals, but were learnable once broken into separate sub-tasks: first perceptual character recognition and then the cognitive arithmetic sub-task.

Similarly, a convolutional neural network was used by [6] to recognise arithmetic operators, and to segment images into digits and operators before performing the calculations on the recognised digits. This again is different to the approach described here, as it is not an attempt to learn the operation itself, but to learn to recognise the operator symbols and equations (and perform the mathematics on the recognised symbols).

In [8], the authors addressed the task of object counting in images where they applied a learning approach in which a density map was estimated directly from the input image. They employed convolutional neural networks with layered boosting and selective sampling. It would be possible to create an experiment based on their work, that would perform arithmetic by counting the values of domino tiles, for example.

Until now, as far as the authors are aware, no work has concentrated on learning the actual mathematical operation itself. Previous work tends to concentrate on first recognising digits and operators within images, and then to perform the mathematical operations separately, after this extraction has been carried out. In the case of this work, an end-to-end algorithm has been developed that performs the digit recognition, representation learning of the values of the digits, and performs the arithmetic operation.

3 Experiment

A convolutional neural network was trained to perform arithmetic on images consisting of two side-by-side hand-written digits. Each image's corresponding label is the sum of the two digits, and the network was trained as a regression problem. For the digits 0–9, up to length 2, there are 100 possible permutation pairs. For each permutation pair, 1,000 unique images were generated using the MNIST hand written digit database [4]. MNIST was chosen due to its familiarity in machine learning, and because LeNet5 (used as the basis for the network trained in this work) clearly functions well on this dataset. Also, MNIST continues to be a focus of research and is used as a benchmarking dataset to this day [2, 7].

Training was performed on images generated from a random 90-long subset of the possible permutations, and testing was performed on images based on the remaining 10 permutations. A number of example input images and their labels are shown in Table 1 where it can be seen that a single input image consists of two MNIST digits side-by-side, and the image's label is the summation of the two digits. For each permutation, 1,000 combined images are generated resulting in 100,000 samples that are separated into a training and test set based on the permutation pairs.

The task of the experiment was to train a neural network with data generated from a subset of the possible 100 permutations, that when presented with a new samples, generated from the unseen permutation pairs, the network would be required to predict the correct summation. This was done in order to ascertain whether a network could learn a simple arithmetic operation such as addition,




given only samples of images and their summations and no indication as to the value of each individual digit contained within the image, while only being trained on a subset of all possible permutation pairs and tested on the remaining pairs.

In summary, the experimental setting is as follows:

- By permutations, it is meant all possible combinations of the digits 0–9 of length 2. Formally, if the set of digits $D = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, all possible permutations is the Cartesian product of $D \times D$, which we define as P , so that $P = \{(0, 0), (0, 1), (0, 2), \dots, (9, 8), (9, 9)\}$.
- Of the 100 possible permutations pairs P , a random 90 are used as a basis to train the network and the remaining 10 pairs are used as a basis to test the network. These are the training permutations, P_t , and the test permutations, P_v . This permutation train/test split was repeated 10 times as a 10-fold cross validation.
- For each permutation, 1,000 samples are generated. So, for each of the permutations in P , 1,000 concatenated images are generated using random MNIST digits \mathbf{M} (appropriate for that permutation).
- By appropriate this means that, for example, generating an image for the permutation pair (3, 1) a random MNIST digit labelled 3 is chosen and a random MNIST digit labelled 1 is chosen and these images are concatenated to create a single sample for this permutation pair. This means each sample is likely unique (likely, as each image is chosen at random with replacement, see Table 2). The generated images are contained in a matrix \mathbf{X} , where \mathbf{X}_t are the training samples and \mathbf{X}_v are the test samples.
- For the generation of the training set images, \mathbf{X}_t , only images from the MNIST training set, \mathbf{M}_t , are used.
- For the test permutation images, \mathbf{X}_v , only images from the MNIST test set, \mathbf{M}_v , are used.
- The network is not given any label information regarding each individual digit within the concatenated images, only the summation is given as label data.
- The permutations pairs in the test set are not seen during training. This means the training set and test set are distinct in two ways: they contain different permutations pairs that do not overlap, and the individual MNIST images used to generate each permutation sample do not overlap between the training set and test set.

The decision to train the network as a regression problem is done for the following reasons. First, the number of output neurons would change depending on the train/test split. For example, the sum of (9, 9) is 18 and cannot be made by any other permutation, meaning a possible discrepancy between the number of possible output neurons of the training set and test set. Second, when training on permutations of longer length, an ever increasing number of output neurons would be required. Last, for measuring how well the network performs, classification poses problems which are mitigated by using regression and mean squared error loss.

Table 2. For each permutation pair, random MNIST digits are used for generating each sample. For example, for the permutation pair (0, 2), each sample that is generated uses a random digit 0 combined with a random digit 2 obtained from MNIST.

Sample 1	Sample 2	...	Sample 1000
		...	

The following sections describe the experiment itself, beginning with a description of the dataset and how it was created. Then, the neural network’s architecture is described as well as the training strategy. Last, the results of the experiment are discussed, and the paper is concluded with a discussion.

4 Dataset

The dataset used for the creation of the concatenated image data was MNIST, a 70,000 strong collection of labelled hand written digits. As per the original dataset, 60,000 digits belong to the training set and 10,000 belong to the test set. Images in the MNIST dataset are 8-bit greyscale, 28×28 pixels in size. The generated images are therefore 28×56 pixels in size as they are the concatenation of two MNIST digits placed side-by-side and stored as a single image (see Table 3, for example). Each generated image’s label is the summation of the two individual digits’ labels (see Table 1 for several examples). For each permutation, one thousand samples are generated, and the MNIST digits are chosen at random in order to create distinct samples.

4.1 Train/Test Split

As mentioned previously, for the digits 0–9, with a maximum of length of $l = 2$, there are n^l or $10^2 = 100$ possible permutation pairs. To generate the training and testing data, the permutations pairs are split into a permutation training set and a permutation test set at random, so that: $P = P_t \cup P_v$ and $P_t \cap P_v = \emptyset$. For training, 90% of the permutations were used to generate the training samples, \mathbf{X}_t and the remaining 10% were used for generating the test set samples \mathbf{X}_v , meaning $|P_t| = 90$ and $|P_v| = 10$ while $|\mathbf{X}_t| = 90,000$ and $|\mathbf{X}_v| = 10,000$ for any particular run. The experiment was performed using a 10-fold cross validation, based on the permutations pairs, and the loss was averaged across the 10 runs.

In terms of the generated samples, the generated training set images and generated test set images honoured the MNIST training set and test set split. This means that the generated training set samples are distinct from the generated test set samples both in terms of the permutation pairs and the images used to create each sample. It should be noted that the images were chosen from MNIST randomly *with replacement*, meaning images could appear twice in different permutation pairs within the training set or test set, but not between both.

Therefore, the generated data set matrix \mathbf{X} contains 100,000 samples, 10,000 for each permutation pair. This also means that \mathbf{X}_t contains the corresponding samples for the permutations P_t , and \mathbf{X}_v contains the samples for P_v . A label vector \mathbf{y} contains the labels, which are the summations of the two digits in the sample. Similarly, $\mathbf{y} = \mathbf{y}_t \cup \mathbf{y}_v$.

4.2 Network Architecture and Training Strategy

The neural network used was a multilayer convolutional neural network similar to the original LeNet5, which was first reported by [5]. However, rather than treating the problem as a classification problem, the network is trained as a regression problem. The network was evaluated using mean squared error loss and optimised with ADADELTA [10]. All experimentation was performed with Keras using TensorFlow [1] as its back-end, running under Ubuntu Linux 14.04.

A LeNet5-type network was chosen due to its association with MNIST, having been optimised and developed for this dataset, and has repeatedly been shown to work well at the general task of character recognition. As the inputs to this network are similar to the original MNIST images, having twice the width in pixels but having the same height in pixels, the only other modification that was made was with the output of the network. Instead of a 10 neuron, fully connected output layer with *Softmax*, the final layer was replaced with a single output neuron and trained as a regression problem, optimising mean squared error loss.

4.3 Data Generation

The data generation procedure algorithm is shown in Algorithm 1. During data generation, the permutations, P , are iterated over and $m = 1000$ samples are generated for each permutation. A sample consists of two random MNIST images, corresponding to the labels in the current permutation, concatenated together as one image (this is represented by the function `ConcatenateImages`). As well as this, the label vector \mathbf{y} is generated, which contains the sum of the two digit labels that make up each individual MNIST image used to create the sample. Note that in Algorithm 1 the symbol \leftarrow represents append, as is the case for the matrix \mathbf{X} and its corresponding label vector \mathbf{y} .

The procedure shown in Algorithm 1 is repeated for the train set permutation pairs, P_t , and the test set permutation pairs, P_v . It is important to note that when generating the data for the training set permutation images, the MNIST training set is used, and conversely when generating the test set permutation images, the MNIST test set is used. This ensures no overlap between the training set or test set in terms of the permutation pairs or the data used to generate the samples.







Algorithm 1. Data Generation

Input: permutation pairs P , MNIST images M , labels y' , size $m \leftarrow 1000$.
 Initialise $X \leftarrow []$.
 Initialise $y \leftarrow []$.
for all (p_1, p_2) **in** P **do**
 for 1 **to** m **do**
 $r^1 \leftarrow$ random index from M with label p_1
 $r^2 \leftarrow$ random index from M with label p_2
 $X \leftarrow$ ConcatenateImages(M_{r^1}, M_{r^2})
 $y \leftarrow y'_{r^1} + y'_{r^2}$
 end for
end for

5 Results

Averaged across the different training/test splits of a 10-fold cross validation of the permutation pairs, mean squared error was generally under 1.0 and averaged 0.85332, as shown in Table 7. Tables 3, 4, and 5 show a number of examples of a trained network’s predictions on permutations from a test set. Table 3 shows a number of sample inputs from the test set and their predictions, as well as their true labels. It is interesting to note that the network learned to deal with permutations with images in reverse order, as is the case for (6, 4) and (4, 6) or (1, 3) and (3, 1) in Table 4. In some cases, three different permutation pairs exist in the test set which sum to the same number, and these were also predicted correctly, as seen in Table 5.

Table 3. Example results for permutation samples from the test set passed through the trained network. As can be seen, all samples use distinct MNIST digits.

Input Image	Prediction	Actual
	11.1652	11
	10.3215	10
	5.01775	5
	8.99357	9
	5.99666	6
	8.6814	9

Although the network was trained as a regression problem, accuracy can also be measured by rounding the predicted real number output to the nearest integer and comparing it to the integer label. When rounding to the nearest digit, accuracy was as high as 92%, depending on the train/test split and averaged

Table 4. Example results of correct predictions for test set permutations that have the same label but consist of different pairs of digits such as (6, 6) and (4, 8) or (9, 2) and (3, 8). Note also that the model was also able to deal with digits in swapped order, as is the case for (1, 3) and (3, 1). Table 5 shows a further example of this.










Input Image	Prediction	Actual
	11.8673	12
	11.8703	12
	10.8862	11
	10.8827	11
	3.7308	4
	4.23593	4

Table 5. Example of correct predictions for three permutations from the same test set that sum to the same value.

Input Image	Prediction	Actual
	9.84883	10
	9.9731	10
	10.0746	10

70.9%. Accuracy increases, if the predicted value is used with a floor or ceiling function, and both values compared to the true value, achieving an accuracy of approximately 88% across a 10 fold cross validation. When allowing for an error of ± 1 , the accuracy, of course, increases further. Table 6 shows the accuracy of the trained network over a 10 fold cross validation. The accuracies are measured across all samples of all test set permutations—a total of 10,000 images. The accuracies are provided here merely as a guide, for regression problems it is of course more useful to observe the average loss of the predictions versus the labels. Errors presented here are likely the result of misclassifications of the images themselves rather than the logic of the operator learned, as the network was trained to optimise the loss and not the accuracy. Also, even for folds with low accuracy there are always correct predictions for all permutation pairs, again a further reason why it is not entirely useful to report accuracies. The mean squared error loss is provided as a truer measure of the network’s performance, and in order to provide as accurate a loss as possible a 10-fold cross validation was performed. The results of a 10-fold cross validation of the permutation pairs can be seen in Table 7. The average mean squared error loss over the 10-fold cross validation was 0.853322.

Table 6. Accuracy of each run of a 10-fold cross validation. For each permutation there are always correct predictions, even for poorly performing folds, such as folds 2 and 4. By using floor and ceiling functions on the predicted values for each of the images, the accuracy increases significantly. Note that for the results here 2,000 samples per permutation were generated.

Fold	Rounding	Floor/ceiling	± 1
1	81.19%	85.51%	94.93%
2	43.00%	90.89%	96.23%
3	80.95%	86.95%	95.12%
4	55.08%	71.21%	86.56%
5	82.35%	93.56%	95.28%
6	92.23%	95.49%	96.76%
7	54.32%	86.94%	95.38%
8	74.86%	94.71%	96.67%
9	87.59%	94.33%	95.91%
10	57.48%	85.53%	96.23%
Avg.	70.91%	88.51%	94.90%

Table 7. Results of each run of a 10-fold cross validation. The average mean squared error (MSE) across 10 runs was ≈ 0.85 on the test set.

Fold	Test set MSE	Train set MSE
1	1.1072	0.0632
2	0.6936	0.0623
3	0.7734	0.0661
4	0.7845	0.0607
5	0.9561	0.0694
6	0.7732	0.0553
7	1.2150	0.0803
8	0.7278	0.0674
9	0.9464	0.0602
10	0.5556	0.0709
Avg.	0.8533	0.0656

6 Conclusion

In this work, we have presented a neural network that achieves good results at the task of addition when trained with images of side-by-side digits labelled with their summations, and tested with digit combination pairs it has never seen. The network was able to predict the summation with an average mean squared error of 0.85 for permutation pairs it was not trained with. By testing the network on

a distinct set of digit combinations that were unseen during training, it suggests the network learned the task of addition, rather than a mapping of individual images to labels. A number of further experiments would be feasible using a similar experimental setup. Most obviously, the use of three digits per image could be performed using permutations up to length 3, or higher. Furthermore, other arithmetic operations could also be tested, such as subtraction or multiplication. More generally, the applicability of this method to other datasets in other domains needs to be investigated more thoroughly, for example whether there is an analogous experiment which could be performed on a dataset that does not involve arithmetic but involves the combination and interpretation of unseen combinations of objects in order to make a classification, such as through the use of the Fashion-MNIST [9] or ImageNet datasets.

References

1. Abadi, M., et al.: TensorFlow: large-scale machine learning on heterogeneous distributed systems. [arXiv:1603.04467](https://arxiv.org/abs/1603.04467) (2016)
2. Delahunt, C.B., Kutz, J.N.: Putting a bug in ML: the moth olfactory network learns to read MNIST. *Neural Netw.* (2019)
3. Hoshen, Y., Peleg, S.: Visual learning of arithmetic operations. In: Thirtieth AAAI Conference on Artificial Intelligence, pp. 3733–3739 (2016)
4. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
5. LeCun, Y., et al.: Handwritten digit recognition: applications of neural network chips and automatic learning. *IEEE Commun. Mag.* **27**(11), 41–46 (1989)
6. Liang, Z., Li, Q., Liao, S.: Character-level convolutional networks for arithmetic operator character recognition. In: International Conference on Educational Innovation through Technology, pp. 208–212 (2016)
7. Phong, L.T., Aono, Y., Hayashi, T., Wang, L., Moriai, S.: Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Trans. Inf. Forensics Secur.* **13**(5), 1333–1345 (2018)
8. Walach, E., Wolf, L.: Learning to count with CNN boosting. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) *ECCV 2016*. LNCS, vol. 9906, pp. 660–676. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_41
9. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. [arXiv:1708.07747](https://arxiv.org/abs/1708.07747) (2017)
10. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. [arXiv:1212.5701](https://arxiv.org/abs/1212.5701) (2012)

Modelling and Reasoning



CatIO - A Framework for Model-Based Diagnosis of Cyber-Physical Systems

Edi Muškardin^(✉), Ingo Pill, and Franz Wotawa

Christian Doppler Laboratory for Quality Assurance Methodologies
for Cyber-Physical Systems Institute for Software Technology, Graz University
of Technology, Inffeldgasse 16b/II, 8010 Graz, Austria
{edi.muskardin, ipill, wotawa}@ist.tugraz.at

Abstract. Diagnosing cyber-physical systems is often a challenge due to the complex interactions between its individual cyber and physical components. With CatIO (From ‘Causarum Cognitio’, Latin for “(seek) knowledge of causes”), we propose a framework that supports a designer in developing corresponding diagnostic solutions that utilize either abductive or consistency-based diagnosis for detecting and localizing faults at runtime. Employing an interface to tools of the modeling language Modelica, a designer is able to simulate a cyber-physical system’s detailed behavior, and based on the observed data she can then assesses the diagnostic solution(s) under development and explore the trade-offs of individual solutions. For the abductive reasoning variant, CatIO supports also in coming up with the required abductive diagnosis model via an automated concept based on fault injection and the simulation of corresponding Modelica models.

Keywords: Model-based diagnosis · Modelica · Cyber-physical system · Co-simulation.

1 Introduction

For an intelligent cyber-physical system (CPS), it is crucial to be aware of its current status and also that of its environment. This allows the CPS to respond intelligently to environmental inputs and changes, and also to make the right choices when having to deal with issues like faults. Model-based diagnosis [4, 10, 18] (MBD) is a powerful means to assess an observed situation, and furthermore supports an engineer in isolating the root cause(s) of some encountered issue(s). Deploying MBD in practice is, however, usually not a trivial task, and this is especially true for CPSs [20]. That is, providing an appropriate model for diagnostic reasoning concerning a CPS—where a computation core monitors and controls an aggregation of physical and cyber components—is a quite complex and cumbersome task.

Authors are listed in alphabetical order.

© Springer Nature Switzerland AG 2020
D. Helic et al. (Eds.): ISMIS 2020, LNAI 12117, pp. 267–276, 2020.
https://doi.org/10.1007/978-3-030-59491-6_25

For consistency-based MBD [10, 13], we need a specific type of system model that describes a CPS’s expected behavior, and when considering fault modes [5] we need to cover also faulty behavior. Choosing the right abstraction level and description format for this model in order to arrive at an attractive trade-off between scalability and diagnostic preciseness is not an easy task.

For abductive diagnostic reasoning [4, 7], we need to come up with a knowledge-base (KB) describing dependencies between faults and their effects considering the system’s observable behavior. Such a KB then allows us to abductively reason backwards from experienced symptoms to possible root causes. While such a KB is usually more abstract than a detailed model of a system’s behavior, we still have to consider details like multiple faults occurring simultaneously [16]. For more information on modeling for consistency-based and abductive diagnosis we refer the interested reader to [22].

While a variety of MBD algorithms, e.g., [9, 10, 13, 18, 21], corresponding libraries [17], and also solutions for automatically generating abductive diagnosis models [15, 16] have been proposed, we are in need of frameworks that assist a designer in exploring the options available for some application scenario. In addition, several tools and languages for supporting designers in coming up with diagnostic systems have been proposed, including DiKe [6] and Rodelica [3].

With CatIO, we are proposing a framework that connects Modelica [8] with diagnostic reasoning in order to allow an engineer to simulate a system’s behavior and directly assess the diagnostic solutions she is developing in the context of such a simulation. Modelica is a flexible and intuitive programming language for modeling a system, and there is commercial as well as free tool support¹ available. One can draw furthermore on a variety of libraries, e.g., for digital circuits, fluids, or mechanics, so that Modelica is a flexible tool for modeling a CPS. Implementing the concept proposed in [15], via this Modelica interface, CatIO can support an engineer also in the development of an abductive diagnosis model based on fault injection and simulation (see Sect. 2.1).

As we will show in Sect. 2, CatIO offers a special interface for connecting an external component to a simulation. All data computed in the framework is available via this interface, and the external component is allowed to dynamically change the input values of the simulation. The motivation for this *controller* interface was to enable connecting a system’s control logic to a simulation, so that a designer can assess and verify a system’s reactions in fault scenarios.

Summing up CatIO’s features, an engineer can (i) simulate a CPS in Modelica to create data for testing a diagnostic solution under development, (ii) compare and assess MBD solutions in the context of different abstraction levels and drawing on either MBD concept, (iii) automatically generate an abductive diagnosis model, (iv) connect an external controller that uses the diagnostic information.

The remainder of our manuscript is organized as follows: After proposing the details of our framework in Sect. 2, we will show how CatIO supports designers in the context of a simple example in Sect. 3. In Sect. 4 we will draw a brief summary and will depict future work.

¹ <https://www.modelica.org/modelicalanguage>.

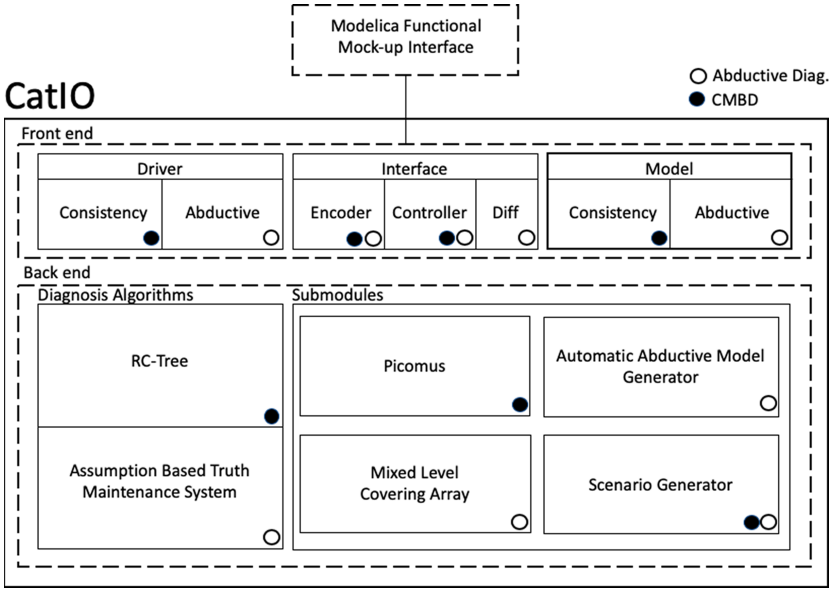


Fig. 1. CatIO’s high level architecture.

2 The CatIO Framework and Its Architecture

In Fig. 1, we illustrate CatIO’s architecture. The front end aggregates all the parts a user interacts with. This includes (i) the *driver* module for configuring the diagnostic features, (ii) the *interface* module that connects to Modelica and where we specify the *encoder* for matching the signals and translating between the simulation and the diagnostic models, the *controller* which is an external component we can connect to the simulation as described in the introduction, *diff* which we use to detect deviations between the observed and expected behavior (for the abductive variant, as will be explained later), and (iii) the *model* module which encapsulates the diagnostic model in either variant. As back end, we use a set of diagnosis algorithms complemented by a set of sub-modules.

2.1 The Back-End: Diagnosis Algorithms and Related Sub-modules

For consistency-based MBD (CMBD), we use RC-Tree [13]. RC-Tree is a variant of HS-DAG [9] and Reiter’s original algorithm [18] that avoids all the redundancy in its search for diagnoses. The computation in this MBD variant is based on isolating and resolving the conflicts between observed and expected behavior via computing the minimal hitting sets of those conflicts [18]. RC-Tree allows for an on-the-fly computation of the conflicts which is of advantage if we are just interested in diagnoses up to a certain size. While the architecture allows the integration of any solver, for our ongoing Java implementation we rely on SAT models as system description and use picomus [1] to isolate the conflicts. We

could connect also, e.g., constraint solvers though—or any other solver that can generate conflicts (ideally minimal conflicts). There is also another algorithmic concept for implementing CMBD where we compute the diagnoses directly in the solver as outlined in [11]. Our performance comparison of [12] did not show a substantial advantage in one or the other direction, so that we currently focus on RC-Tree. Depending on the diagnosis engine configuration, RC-Tree can compute both persistent faults and intermittent faults, and we can combine results from multiple simulation scenarios as was proposed in [14].

Abductive diagnosis in CatIO relies on an Assumption Based Truth Maintenance System (ATMS) [19], where we use a specific knowledge-base encoding our knowledge about a system’s fault-and-effect dependencies and then reason backwards from observed symptoms to probable causes as explanations for the symptoms [15]. Coming up with such a knowledge-base is a complex task, where CatIO helps a designer in its creation via an automation option proposed in [16]. The underlying principle there is that we test the system and faulty system versions (that we can create via fault injection) for collecting possible symptoms caused by faults and also the corresponding fault-and-effect dependency rules for aggregating the knowledge base (in CatIO, these rules use a Prolog-like syntax). In this approach, combinatorial testing (CT) allows us to explore the input space in a controlled way such that we adequately cover all possible fault/input scenario combinations. As CT engine, we use the ACTS [23] library for generating mixed level covering arrays (MCAs) from a simulation’s input variables and their domains. These MCAs then define the set of simulations conducted for generating the abductive diagnosis model [15,16].

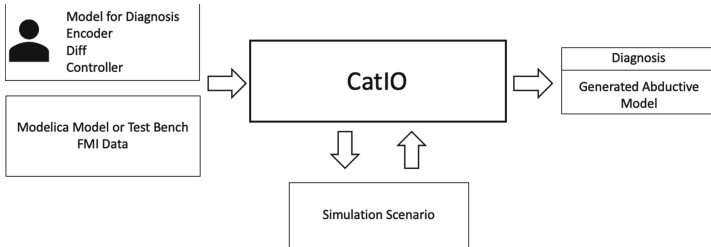


Fig. 2. Inputs and outputs of CatIO.

2.2 Front-End: Modelica Models and Simulations

In Fig. 2, we can see a user’s view of the in- and outputs of our framework. A central part is the Modelica model that allows the simulation of a system’s behavior. As usual, we can consider a simulation as a function that computes a system’s outputs based on the inputs as defined in a simulation scenario. There are three categories of such inputs, namely *parameters*, *inputs* and *mode assignment variables*. System parameters describe values that are usually constant

during a simulation, but which depend on a concrete scenario—like resistor values for an electrical circuit or the distance between a robot’s differential drive’s wheels in our example in Sect. 3. Inputs, on the other hand, are the dynamic inputs that connect a system to its environment—like the signals a robot’s differential drive needs for controlling its left and right wheel. Mode assignment variables are special in that they concern the behavior of a system component, so that we can, e.g., describe whether a resistor is *OK*, *broken*, or *shorted*.

Modelica models can be converted to Functional Mock-up Interfaces (FMIs) [2], which an external program—like our CatIO framework—can then use to conduct and control a corresponding simulation. Unless we configure variables as input variables in Modelica, we can’t change them during the simulation though. Thus, at least the inputs and mode-assignment variables have to be configured as inputs for an FMI. This allows us to support the dynamic control of a simulation (like for the robot control logic example mentioned in the introduction), and to simulate a CPS’s behavior for test cases/simulations defined in a test bench. As we will see in Sect. 3, CatIO offers a graphical user interface (see Fig. 4) for (i) an easier extraction of the necessary data from FMIs and (ii) supporting a designer in the manual creation of simulation scenarios.

When we would like to use observations from a Modelica simulation in our diagnostic reasoning, we have to employ abstraction. In particular, since we have that the simulated signal values in a simulation are expressed in Modelica data types (like Boolean, integer or float), we have to map them to Boolean variables or propositional variables as used in the CMBD diagnosis model or the abductive KB respectively. For this purpose, CatIO uses the *encoder* function that has to be provided by an engineer. For each time step, this function reads the desired signal values of a simulation and performs the checks and comparisons defined in the function in order to derive observations in the correct format.

A central part in the concept for automating the generation of an abductive diagnosis model via Modelica simulations [15,16] (see also Sect. 2.1) is the *diff* function. This function (also to be provided by the engineer) takes observations encoded with respect to the model, and identifies as well as encodes discrepancies between correct and faulty behavior. It so to say acts as deviation detector and encoder. Alternatively, *diff* can interpret values also directly from a simulation, i.e., without *encoder* mapping them to the desired format beforehand.

2.3 Front End: Further Inputs and Interfaces

CatIO’s *controller* interface allows an engineer to connect an external function to a simulation by specifying a class encapsulating the external component to be interfaced. Such a component might describe a system’s control logic that takes diagnostic information into account for coming up with repair and compensation actions, or might implement more general concepts for affecting/controlling a simulation. An example scenario for the case of a robot would be to connect its control logic, so that a designer can assess and verify the correct exploitation of diagnostic information.

For either diagnosis variant the user has to come up with the required diagnosis model/KB and the artifacts described before. Then he or she has to specify the length of a simulation and the time-step size, as well as the desired diagnosis parameters (persistent vs. intermittent faults, single or multiple scenarios, ...).

3 Example

Let us consider a robot’s differential drive as depicted in Fig. 3. For this example, we have the distance d between the wheels as parameter, the voltages u_R, u_L for the left and right wheel as system inputs, and we have mode assignment variables m_L, m_R for the wheels whose domain is $\{ok, faster, slower, stuck\}$. Let us assume that, as suggested in Sect. 2.2, all changing variables in the Modelica model are considered as inputs for the FMI (Fig. 4).

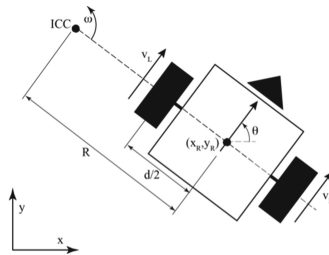


Fig. 3. Mobile robot with differential drive.

The first step now is to select those Modelica signals/variables which we’re going to consider in the diagnostic process (i.e. those relevant for the diff and encoder functions). For our example, these are the inputs and outputs for the two wheels. For our current Java implementation of CatIO, we can see the corresponding graphical user interface for this step in Fig. 4. At the same time, the user can also define an input scenario for a simulation (in the bottom half of the figure) either manually, or via defining possible variable domains (defined in the column labeled “values” in the top table in the Fig. 4) to be considered for an automated generation of scenarios via computing mixed-level-covering arrays—a concept that is used also when generating an abductive knowledge base as described in Sect. 2.1.

In the following listing, we show a part of the abductive KB that we might generate for our example and which would contain also background knowledge like the mutual exclusiveness of some propositions. In particular, it is physically impossible for a robot to move straight forward and to move along a curve to the left at the same time. Note that *nominal*, *slower* and *faster* are propositions relating to a wheel’s spinning speed with respect to the other wheel, while the robot’s motion direction is denoted by *straight*, *left_curve*, *right_curve*.

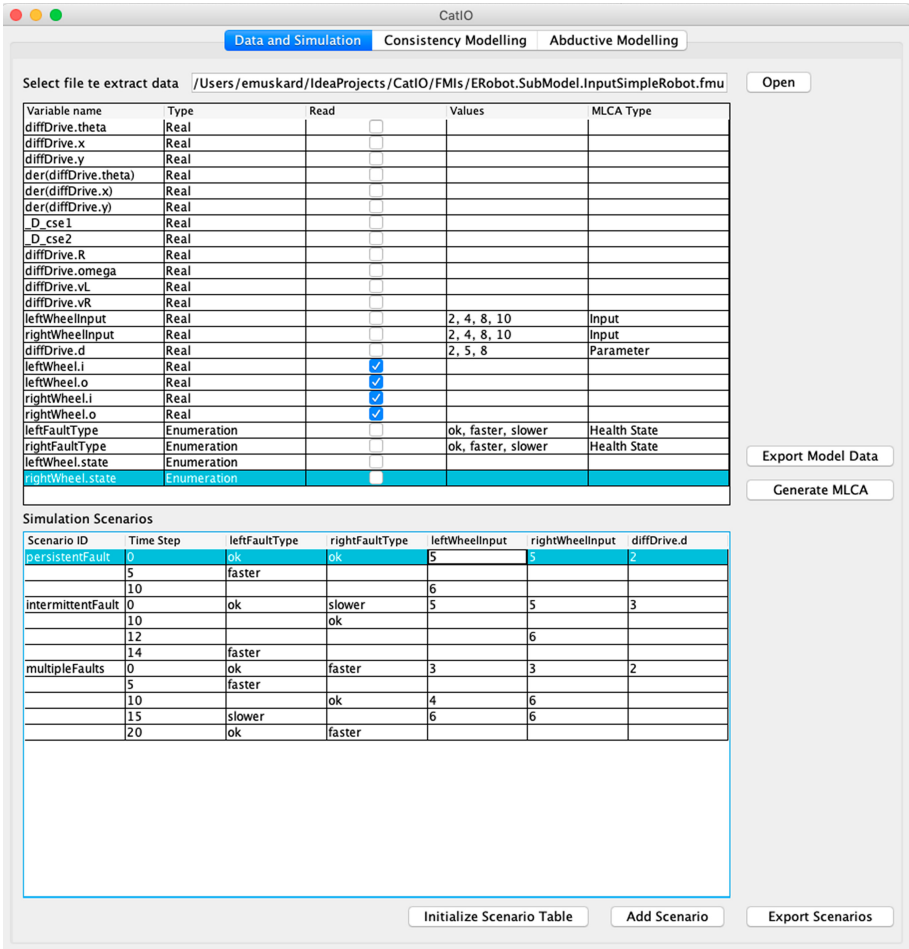


Fig. 4. Graphical user interface depicting FMI data extraction and scenario creation.

```

...
wheel(left).
wheel(right).

wheel(left), Expected(left) -> nominal(left).
wheel(left), Reduced(left) -> slower(left).
wheel(left), Increased(left) -> faster(left).
nominal(left), nominal(right) -> straight.
nominal(left), faster(right) -> left_curve.
faster(left), nominal(right) -> right_curve.
...

```

The model for CMBD would look similar, where for a manual model a designer likely will derive a component-centered model. That is, describe the components individually via arbitrary Boolean formulae first and then add the appropriate connections between the components. Please note that this Boolean description is then automatically converted to conjunctive normal form (CNF) which is the usual input format for SAT solvers.

In Fig. 4, we can see that we selected the in- and output voltages for both wheels (*leftWheel.i*, *rightWheel.i*, *leftWheel.o*, *rightWheel.o*) in the column labeled “Read” for being interfaced to the diagnostic process. For each time step, the corresponding values are passed to the *encoder* in order to map the observations to desired propositions, and in the *diff* function, we check whether the wheels are behaving as expected (such that the input signal equals the output signal).

Obviously we could now also connect the robot’s control logic via our special *controller* interface as described in Sect. 2.3. This could be a Java class that alters the input voltages for the wheels if the robot drives a curve to the left instead of going straight ahead, or where we decide to turn 270 degrees to the left instead of turning 90 degrees to the right if the latter is not possible due to some diagnosed fault. Let us assume though, that we just aim to run our diagnostic reasoning for a simulation scenario, so that we run the diagnostic process with the desired parameters—like diagnosing persistent faults for a single simulation with CMBD.

As we discussed in Sect. 2.1, CatIO offers an implementation of an approach for automatically generating abductive KBs that was proposed in [15, 16]. The basic idea there is to inject faults, simulate and compare faulty behavior against the correct one, and identify deviations in the system behavior in order to encode them in fault-and-effect dependency rules that we then aggregate in the KB. Via the *diff* function mentioned in Sect. 2.2, CatIO detects and encodes such deviations, which are then added to simple rules of the form *fault* \rightarrow *deviation*. The model automatically generated for our example contains the following rules concerning these fault-and-effect dependencies:

```
...
faster(leftWheel) -> right_curve.
faster(rightWheel) -> left_curve.
slower(leftWheel) -> left_curve.
slower(rightWheel) -> right_curve.
faster(rightWheel), slower(leftWheel) -> left_curve.
...
```

4 Summary and Future Work

In this manuscript, we propose an architecture for a framework supporting designers in the implementation of model-based diagnosis for cyber-physical system applications. CatIO’s architecture was designed with a direct interface to Modelica simulation models that allow an engineer to cover a multitude of system domains via Modelica’s versatile modeling options. The framework allows

her to explore both abductive and consistency-based MBD for offering enhanced flexibility, she can conduct corresponding experiments for assessing and comparing developed solutions, and she is supported in the creation of an abductive knowledge base via an automated generation approach. An intrinsic feature is the option to connect an external controller (e.g., a robot's control logic) that can consider all available data and can react to them via dynamically defining the future inputs to the simulation.

Currently we are in the process of implementing and testing our CatIO architecture as described in this manuscript. In future work we will add more of the mentioned MBD algorithm variants, e.g., by interfacing the PyMBD library [17] and we will add more system description formats for CMBD like constraint representations and the corresponding solvers. Also interfaces allowing CatIO to run multiple simulations simultaneously (possibly on multiple machines) for exploiting today's PC's multi-core designs would be an interesting feature for a practical application. While the architecture has already been designed with flexibility on our minds, tuning the (graphical) interfaces for the specific tasks will also be subject to future work such as to foster an intuitive deployment and efficient use in practice.

Acknowledgment. The financial support by the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development is gratefully acknowledged.

References

1. Biere, A.: Picosat essentials. *J. Satisfiability Boolean Modeling Comput.* JSAT (2008)
2. Blochwitz, T., et al.: Functional mockup interface 2.0: the standard for tool independent exchange of simulation models, 09 2012. <https://doi.org/10.3384/ecp12076173>
3. Bunus, P., Lunde, K.: Supporting model-based diagnostics with equation-based object oriented languages. In: *EOOLT. Linköping Electronic Conference Proceedings*, vol. 29, pp. 121–130. Linköping University Electronic Press (2008)
4. Console, L., Torasso, P.: Integrating models of correct behavior into abductive diagnosis. In: *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, pp. 160–166. Pitman Publishing, Stockholm, August 1990
5. De Kleer, J., Williams, B.C.: Diagnosis with behavioral modes. In: *11th International Joint Conference on Artificial Intelligence - Volume 2*, pp. 1324–1330 (1989)
6. Fleischanderl, G., Schreiner, H., Havelka, T., Stumptner, M., Wotawa, F.: DiKe - a model-based diagnosis kernel and its application. In: *Proceedings of the Joint German/Austrian Conference on Artificial Intelligence (KI)*, Vienna, Austria (2001)
7. Friedrich, G., Gottlob, G., Nejd, W.: Hypothesis classification, abductive diagnosis and therapy. In: *Gottlob, G., Nejd, W. (eds.) Expert Systems in Engineering Principles and Applications. LNCS*, vol. 462, pp. 69–78. Springer, Heidelberg (1990). https://doi.org/10.1007/3-540-53104-1_32
8. Fritzson, P.: *Principles of Object-oriented Modeling and Simulation with Modelica 3.3: A Cyber-physical Approach*. Wiley (2014)

9. Greiner, R., Smith, B.A., Wilkerson, R.W.: A correction to the algorithm in Reiter's theory of diagnosis. *Artif. Intell.* **41**(1), 79–88 (1989)
10. de Kleer, J., Williams, B.C.: Diagnosing multiple faults. *Artif. Intell.* **32**(1), 97–130 (1987)
11. Metodi, A., Stern, R., Kalech, M., Codish, M.: Compiling model-based diagnosis to Boolean satisfaction. In: *AAAI Conference on Artificial Intelligence*, pp. 793–799 (2012)
12. Nica, I., Pill, I., Quaritsch, T., Wotawa, F.: The route to success - a performance comparison of diagnosis algorithms. In: *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1039–1045 (2013)
13. Pill, I., Quaritsch, T.: RC-Tree: a variant avoiding all the redundancy in Reiter's minimal hitting set algorithm. In: *IEEE International Symposium Software Reliability Engineering Workshops (ISSREW)*, pp. 78–84 (2015). <https://doi.org/10.1109/ISSREW.2015.7392050>
14. Pill, I., Wotawa, F.: Exploiting observations from combinatorial testing for diagnostic reasoning. In: *International Workshop on Principles of Diagnosis (DX)* (2019, in press). (accepted)
15. Pill, I., Wotawa, F.: Fault detection and localization using Modelica and abductive reasoning, pp. 45–72 (2018)
16. Pill, I., Wotawa, F.: On using an I/O model for creating an abductive diagnosis model via combinatorial exploration, fault injection, and simulation. In: *29th International Workshop on Principles of Diagnosis (DX 2018)* (2018). <http://ceur-ws.org/Vol-2289/paper9.pdf>
17. Quaritsch, T., Pill, I.: PyMBD: a library of MBD algorithms and a light-weight evaluation platform. In: *International Workshop on Principles of Diagnosis (DX)*, pp. 1–5 (2014)
18. Reiter, R.: A theory of diagnosis from first principles. *Artif. Intell.* **32**(1), 57–95 (1987)
19. Reiter, R., Kleer, J.: Foundations of assumption-based truth maintenance systems: preliminary report, pp. 183–189 (1987)
20. Sayed-Mouchaweh, M. (ed.): *Diagnosability, Security and Safety of Hybrid Dynamic and Cyber-Physical Systems*. Springer, Cham (2018). <https://doi.org/10.1007/978-3-319-74962-4>
21. Wotawa, F.: A variant of Reiter's hitting-set algorithm. *Inf. Process. Lett.* **79**, 45–51 (2001)
22. Lughofer, E., Sayed-Mouchaweh, M. (eds.): *Predictive Maintenance in Dynamic Systems*. Springer, Cham (2019). <https://doi.org/10.1007/978-3-030-05645-2>
23. Yu, L., Lei, Y., Kacker, R.N., Kuhn, D.R.: Acts: a combinatorial test generation tool. In: *2013 IEEE Sixth International Conference on Software Testing, Verification and Validation*, pp. 370–375, March 2013. <https://doi.org/10.1109/ICST.2013.52>



Data Publishing: Availability of Data Under Security Policies

Juba Agoun^(✉) and Mohand-Saïd Hacid

Université de Lyon, Université Lyon 1, CNRS UMR 5205, Villeurbanne, France
{juba.agoun,mohand-said.hacid}@univ-lyon1.fr

Abstract. In this paper, we study the problem of data publishing under policy queries. We consider the privacy-aware in the context of data publishing where given a database instance, a published view, and a policy query that specifies what information is sensitive and should be protected. Our approach exploits necessary and sufficient conditions for a view to comply with a policy query. We formulate a preliminary work that consists on a data-independent method to revise the non-privacy-preserving views. With this revising process, we strike a balance between data privacy and data availability.

Keywords: Privacy-preserving · Data publishing · Data availability

1 Introduction

With the rapid growth of stored information, governments and companies are often motivated to derive valuable information. Enterprises and organizations have begun to employ advanced techniques to analyze the data and extract “*useful*” information. Data publishing has fueled significant interest in the database community [3]. It is a widespread solution to make the data available publicly and enable data sharing. It consists on exporting or publishing information of an underlying database through views to be used by peers. At the same time, the publisher tries to ensure that sensitive information will not leak.

There are great opportunities associated with data publishing, but also associated risks [16]. Indeed, it is complex to find a trade-off between preventing the inappropriate disclosure of sensitive information and guarantee the availability of non-sensitive data: Removing of all the data exported by a violating view achieves perfect privacy, but it is a total uselessness, while publishing the entire data, is at the other extreme.

In this paper, we focus on a variant problem of data availability referred also as utility of the data [13], and we specifically consider data privacy in database publishing. The owner of a given database D wishes to publish a set of views \mathcal{V}

This research is performed within the scope of the DataCert (Coq deep specification of privacy aware data integration) project that is funded by ANR (Agence Nationale de la Recherche) grant ANR-15-CE39-0009 - <http://datacert.lri.fr/>.

under a set of restrictions \mathcal{S} . We assume that the restrictions on the views are expressed as a set of queries, called policy queries [11, 12]. Inspired by prior work on privacy-preservation [18], we define a view to be *safe* w.r.t. a policy query if they don't return tuples in common, in other words, the view and the policy query are disjoint. We provide a revision algorithm for the privacy-preservation protocol to ensure data availability. Indeed, instead of neutralizing the unsafe view, we propose a rewriting technique that exploits the interaction between a view and a policy query to enrich the view with relevant information for ensuring the availability of data. Our approach exploits the concept of residue introduced in [2].

We illustrate our data publishing setting by the following running example. Let D be a database schema of a company consisting of two relations:

$$\begin{aligned} &Employee(id_emp, name, dept) \\ &PayrollService(id_emp_ps, accountNum, salary) \end{aligned}$$

The relation *Employee* stores, for each employee her/his identifier, her/his name and her/his department. The relation *PayrollService* stores information related to employees, in particular, their identifier, their account number, and their salary.

Consider the following set \mathcal{S} of policy queries. The policy queries define the information that is sensitive to make secret to public.

$$\begin{aligned} S_1(i, n) &: -Employee(i, n, d) \\ S_2(n, s) &: -Employee(i, n, d), PayrollService(i, a, s), s > 3000 \end{aligned}$$

Query S_1 projects the *identifier* and the *name* of an employee. It states that both attributes are sensitive when they are returned simultaneously. S_2 states that the *name* and the *salary* of employees with a salary over \$3 000 are sensitive if returned together.

Now, consider the following set of published views \mathcal{V} . The view V_1 projects the *name* and the *salary* of the employees in department "info501" whereas V_2 projects the *name* and the *salary* of the employees with a salary under \$10 000.

$$\begin{aligned} V_1(n, s) &: -Employee(i, n, d), PayrollService(i, a, s), d = 'info501' \\ V_2(n, s) &: -Employee(i, n, d), PayrollService(i, a, s), s < 10000 \end{aligned}$$

The policy query S_1 is not violated by any of the views in \mathcal{V} since none of them returns the *identifier* and the *name* of the employees. The view V_1 discloses some sensitive information since $S_2(I)$ and $V_1(I)$ overlap for some database instances, such as $I = \{Employee\langle 'p552', 'Jhon', 'info501' \rangle\}$, $PayrollService\langle 'p552', 'FRB1015', 4500 \rangle\}$. Regarding the view V_2 and the query S_2 , one can notice that there could be some overlapping tuples since the selection conditions in the two queries are both satisfiable for some values of s .

The questions addressed in our paper are the following: **Are the published views \mathcal{V} safe w.r.t. the policy queries \mathcal{S} ? If the views are not safe w.r.t.**

the policy queries, how could we revise them to make available the subset of tuples which are not in the set of tuples identified by \mathcal{S} ?

The paper is organized as follows. Section 2 discusses related work. Section 3 discusses the basic concepts and notions relevant to our approach. Section 4 presents the privacy preservation protocol. Section 5 describes a mechanism for revising the non-privacy-preserving views, while Sect. 6 concludes our paper.

2 Related Works

Access control [1] is considered as traditional database security. It prevents unauthorized access to the database, therefore it usually offers a control at the physical level. However, access controls are the most common solution in several systems. View-based access control is a mechanism for implementing database security [15]. It is mostly based on defining the rights of authorized subjects (e.g., users) to read or modify data. Some rewritten techniques (e.g., [7]) are used to preform access control [14] by answering queries using only information contained in authorized views. Some research works enforce the security in data publishing over XML documents through an access control using cryptography [10]. However, our approach does not deal with authorization methods, it differs since the secret data is specified at the logical level rather than at the physical level.

Perfect-Security is the main method to determine leakage of private data that could occur in a publishing database. The authors of [11] presented a query-view security model based on a probabilistic formal analysis [4]. In this approach, the authors assume that they know the probability distribution of an instance and they model it as the knowledge of an adversary. Then, it compares a prior knowledge of an adversary with the knowledge of the adversary after data has been published. Although involving probability distribution, the main results of [11] shows that it can be converted to a logical test. Our approach differs from the setting considered in [11]. Indeed, the perfect-security is based on a probabilistic distribution on the instances whereas our approach is data-independent.

Determining when two given queries are disjoint was firstly introduced in [5]. The topic of disjoint queries has found application mostly in the *irrelevant updates* domain [6].

The issue of detecting privacy violations in data publishing has been investigated in [19]. The goal of this work is to prevent inferring sensitive information in XML documents. The authors developed algorithms for finding partial documents of a published document that do not leak sensitive information. In addition to the problem of detecting privacy violation. There are other techniques to avoid privacy violations such k-anonymity [17], t-closeness [8], and l-diversity [9]. They rely on the generalization of data which aims at transforming the published data. Thus, specific attribute values are transformed into less specific attribute. Depending on the nature of the attribute, different generalization methods could be considered. For example, category value could be generalized using a hierarchy, and this could replace the category value by its

ancestor according to the hierarchy. Our approach does not prevent violation by transforming the published data. Nevertheless, the views are revised to return only non-sensitive information, thus, ensure availability of data.

3 Preliminaries

In this section, we introduce the relevant concepts to our framework. We consider a relational setting. A *database schema* D consists of a finite set of relation schemas R_1, \dots, R_n , where each relational schema (or just *relation*) consists of a unique name and a finite set of attributes. The set of attributes in a relational schema R_i is denoted by $att(R_i)$. A *tuple* for a relational schema R_i , where $att(R_i) = \{X_1, \dots, X_k\}$ is an element consisting of a set of k constants. A relation r defined over a relational schema R_i , denoted by $r(R_i)$ (or simply by r_i if R_i is understood) consists of a finite set of tuples defined over R_i . A database instance defined over a schema $D = R_1, \dots, R_n$, denoted by $I(D)$ (or simply by I if D is understood) is a finite set of relations $\{r_1(R_1), \dots, r_n(R_n)\}$. We consider nonrecursive DATALOG queries with inequalities defined as follows. For the simplicity of the presentation, we assume that Boolean queries are not allowed¹. Hence, the head of the query contains only variables and at least one variable. Despite the fact that constants do not appear in the head of the query, the essentials of our results could be extended for this case.

Definition 1. We assume a set of variable names \mathcal{N} , and the function $typ: \mathcal{N} \rightarrow att(R_1) \cup \dots \cup att(R_n)$, where $D = \{R_1, \dots, R_n\}$. A conjunctive query Q is defined by:

$$Q(\bar{X}) : -t_1, \dots, t_n, C_{n+1}, \dots, C_{n+m}$$

where t_1, \dots, t_n are terms, C_{n+1}, \dots, C_{n+m} are inequalities, and $\bar{X} \in \mathcal{N}$. A term is of the form $R(X_1, \dots, X_n)$, where R is a relational schema in D and $\{X_1, \dots, X_n\} \subseteq \mathcal{N}$. An inequality has one of the following form:

1. $X \odot c$, where $X \in \mathcal{N}$, c is a constant (i.e., numeric or string), $\odot \in \{=, \leq, \geq, <, >, \neq\}$.
2. $X \odot Y$, where $\{X, Y\} \subseteq \mathcal{N}$, $\odot \in \{=, \leq, \geq, <, >, \neq\}$.

$Q(\bar{X})$ is called the head of the query Q , and $t_1, \dots, t_n, C_{n+1}, \dots, C_{n+m}$ is called the body of Q . \bar{X} is called the schema of Q and is also denoted by $att(Q)$. The queries have the following restrictions:

- a. We assume that a variable can appear at most once in the head of a query.
- b. If a variable X_i appears as the i^{th} variable in a term $R(\dots, X_i, \dots)$ then $typ(X_i) = A_i$, where A_i is the i^{th} attribute of R ;
- c. $\bar{X} \neq \emptyset$.

¹ Policy queries specify tuples to keep private. Thus, a policy query with a set of variables empty in the head is useless.

We recall that the views are considered to be conjunctive queries in our framework. Condition (a) above does not prevent join and self-join queries from being defined in our framework. Condition (c) mentions clearly that Boolean queries are not allowed. We also assume that the queries are range restricted (safety of a DATALOG query), *i.e.*, every variable X in \bar{X} also appears in some term in the body of Q , or there exists a variable Y such that C_{n+1}, \dots, C_{n+m} imply that $X = Y$ and Y appears in some term in the body of Q .

Next we will introduce some notations needed to compare the schema of queries based on the types of the variables.

Definition 2. *Given queries $S(X_1, \dots, X_n)$ and $V(X'_1, \dots, X'_m)$, $att(S)$ is contained in $att(V)$, denoted by $att(S) \preceq att(V)$, if $n \leq m$ and $typ(X_i) = typ(X'_i)$ for all $i \in \{1, \dots, n\}$. The schemas $att(S)$ and $att(V)$ are equivalent, denoted by $att(S) \equiv att(V)$, if $att(S) \preceq att(V)$ and $att(V) \preceq att(S)$. The difference between $att(V)$ and $att(S)$, denoted by \setminus_{typ} is defined as:*

$$att(V) \setminus_{typ} att(S) = \{X'_i \mid \nexists X_j (typ(X'_i) = typ(X_j))\}.$$

4 Privacy Preservation

Our notion of privacy preservation is build upon the protocol introduced in [18]. Below, we formalize the notion of privacy preservation as defined in [18] and in the next section we extend it for the query revision. First, we summarize the notion of disjoint queries.

Definition 3. *If I is an instance of a database schema D and S and V are queries that have the same schema, then S and V are defined to be disjoint if for every I in D , $Q(I) \cap V(I) = \emptyset$.*

The adopted approach for defining privacy consists in specifying the information that is private by a query S , and then the notion of a user query being legal translates to the requirement of our approach that the user query must be disjoint from S for all possible database states. The privacy violation occurs only when the same tuple is returned by both the policy query and the user query, for some database instance. The basic semantic unit of information is a tuple, not an attribute value therefore the intersection is at the tuple level and not the attribute level.

The privacy preservation we want to achieve in this paper can be determined using only the structure of S and V , and so can be checked at compile time. We define privacy preservation as follows.

Definition 4. *Query V is privacy-preserving with respect to a policy query S if either:*

$$a. att(S) \setminus_{typ} att(V) \neq \emptyset; \text{ or}$$

b. $att(S) \preceq att(V)$ and S and V' are disjoint, where V' is the query defined by:
 $V'(att(S)) : -V$

Essentially, the case (a) is the situation where there are some attributes of S that are not in V , for which we do not consider to be a violation. It could be illustrated by V_1 and S_1 from our running example. As it is previously discussed, we do not consider V_1 to be a privacy violation for S_1 . In the situation of case (b) every variable in $att(S)$ has a matching variable in $att(V)$, in which case we require that the projection of V on $att(S)$ be disjoint from S .

The main result established in [18] characterizes when a published query V preserves the privacy of a secret query S . So, given S and V two DATALOG queries over a database scheme D such that \bar{C}_S and \bar{C}_V are separately satisfiable, then V preserves the privacy of S if either:

1. $att(S) \setminus_{typ} att(V) \neq \emptyset$; or
2. the set of inequalities $\pi_{att(S)}(\bar{C}_S) \cup \pi_{att(S)}(\bar{C}_V)$ is unsatisfiable.

Next, we illustrate by an example the detection of privacy violations of a published view *w.r.t.* a policy query.

Example 1. We demonstrate over a simplified version of the running example, where the set of policy queries $\mathcal{S} = \{S_2\}$ and the set of published views $\mathcal{V} = \{V_3\}$.

$$V_3(n, a, s) : -Employee(i, n, d), PayrollService(i, a, s), s \leq 5000, d = 'info501'$$

We note that $\bar{C}_{V_3} = \{s \leq 5000, d = 'info501'\}$ and $\bar{C}_{S_2} = \{s > 3000\}$. Then $\pi_{att(S_2)}(\bar{C}_{S_2}) = s > 3000$ and $\pi_{att(S_2)}(\bar{C}_{V_3}) = \{s \leq 5000\}$. We say that V_3 is not privacy-preserving *w.r.t.* policy query S_2 since $att(S_2) \setminus_{typ} att(V_3) = \emptyset$ and $\pi_{att(S_2)}(\bar{C}_{S_2}) \cup \pi_{att(S_2)}(\bar{C}_{V_3}) = \{s > 3000, s \leq 5000\}$ which is satisfiable.

5 Revising Privacy Violating Views

In the previous section we introduced the privacy preservation and described how to detect the violating views. In this Section, we present a technique for revising the views *w.r.t.* policy queries, over all database instances.

Our approach to revise the views is based on the explorations of residues resulting after the unification process. This technique is used for semantic query optimization [2]. The unification process allows to capture residues from policy queries and will be associated with published views.

Revising the views can be described informally as the process of transforming the non-privacy-preserving view to comply with the requirements of the policy query. When the schema of the view and the query is the same (*see* Definition 4), this means that bodies are unsatisfiable. In this case, we propose using the partial subsumption technique, to extract the residue of a policy query and associate it

with the view. A policy query S partially subsumes a view V if a subclause² of S subsumes the body of V .

In the following, the process is going to be illustrated by an example. Consider a policy query S and a view to be published V over a database with two relations $R_1(a, b, c)$ and $R_2(a, c)$. The policy query S states that the attribute association $\{a, b\}$ is sensitive for tuples in R_1 that could be join with R_2 on the attribute a and where c is greater than 5. The view V project the attribute a' and b' for tuples in R_1 join R_2 on a' and where a' is under 4.

$$\begin{aligned} S(a, b) &: -R_1(a, b, c), R_2(a, c), c > 5 \\ V(a', b') &: -R_1(a', b', c'), R_2(a', c'), a' < 4 \end{aligned}$$

Since V is not privacy-preserving to S we proceed to the revision the view. It consists on generating the residue from the policy query S using the V using the partial subsumption technique. First, the body of V is negated and, thereby, the set $\{\neg R_1(a', b', c'), \neg R_2(a', c'), a' \geq 4\}$ is obtained.

Then, we construct a linear refutation tree (see Fig. 1) by considering the body of S as the root. At each step, an element of the negated body of V is unified using an element in the root: The first step of the refutation tree, unify $\neg R_1(a', b', c')$ and $R_1(a, b, c)$ with the following substitutions $\{a'/a, b'/b, c'/c\}$. In the next step, $\neg R_2(a', c')$ of the negated body V is unified with $R_2(a', c')$ in S according to the substitutions of the previous step.

Finally, we obtain at the bottom of the tree the residue: $c' > 5$, which could interpreted as " c' cannot be over 5". The residue is then associated to the view V :

$$V^r(a', b') : -R_1(a', b', c'), R_2(a', c'), a' < 4, \{c' > 5\}$$

According to the interpretation before, the meaning of the previous view is as follows:

$$V^r(a', b') : -R_1(a', b', c'), R_2(a', c'), a' < 4, c' \leq 5$$

Revising views depends on the type of resulting residue. For instance, if an empty set is obtained at the bottom of the refutation tree called *null residue*, we would associate to the view an empty residue $\{ \}$ which is then replaced by a *false* term. A false term in a query leads to directly exclude the view from the set of published views \mathcal{V} . We recall that we omit the case of unsatisfied residue since we consider only the non-privacy-preserving views for revision.

Above we assumed V and S having the same schema. Now, we consider the case where some elements of the schema $att(S)$ appears in $att(V)$ (i.e., $att(S) \setminus att(V) = \emptyset$ and $att(S) \preceq att(V)$). Same as previously, the residue would be computed using the body of the view V and S and added to the given view. In case the result of the refutation tree is empty, instead of incorporating the

² C is a subclause of D if every literal in C is also in D . A literal could be either a term or an inequality.

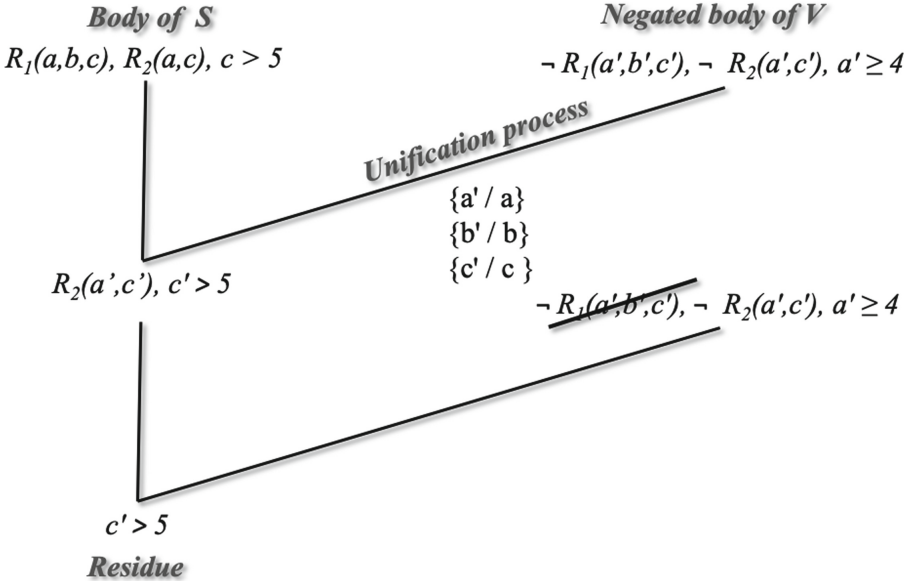


Fig. 1. Refutation tree

term *false*, which implies removing the entire view as seen before, we propose to keep the view and put the sensitive attributes as secret. Indeed, the attributes of $att(S)$ in the view V are replaced by *NULL*. Putting the sensitive attributes to *NULL* prevents the disclosure and helps to redesign the views for a better utility.

Example 2. Consider the view V_1 from our running example and the following policy query S_3 :

$$S_3(n) : \neg Employee(i, n, d), PayrollService(i, a, s)$$

The result of the refutation tree would be a *null residue* and we note that $att(S_3) \setminus att(V_1) = \emptyset$ and $att(S_3) \preceq att(V_1)$. In this case, we propose to rewrite V_1 into V^r having the following form:

$$V^r_1(NULL, s) : \neg Employee(i, n, d), PayrollService(i, a, s), d = 'info501'$$

6 Conclusion

We addressed the problem of privacy-preserving and data availability in data publishing under policy queries. We proposed a preliminary method to revise views. The method is based on the concept of residue usually adopted in semantic query optimization. Our work aims to strike the balance between data privacy and data availability. Indeed, instead of only neutralizing the non-privacy-preserving views, we proposed a data-independent process for revising the views

w.r.t. policy queries that sanitize the views from sensitive information. Our revising model returns changed views that do not guarantee the correctness since some information could be hidden. However, it provides safe access to the published data and considerable availability.

Future work will address more complex data-publishing scenarios. Firstly, we have presented the revising of one view *w.r.t.* to a policy query. We are extending our approach to accommodate data dependencies and complex interactions between views. When there are multiple published views, a violation could potentially occur from a combination of views which, individually, are privacy-preserving *w.r.t.* a set of policy queries. We could pursue our revising process by considering the interplay between multiple views. Finally, we could envision an application in other contexts (*e.g.*, data integration ...).

References

1. Bertino, E., Jajodia, S., Samarati, P.: Database security: research and practice. *Inf. Syst.* **20**(7), 537–556 (1995)
2. Chakravarthy, U.S., Grant, J., Minker, J.: Logic-based approach to semantic query optimization. *ACM Trans. Database Syst. (TODS)* **15**(2), 162–207 (1990)
3. Clifton, C., et al.: Privacy-preserving data integration and sharing. In: *Proceedings of the 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pp. 19–26. ACM (2004)
4. Dalvi, N., Suciu, D.: Management of probabilistic data: foundations and challenges. In: *Proceedings of the Twenty-Sixth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pp. 1–12 (2007)
5. Elkan, C.: A decision procedure for conjunctive query disjointness. In: *Proceedings of the Eighth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pp. 134–139 (1989)
6. Gupta, A., Mumick, I.S., et al.: Maintenance of materialized views: problems, techniques, and applications. *IEEE Data Eng. Bull.* **18**(2), 3–18 (1995)
7. Levy, A.Y., Mendelzon, A.O., Sagiv, Y.: Answering queries using views. In: *Proceedings of the Fourteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pp. 95–104 (1995)
8. Li, N., Li, T., Venkatasubramanian, S.: t -Closeness: privacy beyond k -anonymity and l -diversity. In: *2007 IEEE 23rd International Conference on Data Engineering*, pp. 106–115. IEEE (2007)
9. Machanavaajhala, A., Kifer, D., Gehrke, J., Venkatasubramanian, M.: l -diversity: privacy beyond k -anonymity. *ACM Trans. Knowl. Discov. Data (TKDD)* **1**(1), 3-es (2007)
10. Miklau, G., Suciu, D.: Controlling access to published data using cryptography. In: *Proceedings 2003 VLDB Conference*, pp. 898–909. Elsevier (2003)
11. Miklau, G., Suciu, D.: A formal analysis of information disclosure in data exchange. *J. Comput. Syst. Sci.* **73**(3), 507–534 (2007)
12. Nash, A., Deutsch, A.: Privacy in GLAV information integration. In: Schwenick, T., Suciu, D. (eds.) *ICDT 2007*. LNCS, vol. 4353, pp. 89–103. Springer, Heidelberg (2006). https://doi.org/10.1007/11965893_7
13. Rastogi, V., Suciu, D., Hong, S.: The boundary between privacy and utility in data publishing. In: *Proceedings of the 33rd International Conference on Very Large Data Bases*, pp. 531–542. Citeseer (2007)

14. Rizvi, S., Mendelzon, A., Sudarshan, S., Roy, P.: Extending query rewriting techniques for fine-grained access control. In: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, pp. 551–562 (2004)
15. Rosenthal, A., Sciore, E.: Content-based and view-based access control (2011)
16. Shay, R., Blumenthal, U., Gadepally, V., Hamlin, A., Mitchell, J.D., Cunningham, R.K.: Don't even ask: database access control through query control. *ACM SIGMOD Rec.* **47**(3), 17–22 (2019)
17. Sweeney, L.: k-anonymity: a model for protecting privacy. *Int. J. Uncertainty Fuzziness Knowl.-Based Syst.* **10**(05), 557–570 (2002)
18. Vincent, M.W., Mohania, M., Iwaihara, M.: Detecting privacy violations in database publishing using disjoint queries. In: Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology, pp. 252–262. ACM (2009)
19. Yang, X., Li, C.: Secure XML publishing without information leakage in the presence of data inference. In: Proceedings of the Thirtieth International Conference on Very Large Data Bases-Volume 30, pp. 96–107 (2004)



Matrix Factorization Based Heuristics Learning for Solving Constraint Satisfaction Problems

Seda Polat Erdeniz^(✉), Ralph Samer, and Muesluem Atas

Graz University of Technology, Inffeldgasse 16B/2, 8010 Graz, Austria
{spolater,rsamer,muatas}@ist.tugraz.at
<http://ase.ist.tugraz.at>

Abstract. In configuration systems, and especially in Constraint Satisfaction Problems (CSP), *heuristics* are widely used and commonly referred to as variable and value ordering heuristics. The main challenges of those systems are: producing high quality configuration results and performing real-time recommendations. This paper addresses both challenges in the context of CSP based configuration tasks. We propose a novel learning approach to determine transaction-specific variable and value ordering heuristics to solve configuration tasks with high quality configuration results in real-time. Our approach employs matrix factorization techniques and historical transactions (past purchases) to learn accurate variable and value ordering heuristics. Using all historical transactions, we build a sparse matrix and then apply matrix factorization to find transaction-specific variable and value ordering heuristics. Thereafter, these heuristics are used to solve the configuration task with a high prediction quality in a short runtime. A series of experiments on real-world datasets has shown that our approach outperforms existing heuristics in terms of runtime efficiency and prediction quality.

1 Introduction

Configuration systems [5] can find solutions for problems which have many variables and constraints. For example, a configuration problem can be *the customization of cars* where many hardware and software components exist and all should work together without any conflicts. Therefore, a conflict-free solution should be found. Many scheduling and configuration problems can be defined as constraint satisfaction problems (*CSPs*) [12], in which there are a set of variables, domains (a set of possible values of each variable), and a set of constraints. A solution of a CSP is a complete set of variable assignments where all constraints are satisfied [4].

Various search techniques can be applied in order to improve the performance of CSP solvers. Variable and value ordering heuristics are common intelligent search techniques which are used for solving many kinds of problems such as *configuration*, *job shop scheduling*, and *integrated circuit design* [10].

Selecting the most appropriate (the best performing) heuristics within constraint solvers is challenging due to the following reasons:

- *Runtime.* Solving a configuration task on a finite domain is an NP-complete problem with respect to the domain size and the complexity of the given constraints. The runtime performance of the configurator becomes crucial for real-time configuration tasks.
- *Accuracy.* When there are too many solutions for a given CSP, it would be very hard for a user to select a preferred solution out of a very long solutions list. A successful configuration system should be able to provide a configuration result with high probability to be *accepted/preferred by the user* as the first solution.

In this paper, we propose a novel learning approach to find variable and value ordering search heuristics for configuration systems to overcome the aforementioned two challenges. Our approach uses historical transactions which are composed of past user interactions and stored on the configuration system. Using all historical transactions, our approach builds a sparse matrix and then applies matrix factorization to learn transaction-specific variable and value ordering heuristics. Thereafter, these heuristics are used to solve the configuration task with a high prediction accuracy in a short runtime.

2 The Example Domain: An Online Bike Shop

To demonstrate our approach on a small example, we use an online personalized bike shop example on the basis of a real world knowledge base¹. This online bike shop offers a variety of bikes which are personalized according to the user requirements and product constraints of the available bike modules.

In configuration systems, a user may specify own preferences. In addition to that, the products themselves have constraints. Given these user requirements and product constraints, the configuration system is then used to determine a configuration which satisfies both the user requirements and the product constraints.

We classify user transactions into three as follows.

Complete Transaction (CT): CT represents a complete and historical transaction (i.e., a complete past purchase). A transaction of this type can later serve as valuable input for configurations in the future.

Incomplete Transaction (IT): Another possible scenario refers to the situation where a user may leave the online shop without having purchased a configuration result. Such transactions are incomplete and denoted as IT. These transactions represent stored incomplete transactions which only contain user requirements but not a complete specification of a product. Another scenario refers to situations where new product features/services are introduced to existing products for which some complete and incomplete transactions already exist.

¹ <https://www.itu.dk/research/cla/externals/clib/bike2.cp>.

With the introduction of new product features/services, all existing CTs are converted into ITs.

Active Transaction (AT): Whenever a new user starts a new configuration session, he or she will receive instant configurations while he or she is defining requirements in the online bike shop. This scenario is referred to as an active transaction. In sharp contrast to IT, the transaction is active (i.e., ongoing) and the user can receive instant configurations based on the requirements provided by him or her.

In our online bike shop example, there are five users: *Alice*, *Bob*, *Tom*, *Ray*, and *Joe* as shown in the historical transactions table (see Table 1). These users interacted with the online personalized bike shop in the past.

Table 1. Historical transactions of the online bike shop

User name:	Alice	Bob	Tom	Ray	Joe
<i>frame_biketype</i>	3	4	0		2
<i>frame_internal</i>	1	0		1	
<i>extra_propstand</i>	1	1	1		
<i>gear_internal</i>	1	0	1	1	1
Type:	CT	CT	IT	IT	IT

In the same bike shop example, Lisa is an active user who has not left the online shop yet and has defined own preferences (REQ_{Lisa}) as shown in Table 2.

Table 2. A configuration satisfaction problem: CSP_{Lisa}

V_{bike}	$frame_biketype = \{0, \dots, 4\}$, $frame_internal = \{0, \dots, 1\}$, $extra_propstand = \{0, \dots, 2\}$, $gear_internal = \{0, \dots, 1\}$
C_{bike}	$c_1 : (frame_biketype = (1 \vee 2)) \implies (frame_internal = 1)$ $c_2 : (frame_biketype = 4) \implies (gear_internal = 0)$ $c_3 : (frame_internal = gear_internal)$
REQ_{Lisa}	$c_4 : (frame_biketype = 2)$ $c_5 : (gear_internal = 1)$

3 The Proposed Method

State-of-the-art variable and value ordering heuristics lack of possibilities to consider both *runtime performance* and *prediction accuracy* for configuration

systems. The motivation of our approach is to decrease the runtime in accordance with the increase of prediction quality for configuration systems.

Our approach calculates transaction-specific variable and value ordering heuristics to solve two kind of configuration tasks which are based on: *incomplete historical transactions (IT)* and *active transactions (AT)*.

In order to find configurations for IT based configuration tasks, we generate a sparse matrix using the historical transactions and decompose it into user and value factors. Then, we obtain a dense matrix which holds similar values of the sparse matrix. In addition, it has the estimated values for the missing values in the sparse matrix. Therefore, in this dense matrix, we have predictions for the missing values of ITs. These predictions represent the probabilities of the corresponding values of ITs. Therefore, we start the configuration search with the highest probability values to find a configuration for an IT.

However, ATs are not included in these matrices since the online computation of such matrix factorization can be time taking. Thus, to find configurations for AT based configuration tasks, we use the calculated heuristics of the most similar historical transaction of AT and use them to solve the configuration task of an AT.

In our working example, M_TRX_{bike} is a sparse bitmap matrix (see Table 3-(a)) which is composed of transactions in Table 1. The bitmap matrix contains only 0s and 1s (i.e., bits). In a row (transaction), 1s mean that the variables hold as values the *corresponding* values of these columns. In return, all of the other domain values of these variables are set to 0. The blank values represent the undefined variables in the user requirements. For example, *Alice* has specified $frame_biketype=3$ which is the fourth value in its domain, so in the *Alice*'s row in the bitmap matrix (M_TRX_{bike}), the fourth bit ($b4$) is set to 1, and all the rest domain values of $frame_biketype$ (bits: $b1, b2, b3, b5$) are set to 0.

3.1 Configurations for ITs

When a user revisits the online shop, the corresponding pre-calculated configuration result is offered to him/her according to his/her incomplete historical transaction (IT). This configuration is calculated using the estimated historical transaction in the dense matrix as variable and value ordering heuristics to guide the search of configuration.

Matrix Factorization. Matrix factorization based collaborative filtering algorithms [1,6–8] introduce a rating matrix R (a.k.a., user-item matrix) which describes preferences of users for the individual items the users have rated.

We use a bitmap matrix [3,9] to hold the transactions as shown in Table 3-(a). Each row of the matrix represents a *transaction*. Each column of the matrix represents a domain value of a variable. In terms of *matrix factorization*, the sparse matrix M_TRX_{bike} is decomposed into a $m \times k$ *user-feature matrix* and a $k \times n$ *value-feature matrix* which both contain the relevant information of the sparse matrix. Thereby, k is a variable parameter which needs to be adapted accordingly depending on the internal structure of the given data.

As shown in Table 3, the dense matrix, and user-feature and value-feature matrices are calculated by applying matrix factorization based on singular-value decomposition (SVD) [2]².

Table 3. The factorization result of the sparse matrix M_TRX_{bike} . Columns represent frame_biketype, frame_internal, extra_propstand, and gear_internal.

(a) Sparse matrix M_TRX_{bike}												(b) Dense matrix $M_TRX'_{bike}$													
	0	1	2	3	4	0	1	0	1	2	0	1	0	1	2	3	4	0	1	0	1	2	0	1	
	b1	b2	b3	b4	b5	b6	b7	b8	b9	b10	b11	b12	b1	b2	b3	b4	b5	b6	b7	b8	b9	b10	b11	b12	
Alice	0	0	0	1	0	0	1	0	1	0	0	1	0.5	-0.3	0.3	0.4	0.1	0.3	1.3	-0.2	2	-0.2	-0.3	2	
Bob	0	0	0	0	1	1	0	0	1	0	1	0	0.4	-0.3	0.3	0	0.6	0.9	0.6	-0.2	1.9	-0.2	-0.3	2	
Tom	1	0	0	0	0			0	1	0	0	1	0.5	-0.3	0.2	0.3	0.3	0.5	1.1	-0.3	2.1	-0.3	-0.4	2.1	
Ray						0	1					0	1	0.6	-0.5	0.5	0.5	0.1	0.5	1.7	-0.4	2.7	-0.4	-0.5	2.9
Joe	0	0	1	0	0							0	1	0.3	-0.4	0.6	0.2	0.2	0.6	1.1	-0.3	2.1	-0.3	-0.4	2.3

Calculating the Heuristics. Using the matrix factorization outputs (user factors and value factors), we find variable and value ordering for searching configurations for an IT. To find heuristics for CT and ITs, we use the dense matrix (see Table 3-(b)) whereas to find heuristics for ATs, we use the value factors matrix.

For example, for *Ray* we sort the probabilities of the dense matrix $M_TRX'_{Ray}$. We take the first bit in the sorted dense matrix which is *b12*. It represents the assignment $gear_internal = 1$. Therefore, we set $gear_internal$ as the first variable to assign in the variable ordering and the value ordering of $gear_internal$ starts with 1 as shown in Table 4-(b). Then, we take the next bit in the dense matrix which is *b9*. It represents the assignment $extra_propstand = 1$. Thus, we set $extra_propstand$ as the second variable in the variable ordering and its value ordering starts with the value 1. Then, we complete the variable and value ordering heuristics for all CTs and ITs as shown in Table 5.

Table 4. Learning heuristics for CSP_{Ray} from the dense matrix. Columns represent frame_biketype, frame_internal, extra_propstand, and gear_internal.

(a) Dense matrix of $M_TRX'_{Ray}$												(b) Variable and Value Ordering for CSP_{Ray}											
0	1	2	3	4	0	1	0	1	2	0	1	1	0	1	0	2	1	0	0	2	3	4	1
b1	b2	b3	b4	b5	b6	b7	b8	b9	b10	b11	b12	b12	b11	b9	b8	b10	b7	b6	b1	b3	b4	b5	b2
0.6	-0.5	0.5	0.5	0.1	0.5	1.7	-0.4	2.7	-0.4	-0.5	2.9												

² We set the parameters in SVD as $latent_factors = 6$ and $iterations = 1000$.

Searching for a Configuration. After calculating the variable and value ordering heuristics by sorting the values in the dense matrix, we solve the configuration tasks of ITs using these heuristics. We provide the variable and value ordering heuristics to the CSP solver as an input. Then the solver searches for a consistent configuration based on the orders in the given heuristics.

We store the calculated heuristics and configuration results for historical transactions as shown in Table 5. We use these configuration results directly. For example, when *Ray* revisits the online bike shop, the pre-calculated personalized bike configuration (see the column *CONF* in Table 5) is directly provided without any online calculations. The heuristics are used to guide the search for configurations for similar ATs. That’s why, we also calculate the heuristics for CT even though they do not need heuristics since they already consist a complete personalized bike settings.

Table 5. Calculated heuristics and configurations

Type	User	Variable Ordering	Value Ordering					<i>CONF</i>
CT	Alice	1. extra_propstand	1	0				
		2. gear_internal	1	0	2			
		3. frame_internal	1	0				
		4. frame_biketype	0	3	2	4	1	
CT	Bob	1. gear_internal	1	0				
		2. extra_propstand	1	0	2			
		3. frame_internal	0	1				
		4. frame_biketype	4	0	2	3	1	
IT	Tom	1. extra_propstand	1	0	2			1
		2. gear_internal	1	0				1
		3. frame_internal	1	0				1
		4. frame_biketype	0	3	4	2	1	0
IT	Ray	1. gear_internal	1	0				1
		2. extra_propstand	1	0	2			1
		3. frame_internal	1	0				1
		4. frame_biketype	0	2	3	4	1	0
IT	Joe	1. gear_internal	1	0				1
		2. extra_propstand	1	0	2			1
		3. frame_internal	1	0				1
		4. frame_biketype	2	0	3	4	1	2

3.2 Configurations for ATs

In order to calculate a variable and value ordering heuristics for ATs, we cannot use the dense matrix since ATs do not yet exist in the transaction matrix. If we

would include an AT in the dense matrix and again decompose it, the runtime would not be feasible for a real-time configuration task. Therefore, for an AT, we use the information of k most similar ITs and CTs from the dense matrix of transactions to calculate heuristics for ATs.

Calculating the Heuristics. We use Euclidean Distance [13] based similarity to find the most similar historical transactions to an AT as shown in Formula 1. $M_TRX'_{HT}$ is a historical transaction (CT or IT) from the dense matrix and $M_TRX'_{AT}$ is the bit-mapped requirements of the active user. b_i represents the i th bit of the bitmap matrix. The bits of $M_TRX'_{AT}$ and $M_TRX'_{HT}$ are compared on the basis of the instantiated bits in the $M_TRX'_{AT}$. Therefore, i stands for the index of the instantiated bits in $M_TRX'_{AT}$.

$$\sqrt{\sum_{b_i \in AT.REQ} \|M_TRX'_{AT} \cdot b_i - M_TRX'_{HT} \cdot b_i\|^2} \tag{1}$$

For the working example, to find the most similar transactions, first we need to convert REQ_{Lisa} (see Table 2) into a bitmap form ($M_TRX'_{AT}$) as shown in Fig. 1.

$$M_TRX'_{Lisa} = [0\ 0\ 1\ 0\ 0\ _ _ _ _ _ _ _ \ 0\ 1]$$

Fig. 1. The active transaction of Lisa ($M_TRX'_{Lisa}$).

In order to find the most similar ITs and CTs to $M_TRX'_{Lisa}$, we find euclidean distance between $M_TRX'_{Lisa}$ and each row of the dense matrix $M_TRX'_{bike}$ based on the instantiated bits (the first five and the last two) in $M_TRX'_{Lisa}$. According to the calculated distances, k most similar transactions to $M_TRX'_{Lisa}$ are selected as $M_TRX'_{Alice}$ and $M_TRX'_{Bob}$, since k is set to 2 for this working example and they have the shortest distances to $M_TRX'_{Lisa}$.

Thus, during the search for a consistent configuration for *Lisa*, we use the variable and value ordering heuristics of $M_TRX'_{Lisa}$ which is calculated by multiplying the aggregated user features matrix of the most similar transaction with the item features matrix. The predicted user-feature matrix of Lisa $M_UF'_{Lisa}$ is multiplied with the value-feature matrix M_VF_{bike} in order to find the estimated transaction matrix for Lisa as shown in Table 6. Using the sorted values of the estimated transaction matrix, we obtain the variable and value ordering heuristics to solve CSP_{Lisa} .

Table 6. The estimated transaction for CSP_{Lisa}

b1	b2	b3	b4	b5	b6	b7	b8	b9	b10	b11	b12
0.4	-0.2	0.7	0.3	0	0.8	0.7	-0.1	1.7	-0.1	-0.2	1.9

4 Experiments

In this section, we first describe the experimental settings, the knowledge bases, evaluation criteria and comparative approaches. Then, we demonstrate the runtime and accuracy performance of our approach compared to the combinations of existing variable and value ordering heuristics.

Our experiments are executed on a computer with an Intel Core i5-5200U, 2.20 GHz processor, 8 GB RAM and 64 bit Windows 7 Operating System and Java Run-time Environment 1.8.0. For constraint satisfaction, we used a Java based CSP solver *choco-solver*³. For decomposing the bitmap matrix, we have used the *SVDRecommender* of Apache Mahout library [11] with a latent factor $k=100$, number of iterations =1000. We present all other parameters and test cases in Table 7. We have used real-world constraint-based knowledge bases from a *Configuration/Diagnosis Benchmark in Choco (CDBC)*⁴. The test parameters based on knowledge bases and corresponding transactions are in Table 7.

Table 7. Parameters of experimental tests

Experimental knowledge bases and datasets			
	Bike KB	PC KB	Camera KB
# of CT	80*	50*	150**
# of IT	20*	50*	50**
# of AT	50*	50*	64**
# of variables	31	45	10
# of constraints	32	42	20
# of variable ordering heuristics	10	10	10
# of value ordering heuristics	7	7	7
# of test cases for each configuration task	70	70	70
# of configuration tasks based on ATs	50	50	50
# of configuration tasks based on ITs	20	50	50

*synthetic transactions dataset,

**real world transactions dataset

We evaluate the performance of our approach based on the following two performance criteria: time (τ) and accuracy (π) which are calculated as explained in the following paragraphs.

Runtime (τ). Runtime, as one of our performance indicators, represents the time spent in calculating the configuration result (n represents the number of configuration tasks, see Formula 2).

$$\tau = \frac{1}{n} \times \sum_{i=1}^n runtime(CSP_i.solve()) \quad (2)$$

³ <http://www.choco-solver.org/>.

⁴ <https://github.com/CSPHeuristix/CDBC>.

Table 8. Performance results of variable ordering (on the left column) and value ordering (on the top row) combinations. Our proposed method is the combination of *MF_var* (MF based learned variable ordering heuristics) and *MF_val* (MF based learned value ordering heuristics).

(a) Runtime to solve AT based configuration tasks (given in ms)							
	Int-Domain-Max	Int-Domain-Min	Int-Domain-Median	Int-Domain-Middle	Int-Domain-Random	Int-Domain-Random-Bound	<i>MF_val</i>
Smallest	14.96	5.46	5.49	7.05	5.48	4.78	6.37
Largest	5.57	4.91	7.47	4.19	5.23	4.37	5.20
FirstFail	6.89	6.71	8.25	7.67	6.60	7.47	3.99
Anti- FirstFail	6.81	7.06	6.94	5.88	6.78	7.93	4.67
Occurance	4.37	4.62	3.28	3.01	3.72	3.38	3.76
Input Order	4.88	4.32	4.77	6.51	7.42	6.16	3.91
DomOver Dweg	25.50	3.83	3.99	3.91	4.17	5.16	4.01
Generalized MinDomain	4.23	3.15	4.21	3.88	4.02	3.97	4.05
Random	3.90	4.14	3.77	4.27	4.15	4.50	4.34
<i>MF_var</i>	3.45	3.77	3.55	3.24	3.26	3.35	2.36
(b) Prediction Accuracies for AT based configuration tasks							
	Int-Domain-Max	Int-Domain-Min	Int-Domain-Median	Int-Domain-Middle	Int-Domain-Random	Int-Domain-Random-Bound	<i>MF_val</i>
Smallest	0.69	0.67	0.69	0.67	0.67	0.64	0.67
Largest	0.69	0.64	0.72	0.72	0.72	0.64	0.72
FirstFail	0.72	0.64	0.69	0.61	0.67	0.64	0.67
Anti- FirstFail	0.69	0.67	0.67	0.69	0.67	0.67	0.69
Occurance	0.72	0.64	0.67	0.69	0.69	0.75	0.69
Input Order	0.72	0.64	0.67	0.69	0.67	0.67	0.69
DomOver-Dweg	0.69	0.64	0.67	0.64	0.64	0.61	0.58
Generalized-MinDomain	0.72	0.64	0.69	0.61	0.75	0.69	0.67
Random	0.67	0.61	0.67	0.72	0.72	0.75	0.69
<i>MF_var</i>	0.61	0.64	0.64	0.61	0.58	0.61	
(c) Prediction Accuracies for IT based configuration tasks							
	Int-Domain-Max	Int-Domain-Min	Int-Domain-Median	Int-Domain-Middle	Int-Domain-Random	Int-Domain-Random-Bound	<i>MF_val</i>
Smallest	0.69	0.67	0.69	0.67	0.67	0.64	0.69
Largest	0.69	0.64	0.72	0.72	0.72	0.64	0.74
FirstFail	0.72	0.64	0.69	0.61	0.67	0.64	0.69
Anti- FirstFail	0.69	0.67	0.67	0.69	0.67	0.67	0.72
Occurance	0.72	0.64	0.67	0.69	0.69	0.75	0.75
Input Order	0.72	0.64	0.67	0.69	0.67	0.67	0.72
DomOver-Dweg	0.69	0.64	0.67	0.64	0.64	0.61	0.61
Generalized-MinDomain	0.72	0.64	0.69	0.61	0.75	0.69	0.69
Random	0.67	0.61	0.67	0.72	0.72	0.75	0.72
<i>MF_var</i>	0.64	0.64	0.67	0.67	0.67	0.66	0.88

Accuracy (π). The configuration result ($CONF$) can be purchased by the user or not. If the purchased product is the same as $CONF$, then $CONF$ is considered as an accurate configuration $Accurate_CONF$, otherwise an inaccurate configuration. The prediction quality of a configuration approach is calculated by dividing the number of $Accurate_CONF$ s by the total number of solved CSP s (see Formula 3).

$$\pi = \frac{\#(Accurate_CONF)}{\#(CSP)} \quad (3)$$

Runtime performance is not critical for IT based configuration tasks since their calculations are executed when the corresponding user is offline. Therefore, we have not included its runtime performance into evaluations. For the real-time (AT-based) configuration tasks, we have not used matrix factorization to decrease the runtime. Instead of that we have selected the most similar transactions from the dense matrix of historical transactions. By this way, our approach outperforms the compared heuristics combinations in terms of runtime performance as shown in Table 8-(a). On the other hand, our approach outperforms the compared variable and value ordering heuristics for both IT and AT based configuration tasks in terms of prediction accuracy as shown in Table 8-(b) and Table 8-(c). IT based configuration tasks are solved with better prediction accuracy results rather than AT based configuration tasks. This is because, we have used matrix factorization based heuristics estimation for IT-based configuration tasks since the trade-off factor runtime is not critical in these offline calculations.

5 Conclusion

The quick generation of configurations within a reasonable time frame (i.e., before users leave the web page) can be very challenging especially for configuration results in a high prediction accuracy.

In this paper, we have proposed a novel learning approach for variable and value-ordering to guide the configuration search to address the runtime and prediction accuracy challenges. According to our experimental results on the basis of real-world configuration benchmarks, our approach outperforms the compared combinations of variable and value ordering heuristics in terms of *runtime efficiency* and *prediction quality* for AT based (real-time) configuration tasks. Moreover, for also IT based (offline) configuration tasks, our approach outperforms the compared approaches in terms of *prediction quality*.

References

1. Baltrunas, L., Ludwig, B., Ricci, F.: Matrix factorization techniques for context aware recommendation. In: Proceedings of the Fifth ACM Conference on Recommender Systems, pp. 301–304. ACM (2011)
2. Boutsidis, C., Gallopoulos, E.: Svd based initialization: a head start for nonnegative matrix factorization. *Pattern Recogn.* **41**(4), 1350–1362 (2008)

3. Chan, C.Y., Ioannidis, Y.E.: Bitmap index design and evaluation. In: ACM SIGMOD Record, vol. 27, pp. 355–366. ACM (1998)
4. Eiben, A., Ruttkay, Z.: Constraint satisfaction problems (1997)
5. Felfernig, A., Hotz, L., Bagley, C., Tiihonen, J.: Knowledge-Based Configuration: From Research to Business Cases, 1st edn. Morgan Kaufmann Publishers Inc., San Francisco (2014)
6. Gemulla, R., Nijkamp, E., Haas, P.J., Sismanis, Y.: Large-scale matrix factorization with distributed stochastic gradient descent. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 69–77. ACM (2011)
7. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. *Computer* **42**(8), 30–37 (2009)
8. Mnih, A., Salakhutdinov, R.R.: Probabilistic matrix factorization. In: Advances in Neural Information Processing Systems, pp. 1257–1264 (2008)
9. Papagelis, M., Plexousakis, D.: Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents. *Eng. Appl. Artif. Intell.* **18**(7), 781–789 (2005)
10. Sadeh, N., Fox, M.S.: Variable and value ordering heuristics for the job shop scheduling constraint satisfaction problem. *AI J.* **86**(1), 1–41 (1996). [https://doi.org/10.1016/0004-3702\(95\)00098-4](https://doi.org/10.1016/0004-3702(95)00098-4)
11. Schelter, S., Owen, S.: Collaborative filtering with apache mahout. In: Proceedings of ACM RecSys Challenge (2012)
12. Tsang, E.: Foundations of Constraint Satisfaction. Academic Press, Cambridge (1993)
13. Weinberger, K.Q., Blitzer, J., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. In: Advances in Neural Information Processing Systems, pp. 1473–1480 (2006)



Explaining Object Motion Using Answer Set Programming

Franz Wotawa^(✉)  and Lorenz Klampfl

Christian Doppler Laboratory for Quality Assurance Methodologies
for Cyber-Physical Systems, Institute for Software Technology,
Graz University of Technology, Inffeldgasse 16b/2, 8010 Graz, Austria
{wotawa, lklampfl}@ist.tugraz.at

Abstract. Identifying objects and their motion from sequences of digital images is of growing interest due to the increasing application of autonomous mobile systems like autonomous cars or mobile robots. Although the reliability of object recognition has been increased significantly, there are often cases arising where objects are not classified correctly. There might be objects detected in almost all images of a sequence but not all. Hence, there is a need for finding such situations and taking appropriate measures to improve the overall detection performance. In this paper, we contribute to this research direction and discuss the use of logic for identifying motion in sequences of images that can be used for this purpose. In particular, we introduce the application of diagnosis and show an implementation using answer set programming.

Keywords: Spatial reasoning · Qualitative reasoning · Diagnosis · Application of answer set programming

1 Introduction

With scientific advancements already achieved in artificial intelligence there is also a growing trend of using the resulting methods and techniques in applications. In the automotive industry more and more automated and autonomous functions have been made available for customers including automated emergency braking or lane assist. All these functions depend on certain sensors like cameras, laser scanners, or radar and have to interpret the sensor data to finally identify scenarios where a function has to be executed, i.e., reducing speed because of a braking car in front. When gaining more and more autonomy such cars must take actions under tight time constraints so that under no circumstances it harms people but also be reliable assuring a smooth and non-obstructive driving behavior. Hence, what we want are dependable and trustworthy systems.

In this paper, we contribute to this vision of trustworthy and dependable autonomous systems. In particular, we discuss steps towards utilizing logics for

Authors appear in reverse alphabetical order.

identifying object motion an autonomous system may have to deal with. Using logics is appealing in this context because it allows to explain decisions drawn from the underlying knowledge base. Its declarative nature allows us to state what we want and not requires us to come up with an implementation. Tracing object movements is important because it allows us to classify objects in being potential dangerous or not. The former may require additional actions to prevent the autonomous system from crashes or other not wanted interactions. Let us have a look at an example. In Fig. 1 we depict a typical view a driver has through the front window of a car (and an autonomous vehicle has as well).

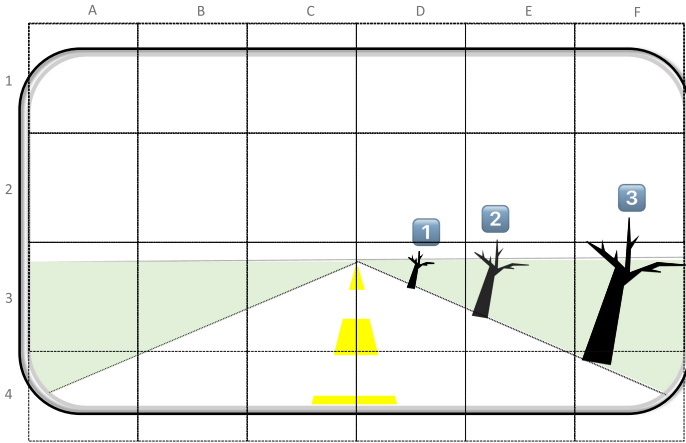


Fig. 1. View from the front window of a driving car recognizing a passing tree on the right side considering three different time steps.

In this figure, we see a tree on the right at three points in time, i.e., 1, 2, and 3. The tree is coming closer when we drive and also its visual appearance increases. However, it seems to move to the right and thus we would not expect any crash or other dangerous situation (assuming a certain uniformity of nature, where we do not consider that the tree might fall on the street when we are passing with our car). Hence, identifying movements of objects over time in our visual view is important and there are many application scenarios, e.g., (i) distinguishing driving scenarios as being critical or not, (ii) identifying spurious objects caused by wrong classifications of the used sensor, or (iii) keeping track of objects that might not be always correctly seen by the sensor. The later application may deal with a situation where the tree from Fig. 1 is seen by the sensor at time 1 and 2 but not 3. In this case, we are able to predict that the tree has to be at position F3 at time 3 because an object like a tree cannot vanish.

In the following, we focus on identifying and explaining motion of objects, i.e., going from left to right, using answer set programming (ASP) [6]. We follow ideas

outlined in Suchan et al. [12] but give attention to finding a certain movement using concepts obtained from model-based diagnosis [5, 8, 10]. In particular, we show how model-based diagnosis can be easily implemented using ASP and how we obtain a model for identifying objects' motion.

2 Basic Foundations

Diagnosis is an activity aiming at providing an explanation for certain symptoms, i.e., behaviors deviating from expectation. In artificial intelligence diagnosis has been considered early, finally leading to the field of model-based diagnosis (MBD) where the idea is to utilize models of systems directly for obtaining the diagnoses. Davis [5] was one of the first introducing the basic concepts of MBD followed by work from Reiter [10] and de Kleer and Williams [8] mainly dealing with formalizing MBD, its algorithms, and probing. The original formalization of MBD is tailored towards diagnosis of systems comprising connected components. Because we are interested in finding explanations, we are going to adapt the formalization accordingly. Instead of models we make use of a logical theory, and instead of considering components we are considering assumptions. We start the formalization stating the parts necessary to formulate a diagnosis problem comprising a theory, assumptions, and given observations, and where we are interested in obtaining those assumptions necessary and keep them in order to explain the given observations.

Definition 1 (Diagnosis Problem). *A tuple (Th, A, O) is called a diagnosis problem where Th is a model describing the system's behavior or structure, or a knowledge base, A is a set of assumptions, and O is a set of observations.*

Let us use this definition, to formulate the Nixon diamond as a diagnosis problem. We know that quakers are usually pacifists. We further assume that republicans are usually not pacifists, and we also know as facts that the former president of the U.S. Nixon was both a quaker and a republican. For assumptions like usually quakers or republicans behave as expected, we introduce the formal assumptions *nab_quaker* and *nab_republican* respectively. Hence, the corresponding diagnosis problem can be formalized as follows:

$$\begin{aligned} Th &= \left\{ \begin{array}{l} nab_quaker \wedge quaker \rightarrow pacifist, \\ nab_republican \wedge republican \rightarrow \neg pacifist \end{array} \right\} \\ A &= \{nab_quaker, nab_republican\} \\ O &= \{quaker, republican\} \end{aligned}$$

When we put together $Th \cup A \cup O$, i.e., assume that all assumptions are true, we obviously obtain an inconsistent sentence, and we further need to find out the reason behind. The following definition of diagnosis allows to characterize explanations for such unsatisfiable logical sentences formally.

Definition 2 (Diagnosis). *Given a diagnosis problem (Th, A, O) . A set $\Delta \subseteq A$ is a diagnosis if and only if the logical sentence $Th \cup \Delta \cup \{\neg a \mid a \in A \setminus \Delta\} \cup O$ is consistent. A diagnosis is minimal if no proper subset is a diagnosis.*

For the Nixon diamond, we obtain two minimal diagnosis: $\Delta_1 = \{nab_quaker\}$ and $\Delta_2 = \{nab_republican\}$. Hence, either Nixon was an ordinary quaker or an ordinary republican but not both. It is worth noting that if we know that Nixon was not a pacifist, i.e., adding $\neg pacifist$ to the set of observations O , we receive only the one minimal diagnosis Δ_1 and know that he was not behaving like an ordinary quaker.

For more details about MBD including algorithms we refer the interested reader to [14]. In the following we briefly outline how to use ASP [6] to implement MBD. Informally an answer set can be characterized as follows: First we assume a logical propositional theory comprising facts, rules, and integrity constraints. The latter is a logical rule that allows to derive a contradiction, which is used to state that the given propositions on the left side of a rule cannot be true at the same time. A logical model is a set of propositions used in the rules and integrity constraints that satisfies all of them. A logical model is an answer set if every proposition has an acyclic logical derivation from the given logical theory.

For example, given the theory of the Nixon diamond together with its observations, i.e., $Th \cup O$, we obtain the one answer set $\{quaker, republican\}$ because nothing else can be directly derived. If we add the fact nab_quaker , the answer set is $\{quaker, nab_quaker, pacifist, republican\}$. In order to allow an answer set solver to implement MBD, we need a way such that the solver can decide a truth value for nab_quaker and $nab_republican$ by itself. This can be achieved via introducing the following rules:

$$\begin{aligned} \neg nab_quaker &\rightarrow ab_quaker \\ \neg ab_quaker &\rightarrow nab_quaker \end{aligned}$$

for nab_quaker (and similar ones for $nab_republican$). These rules allow the answer set solver to select either nab_quaker or ab_quaker to be true, but not both.

Hence, in general we only need to come up with an answer set theory that comprises $Th \cup O$ and rules

$$\begin{aligned} \neg a &\rightarrow n_a \\ \neg n_a &\rightarrow a \end{aligned}$$

for all assumptions $a \in A$ to have an answer set solver representation of a diagnosis problem. Using this representation, we can compute diagnoses using the answer set solver directly. For our implementation we rely on the Potassco answer set solver¹, which makes use of Prolog syntax and where \neg is represented as `not`.

3 Modeling Object Motion

The Artificial Intelligence (AI) community today, has widely recognized the possibilities and significance of **qualitative physics**. Unfortunately codifying qualitative knowledge about the physical world has turned out to be surprisingly

¹ See <https://potassco.org/>.

difficult [13]. As stated in [4], a qualitative representation of the world with the goal to later on qualitatively reason about this world is not new. Many subareas, like qualitative spatial reasoning, qualitative temporal reasoning and qualitative simulation, were established over time. The idea in this paper focuses on the field of qualitative spatial reasoning (QSR) which provides a calculus that allows us to model object motion without the need of precise quantitative entities. The representation of objects in the model is done by stating spatial qualitative knowledge on which later on reasoning is applied with decision-making methods and techniques.

In AI systems like robotics, geographical information systems and medical analysis systems, spatial information plays a crucial role where spatial reasoning can be a potential application [2]. Cohn et al. [3] state that in most cases not a complete a priori quantitative knowledge but qualitative abstractions drive spatial reasoning. Furthermore they indicate that the challenge of QSR lies in providing a calculus which allows a machine to represent and reason with spatial entities but without using the traditional quantitative techniques which are for instance common in the computer graphics or computer vision communities. Since the concept of space in commonsense knowledge and spatial reasoning is quite complex, due to its multi-dimensionality, most work in this area has focused on single aspects of space such as topology, orientation and distance [11].

In the example given in the introduction, we approach the underlying problem by explaining object motion with the single aspect of topology. In topological approaches qualitative spatial reasoning in most cases describe relationships between spatial regions rather than points that are usually used in the case of approaches dealing with orientation [11]. The best known approach in the topological domain is the Region Connection Calculus (RCC) by Randell, Cui and Cohn [9] and includes the eight basic relations DisConnected (DC), Externally Connected (EC), Partially Overlapping (PO), Equal (EQ), Tangential Proper Part (TPP), Non-Tangential Proper Part (NTPP) and the inverse of the latter two TPP^{-1} and $NTPP^{-1}$ which are shown in Fig. 2. In addition to this basic relations, Bennett [1] presented a method for reasoning about spatial relationships on the basis of entailments in propositional logic and showed that reasoning on spatial relations represented in a logical representation is decidable.

In the following, we outline the use of **MBD for spatial reasoning using ASP** using the example discussed in the introduction for illustration purposes. In the `clingo` source code given after this paragraph we depict a simplified ASP model together with necessary integrity constraints. This ASP model considers only movements along the x -axis. With this simplification we can reduce our possible states to one dimension which results in three possible explanations of an object movement, namely `stable`, `toLeft` and `toRight`. `stable` states that the object in the following time step is still in the same grid region, which may indicate that the object is moving towards our direction which could result in a crash or dangerous situation. `toLeft` and `toRight` indicate that the object is moving to the left or right field in our projected grid during the transition from

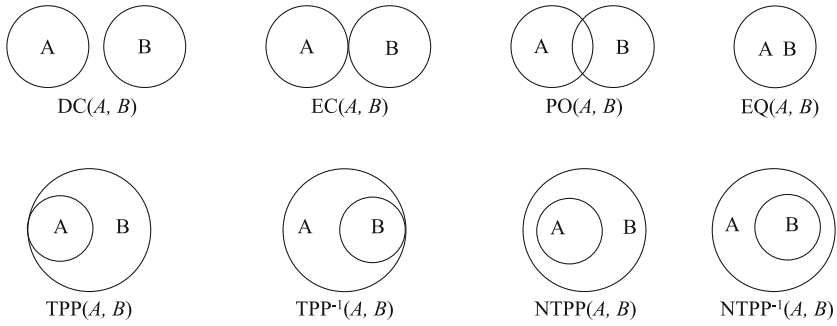


Fig. 2. A representation of the basic relations in the RCC theory by Randell et al. [9].

time step T to $T+1$, which is usually an indicator that the object is consistent in the environment and we would not expect any crash or other dangerous situation.

In source code line 3-10 the above mentioned explanations for a movement on the x -axis are defined. In addition the predicate `stable` is satisfied if the predicate `n_stable` (not stable) is not satisfied and vice versa. Similar this is done for `toRight` and `toLeft`. Further on the predicate `no_expl(N)` in line 14 counts the number of explanations for given observations and with its following integrity constraint in line 16 it is ensured that we only search for a single explanation. Line 20-28 define the basic definitions which are used if an object of interest is in the `next` or `prev` field of our projected grid. The knowledge for stating that an actual position change from time step T to time step $T+1$ based on the defined action was performed is implemented in line 33-35.

```

1  % defining explanations
2
3  stable :- not n_stable.
4  n_stable :- not stable.
5
6  toRight :- not n_toRight.
7  n_toRight :- not toRight.
8
9  toLeft :- not n_toLeft.
10 n_toLeft :- not toLeft.
11
12 % counting the number of explanations using predicate
    no_expl
13
14 no_expl(N) :- N = #count {1:stable; 2:toRight; 3:toLeft}.
15
16 :- not no_expl(1). % Search for single explanations only
17
18 % basic definition of next and previous operator
19
    
```



```

20 next(outofsight,a).
21 next(a,b).
22 next(b,c).
23 next(c,d).
24 next(d,e).
25 next(e,f).
26 next(f,outofsight).
27
28 prev(X,Y) :- next(Y,X).
29
30 % knowledge base stating that an object position change
31 % from time step 1 to 2 based on the action performed.
32
33 object(Xn,2) :- object(X,1), toRight, next(X,Xn).
34 object(Xp,2) :- object(X,1), toLeft, prev(X,Xp).
35 object(X,2) :- object(X,1), stable.

```

KB for explaining motion

In addition, we require an integrity constraint to assure that an object cannot be at two different positions at the same time step T .

```

36 :- object(X,T), object(Y,T), not X = Y.

```

Used integrity constraint

The described ASP model allows for capturing the case of identifying an object moving along the x -axis. For example, we might call `clingo` using additional observations that an object (like the tree in Fig. 1) is at position $d3$ at time step 1 and at position $e3$ at time step 2. In order to allow `clingo` computing diagnoses, i.e., explanations for this movement, we have to add the following facts (representing the observations) to the ASP source code (where we ignore the y -axis):

```

37 object(d,1).
38 object(e,2).

```

Observations for our running example

When calling `clingo` using the described file, we obtain the following result:

```

Solving...
Answer: 1
next(outofsight,a) next(a,b) next(b,c) next(c,d) next(d,e
) next(e,f) next(f,outofsight) prev(a,outofsight)
prev(b,a) prev(c,b) prev(d,c) prev(e,d) prev(f,e)
prev(outofsight,f) object(d,1) object(e,2) toRight
n_toLeft n_stable no_expl(1)
SATISFIABLE

Models      : 1
Calls      : 1

```

```

Time           : 0.002s (Solving: 0.00s 1st Model: 0.00s
                  Unsat: 0.00s)
CPU Time      : 0.002s
    
```

Result obtained using `clingo`

From the result we see that the object is moving to the right (`toRight` is true) because there is only one valid model.

After discussing the basic spatial model, we further want to discuss its potential application in practice. In Fig. 3 we illustrate a possible **application scenario of qualitative spatial reasoning** on real-world data. With the same idea as in the example outlined in the introduction, we are now looking at three frames of real-world data extracted from the well-known KITTI raw dataset [7]. On the left side the possible output of an object detector performing car detection within each frame is depicted. For simplicity we only consider one object of interest, the parked car on the right side of the street which is correctly detected in the first two frames. Let us assume that in the third frame the object detector is not able to detect the car due to an occurring sensor error. In this case the autonomous car may decide to drive too far on the right side which could result in an accident or dangerous situation. In such situations, spatial reasoning and the methods described can serve as a supplementary system capable of performing sanity checks in addition to the object detector. With spatial reasoning and the previous observations of the object detector, we are able to predict that the parked car has to be at position E3 in frame three due to the fact that the car cannot vanish within one time step. With this prediction we are now able to revise the initial output of the detector to a correct detection of the car which furthermore resolves the possible dangerous situation, and which can also result in an improvement of the overall detection performance.



Fig. 3. *Left:* an object detector recognizing the object of interest in the first two frames but missing the object in the third frame. *Right:* revised output at frame three after the application of spatial reasoning in addition to the object detector.

For simplicity reasons only movements along the x -axis were considered. Nevertheless it should be noted that with similar rules as used for movements along the x -axis, the approach can be extended to recognize movements also along the y -axis. Therefore it is also possible to extend the approach to larger problems.

Besides improving detection performance when coupling logical reasoning with object recognition, there are other application scenarios possible. We might come up with rules evaluating traffic situations as being dangerous or not as a first application. For example, whenever there is an object that remains in the center of our view but becomes bigger as well, there is danger of a crash and appropriate actions are required. We also might handle the case of total occlusion of a detected object due to other traffic participants via introducing additional spatial rules following the approach of Suchan et al. [12].

4 Conclusion

In this paper, we introduced the use of model-based diagnosis implemented using answer set programming for spatial reasoning. In particular, we discussed logical rules allowing to identify the movement of objects considering two consecutive images in time comprising the same object but at different positions. The obtained direction of movement can be used for improving the detection performance of objects and other application scenarios. To illustrate improving object detection, we made use of an example scenario obtained from a dataset used in the domain of autonomous driving.

In order to bring the approach into practice, we intend to further work on directly coupling object recognition with logical reasoning requiring to come up with the right time steps that allow for mapping the quantitative world into an abstract qualitative representation. We want to further work on adding rules for identifying critical scenarios where we need to consider object movements that may lead to a crash situation. Combining quantitative approaches like distances with qualitative ones seems to be a promising choice for future applications. In addition, we want to carry out an experimental evaluation showing that the presented approach really improves detection considering adversarial attacks (like adding rain or other disturbances) to some of the images in the sequence.

Acknowledgement. The financial support by the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development is gratefully acknowledged.

References

1. Bennett, B.: Spatial reasoning with propositional logics. In: Doyle, J., Sandewall, E., Torasso, P. (eds.) *Principles of Knowledge Representation and Reasoning*. The Morgan Kaufmann Series in Representation and Reasoning, pp. 51–62. Morgan Kaufmann (1994). <https://doi.org/10.1016/B978-1-4832-1452-8.50102-0>, <http://www.sciencedirect.com/science/article/pii/B9781483214528501020>
2. Bennett, B.: Modal logics for qualitative spatial reasoning. *Logic J. IGPL* **4**(1), 23–45 (1996). <https://doi.org/10.1093/jigpal/4.1.23>
3. Cohn, A., Hazarika, S.: Qualitative spatial reasoning. *Fundamenta Informaticae* **46**(1–2), (2001)
4. Dague, P.: Qualitative reasoning: a survey of techniques and applications. *AI Commun.* **8**(3/4), 119–192 (1995)
5. Davis, R.: Diagnostic reasoning based on structure and behavior. *Artif. Intell.* **24**, 347–410 (1984)
6. Eiter, T., Ianni, G., Krennwallner, T.: Answer set programming: a primer. In: Tessaris, S., et al. (eds.) *Reasoning Web 2009*. LNCS, vol. 5689, pp. 40–110. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03754-2_2
7. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: the kitti dataset. *Int. J. Rob. Res. (IJRR)* **32**, 1231–1237 (2013)
8. de Kleer, J., Williams, B.C.: Diagnosing multiple faults. *Artif. Intell.* **32**(1), 97–130 (1987)
9. Randell, D., Cui, Z., Cohn, A.: A spatial logic based on regions and connection. *KR* **92**, 165–176 (1992)
10. Reiter, R.: A theory of diagnosis from first principles. *Artif. Intell.* **32**(1), 57–95 (1987)
11. Renz, J., Nebel, B.: Qualitative spatial reasoning using constraint calculi. In: Aiello, M., Pratt-Hartmann, I., Van Benthem, J. (eds.) *Handbook of Spatial Logics*, pp. 161–215. Springer, Dordrecht (2007). https://doi.org/10.1007/978-1-4020-5587-4_4
12. Suchan, J., Bhatt, M., Varadarajan, S.: Out of sight but not out of mind: an answer set programming based online abduction framework for visual sensemaking in autonomous driving. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence* (2019). <https://doi.org/10.24963/ijcai.2019/260>
13. Weld, D., de Kleer, J. (eds.): *Readings in Qualitative Reasoning about Physical Systems*. Morgan Kaufmann, Burlington (1989)
14. Wotawa, F.: Reasoning from first principles for self-adaptive and autonomous systems. In: Lughofer, E., Sayed-Mouchaweh, M. (eds.) *Predictive Maintenance in Dynamic Systems*, pp. 427–460. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-05645-2_15



The GraphBRAIN System for Knowledge Graph Management and Advanced Fruition

Stefano Ferilli¹(✉) and Domenico Redavid²

¹ University of Bari, Bari, Italy
stefano.ferilli@uniba.it

² Artificial Brain S.r.l, Bari, Italy
redavid@abrain.it

Abstract. The possibility of inter-relating different information items is crucial in the perspective of enhanced storage, handling and fruition of knowledge. GraphBRAIN is a general-purpose tool that allows to design and collaboratively populate knowledge graphs, and provides advanced solutions for their fruition, consultation and analysis. Its functionalities are also provided as Web services to other applications. A peculiarity of GraphBRAIN is its fusion of methods and tools coming from different research areas: ontologies to describe such varied knowledge, collaborative tools to collect the knowledge scattered across many people, graph databases to store the knowledge base, data mining and social network analysis tools for personalized fruition of the collected knowledge. It is currently used as the knowledge management platform in a tourism-related project.

1 Introduction

To better support the information needs of scholars, practitioners, and even non-expert end users, considering also the context of the information items they handle is of utmost importance to put them in perspective and grasp a deeper understanding thereof. This requires a step up from simple ‘information’ handling to ‘knowledge’ handling. Collecting, storing and using knowledge is not trivial. It might be scattered and spread across many people with different expertise, culture, background and perspectives. Its effective exploitation might involve complex information patterns and aggregates, that might be domain- or user-dependent. From a technological perspective, a switch from databases to knowledge bases is required, so as to enable high-level analysis and reasoning tasks. In turn, knowledge bases require suitable schemes to represent and organize the knowledge, often in the form of ontologies, but a proper formalization might be unavailable in mainstream research for some domains. Last but not least, different domains might be inter-related, and cross-fertilization among them should be enforced.

Tackling all these issues together requires a suitable infrastructure, including advanced data representation and storage facilities, and advanced tools and

algorithms for knowledge handling. The solution we propose is **GraphBRAIN**, a general-purpose system aimed at supporting all stages and tasks in the lifecycle of a knowledge base—from knowledge base design, to knowledge acquisition, to knowledge organization and management, to (personalized) knowledge fruition and delivery—, also providing an on-line tool for interactive exploitation of these functionalities¹. Its main peculiarities and innovative features include: the mix of an ontology-based approach with a graph database, so as to take advantage of both the efficiency of the latter and the flexibility and power of the former for storing and handling knowledge; the cooperation among different ontologies/domains, obtained through shared entities and relationships, and through a general ontology that interconnects all the domain-specific ones, yielding a single knowledge base in which the various domains cross-fertilize each other; the integration of data mining and social network analysis tools to provide high-level functionalities for knowledge handling and fruition aimed at finding relevant, personalized and non-trivial information; an interactive interface for collaborative knowledge enrichment and personalized consultation.

This paper is organized as follows. The next section describes the main features and interface of GraphBRAIN. Then, Sect. 3 discusses its current use and relevance, also by relating it to other works in the literature. Finally, Sect. 4 concludes the paper and outlines future work issues.

2 GraphBRAIN

This section will highlight the most relevant architectural and functional features of GraphBRAIN, and how they concur in providing its advanced support to knowledge management.

The knowledge base is implemented as a graph database, using the Neo4j [13] DBMS. Neo4j implements the Labeled Property Graphs (LPG) model, in which both nodes and arcs in the graph may have associated attribute-value maps; one or many labels (usually representing classes) may be associated to nodes (representing class instances), while each arc (representing a relationship) may be labeled with one type only. Neo4j is schema-free: the user may apply any label and/or attribute to each single node or arc. While ensuring great flexibility, this does not allow to associate a clear semantics to the graph items. To overcome this, and enable high-level reasoning on the available knowledge, GraphBRAIN imposes the use of pre-specified data schemes, expressed in the form of ontologies, so that only data that are compliant to the ontologies may be added to the graph. In this way, it brings to cooperation a database management system for efficiently handling, mining and browsing the individuals, with an ontology level that allows it to carry out formal reasoning and consistency or correctness checks on the individuals.

GraphBRAIN's administrators may build and maintain several ontologies by specifying for each the hierarchies of classes and relationships to be considered, each with its attributes and associated datatypes. The universal class is

¹ A demo of the system is available at <http://193.204.187.73:8088/GraphBRAIN/>.

implicit, and common to all ontologies. Such ontologies act as DB schemes that drive and support all functionalities: knowledge base creation and enrichment; advanced tools for searching and browsing the knowledge base; mining, analysis and knowledge extraction tools that may be used interactively by end users or provided as services to other systems for obtaining selective and personalized access to the stored knowledge. Some classes and relationships may appear in different ontologies, possibly with different attributes, in order to reflect different perspectives on them. This is another innovative feature, that allows cross-fertilization among, and knowledge reuse across, different domains: individuals of shared classes act as bridges, allowing the users of a domain to reach information coming from other domains. Instead of taking external ontologies and relying on ontology alignment techniques, which may return wrong associations, GraphBRAIN requires administrators to define their ontologies from inside the GraphBRAIN environment. When defining their ontologies, they may see the classes and relationships of existing ontologies, and may reuse them, possibly extending or refining them, or defining super- or sub-classes thereof.

In particular, GraphBRAIN provides a top-level ontology, defining very general and highly reusable concepts (e.g., **Person**, **Place**) and relationships (e.g., **Person.wasIn.Place**). It acts as a hub and plays a crucial role to interconnect the domain-specific ontologies, ensuring that a single, overall connected, knowledge graph underlies all the available domains. Particularly interesting is class **Category**, aimed at storing items from different taxonomies as individuals in the knowledge graph. This allows to handle them within the graph. Currently, it includes the WordNet lexical ontology [12] (also using class **Word** for its lexical part) and the standard part of the Dewey Decimal Classification (DDC) system [2]. So, Category and Word nodes may be linked by arcs to individuals of other classes (e.g., documents, persons, places) and used as tags to express information about them. Specifically, words may be used for lexically tagging other items (e.g., this paper, as a Document node, might be linked to ‘graph’, ‘ontology’, etc.), while categories may be used to semantically tag them (e.g., this paper might be linked to ‘Computer Science’, ‘Artificial Intelligence’, etc.) without formalizing thousands of classes in the ontologies. Category and Word nodes are also related to each other, thus enabling a form of reasoning as graph traversal and allowing to find non-trivial paths between instances of other classes.

The ontologies are saved in a proprietary XML format, purposely designed to be used as a schema for the graph database. However, the tool may also export them into standard Semantic Web formats, to make them publicly available for reuse. Currently, serialization to Ontology Web Language (OWL)² format is provided, so that they can be published and exploited for ensuring semantic access to the knowledge base and make it interoperable with other resources.

Information is fed into the knowledge base by interaction with users or by automatic knowledge extraction from documents and other kinds of resources (e.g., the Internet). The interactive interface, shown in Fig. 1, includes two form-based tabs, one for entities (Fig. 1, left) and one for relationships (Fig. 1, right),

² <http://www.w3c.org/owl>.

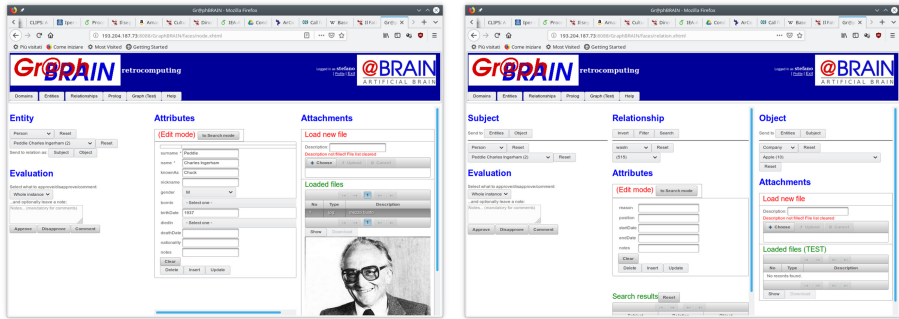


Fig. 1. GraphBRAIN interface for managing and consulting the knowledge base.

allowing the user to insert/update/remove instances and/or their attribute values. The same form-based interfaces can be used to query the knowledge base for instances of entities and relationships. The retrieved instances may be graphically displayed in another tab, as nodes and arcs in the graph. This allows the user to continue his search in a less structured way, by directly browsing the graph (e.g., expanding or compressing node neighbors). This is useful to explore the available knowledge without a pre-defined goal in mind, but letting the data themselves drive the search. Thus, serendipity in information retrieval is supported, and the users may find unexpected information that is relevant to their information needs.

The forms are automatically generated by the system from the XML format specification of the ontologies: top-level classes and relationships are shown in drop-down menus; sub-classes are shown as a tree after the top-level class has been selected; class and relationship attributes are reported in a form where names are labels and values are in textboxes, dropdown menus or other widgets depending on their type. Available types are Integer, Real, Boolean, Date, Enumeration, Instance (another instance in the knowledge base).

Albeit GraphBRAIN may handle several ontologies, each specifying a different domain, the form-based interface for data management and querying requires the user to select one of the available domains in order to load the corresponding scheme/ontology to be used. While the knowledge base content may be published as linked open data (LOD) [8], it is not available in its entirety as LOD. It is accessible only through the querying and graph browsing facilities in the on-line interface, or through pre-defined tools exposed as services, that, based on their input parameters, return relevant portions of the graph serialized as RDF.

Users may also manage (add, show, delete) attachments for each instance. In this way GraphBRAIN also acts as an archive, whose content is indirectly organized according to formal ontologies, and thus may foster interoperability with other systems. Finally, users may add comments, or approve/disapprove, each entity or relationship instance, and even each single attribute value thereof. Using the comments, the users may also provide suggestions to improve and

extend the ontologies. Through the approval/disapproval mechanism, the system may establish a trust mechanism for the users that supports ‘distributed’ quality assurance on the content of the knowledge base. Users are encouraged to provide high-quality knowledge, because using a combination of their number of contributions and trust they are assigned ‘credits’ that they may spend in using advanced features provided by GraphBRAIN. Interactions of users are tracked in order to build models of their preferences to be used for personalization purposes.

Finally, GraphBRAIN provides several analysis, mining and information extraction functionalities, including tools to:

- perform consistency checks and various kinds of reasoning on the knowledge base (currently: ontological reasoning, using OWL reasoners, by providing in OWL format and giving them access to the individuals contained in the graph in RDF; or multi-strategy (deduction, abduction, abstraction, induction, argumentation, probabilistic inference, analogy) reasoning based on an inference engine implemented in Prolog);
- assess relevance of nodes and arcs in the graph, and extract the most relevant ones (currently: Closeness centrality, Betweenness centrality, PageRank, Harmonic centrality, Katz centrality);
- extract a portion of the graph that is relevant to some specified starting nodes and/or arcs (currently: Spreading Activation and a proprietary variation thereof);
- extract frequent patterns and associated sub-graphs (currently using the Gspan algorithm, and a variation thereof that is bound to include specific nodes in the graph);
- predict possible links between nodes (currently: Resource Allocation, Common Neighbors, Adamic Adar, and a proprietary approach);
- retrieve complex patterns of nodes and relationships (currently based on the application of Prolog deduction, using user-defined or automatically learned rules, on selected portions of the knowledge graph, suitably translated into Prolog facts);
- translate selected portions of the graph into natural language (using a Prolog-based engine to organize the selected information into coherent and fluent sentences).

Some of the underlying algorithms are reused from the literature; others have been purposely extended to improve their ability to return personalized outcomes that may better satisfy the user’s information needs, which is another novelty introduced by GraphBRAIN. For instance, since the graph is too large to be entirely displayed, when opening the graph tab, a connected neighborhood of the most relevant nodes is shown. If a user model is available, based on statistics collected about his previous interaction with the system, the starting nodes may be those more related to his interests, preferences, aims, background, etc. Of course, the displayed portion of the graph may also be the result of a specific user query. For instance, Fig. 2 shows a portion of the graph automatically selected starting from 4 nodes selected by the user (indicated by arrows). Different colors of nodes are associated to different classes, and a clear predominance of some

classes (those reported as more relevant to the user by his user model) is visible. Also, several clusters of instances (some of which indicated by circles) clearly emerge, that may be analyzed by the user to find potentially interesting but possibly unexpected information.

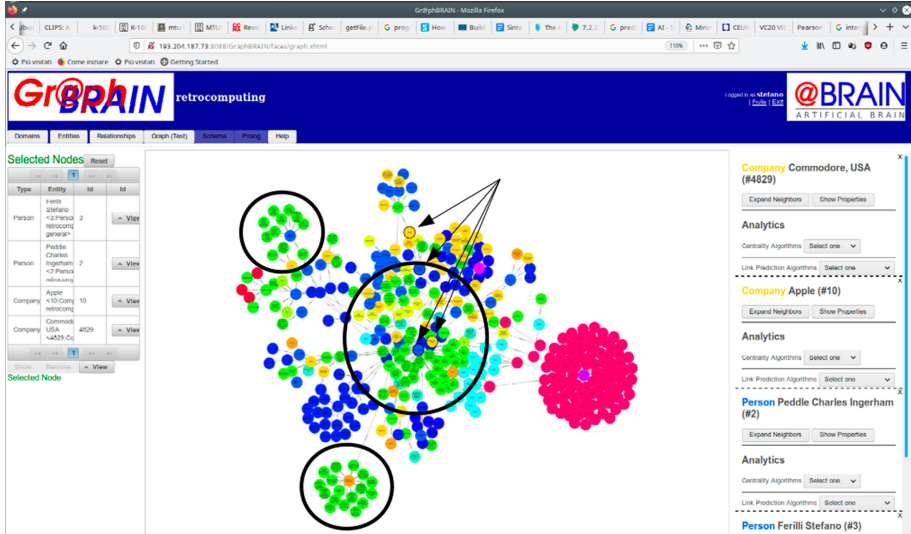


Fig. 2. Portion of GraphBRAIN's knowledge base.

3 Discussion and Related Work

We believe that GraphBRAIN is the first example of a methodology by which many different, currently separate, Artificial Intelligence tasks, techniques and approaches can be brought to cooperation for improving knowledge management and (personalized) fruition by users: database technology, ontologies, data mining, machine learning, automated reasoning, natural language processing, personalization and recommendation, social interaction. The relevance of this proposal is in these approaches being really integrated, and not simply juxtaposed, so that each of them takes directly or indirectly advantage from all the others. E.g., a social approach is used to build and integrate ontologies; user models are used to guide data mining; ontologies are used to guide database interaction and interface generation; data mining is used to filter a manageable and relevant portion of a huge graph on which carrying out automated reasoning, etc.

A prototype of GraphBRAIN is currently in use as part of a larger ongoing project [5], aimed at providing advanced support to end-users, entrepreneurs

and institutions involved in touristic activities. It currently includes the following domain-specific ontologies: **tourism** (concerning history, cultural heritage items, points of interest, logistics and services, etc.); **food** (concerning typical dishes and beverages from specific regions); **computing** (concerning computing devices and their history); **lam** (concerning libraries, archives and museums). While this paper provided a detailed account on the general structure and functionality of GraphBRAIN, [4,6] focused on the development and population of the **computing** and **lam** ontologies, respectively. Table 1 reports statistics on the current ontologies and knowledge base content of GraphBRAIN. Obviously, the vast majority of knowledge items is in the **general** subgraph, including items automatically loaded from WordNet and DDC. Next comes the **lam** subgraph, also mostly automatically loaded from the records of a private collection, including 4295 titles. Then, with much less items, come the other ontologies, whose knowledge items were manually entered using the on-line collaborative interface. There are less class instances than relationship instances, indicating a quite connected graph, which is important for interlinking the knowledge and enabling effective graph browsing by the users. The **general** subgraph is the most connected. As expected, the average number of attributes per instance is larger for class instances than for relationship instances. Indeed, relationships are by themselves information carriers. The ‘information density’ is different between classes and relationships for the various domains. Specifically, much relevant information in the **lam** domain is in the relationships rather than in the attributes, which makes sense considering the strict interplay among documents, authors, publishers, places, categories, series.

Table 1. Statistics on ontologies in GraphBRAIN

Ontology	Classes		Attributes		Relationships		Attributes	
General	17 + 27	333 020	79	1 744 116	88	488 639	23	39 186
Tourism	9 + 60	250	64	1173	49	318	13	54
Food	9 + 24	181	40	405	23	65	3	0
Computing	15 + 97	551	111	2 096	117	739	21	343
Lam	12 + 63	9 902	51	31 615	51	13 649	24	10 614
<i>Total</i>	62 + 271	343 904	345	1 779 405	328	503 410	84	50 197

Concerning the ontology development functionality, several tools have been proposed in the literature, each pursuing specific objectives as regards the construction, editing, annotation and merging of ontologies [1]. The most popular and mature one is protégé³, based on the OWL-API, which is fully compliant with the OWL specifications by W3C⁴. GraphBRAIN adopted the same OWL-API for its ontology export functionality, so that the generated ontologies are

³ <https://protege.stanford.edu>.

⁴ <http://owlcs.github.io/owlapi>.

fully compliant with the standard and may be edited using protégé. We developed a specific ontology definition and handling tool for several reasons. First, it had to be embedded into GraphBRAIN's interface, so that the administrators could seamlessly and collaboratively build and refine the ontologies. Second, while existing tools are mainly aimed at defining formal ontologies starting from an RDF knowledge base model, our motivation was in the need to define a schema for the graph DB, and the translation in standard ontology format was a consequential objective in order to enable OWL reasoning capabilities. Third, also due to the use of Neo4j as the DBMS infrastructure, our tool allows one to develop ontologies in which also relationships may have attributes, which is not allowed by current ontological standards.

Of course, the availability in GraphBRAIN of an ontology editor does not prevent the reuse of the ontologies and/or knowledge graphs already available in the literature or in the practice. Indeed, standard ontologies exist for describing many domains (e.g., the DCMI [9] for the library/archive domain, or Cultural-ON [11] and its evolution ArCo⁵ for cultural heritage). However, to the best of our knowledge, nothing exists aimed at expanding the area of interest to a broader, 'contextual' perspective that may leverage different related domains and be attractive also for non-specialized users.

On the methodological side, a few works analyze the possibilities of cooperation between ontologies and graph DBs. Some, taken from research literature, are more theoretical. [3] outlines the potential of applying graph DBMSs to an ontological context in order to create essentially an ontological tensor, and assesses its complexity. [10] discusses technical issues that might limit the impact of symbolic Knowledge Representation on the Knowledge Graph area, and summarizes some developments towards addressing them in various logics. Some other, more practical, were specifically developed for Neo4j (<https://neo4j.com/blog>). One approach consists in just expressing ontologies as nodes and arcs in Neo4j, and mapping some kinds of ontological reasoning onto graph queries. We also provide this opportunity in the **general** ontology, but additionally allow to link the ontological items to the instance nodes. Another approach is driven by an ontology in determining the graph content, but this is done 'manually' without a system module that ensures that nothing not compliant with the ontology is entered, as we do.

More in general, some NoSQL DBMSs already support the integration with ontologies. The semantic graph database GraphDB is based on the RDF graph⁶ model. So, it does not allow to represent node and relationship attributes, which causes a significant increase in nodes and relationships to represent, and thus a decrease in efficiency compared to the LPG model. Also triplestores can be considered as database management systems aimed at managing knowledge graphs. Compared to triplestores, our aim is keeping separate instance representation (owl:Individual) from representation of owl:Class, owl:ObjectProperty and owl:DatatypeProperty in the ontology, so as to be able to leverage the

⁵ <http://wit.istc.cnr.it/arco/index.php?lang=en>.

⁶ <https://db-engines.com/en/system/GraphDB%3BNeo4j>.

advantages of LPG (scalability, small size, etc.) on Individuals and the advantages of OWL Reasoners (main reasoning tasks over ontologies: consistency of the ontology, concept and role consistency, concept and role subsumption, instance checking, instance retrieval, query answering) on the ontological part. In particular, we may use a number of available Semantic Web reasoners⁷ that provide implementations for all or part of the inferences. When performing ontological reasoning that involves instances, we may still make available the LPG part as RDF thanks to the existence of a formal mapping between RDF and LPG [7], that can be leveraged (e.g., in the form of a Neo4j plugin) to keep the representation of Node and Relation properties in LPG and render instances in RDF. More in detail, the issue with RDF is that it cannot express datatype properties on relationships, and that datatype properties on classes must be also specified as triples, this way scattering the a single entity into many nodes of the graph. By representing the ontology within the graph, one needs to explicitly specify node and relationship attributes as specific relationships. In order to exploit the ontology to keep control over the data that are inserted into the graph, one must necessarily make explicit which should be considered node and attribute relationships. This amounts to representing all RDF triples as if a classical RDF store were used (some estimate that an increase in triples with respect to nodes of up to one order of magnitude might be required⁸): as a consequence, functionality that might be used on LPG graphs, such as centrality and link prediction, would at least become inefficient.

4 Conclusions and Future Work

The possibility of inter-relating different information items is crucial in the perspective of enhanced storage, handling and fruition of knowledge, that goes beyond ‘simple’ information retrieval based on lexical content or metadata. GraphBRAIN is a general-purpose tool that allows to design and collaboratively populate knowledge graphs, and provides advanced solution for their fruition, consultation and analysis. Its functionality are also provided as Web services to other applications. A peculiarity of GraphBRAIN is its strategy for cooperation of several components coming from different research areas: ontologies to describe such varied knowledge, collaborative tools to collect the knowledge scattered across many people spread all over the world, and to store it in a knowledge base, data mining and social network analysis tools for personalized fruition of the collected knowledge by interested stakeholders and end users. A prototype of GraphBRAIN is currently used as the knowledge management layer of a platform for providing advanced support to stakeholders in turistic applications.

Based on the feedback of users of the on-line prototype, we are currently extending and improving the functionality and content of GraphBRAIN. We are also developing specific analysis and mining algorithm that can leverage the

⁷ <http://owl.cs.manchester.ac.uk/tools/list-of-reasoners/>.

⁸ <https://neo4j.com/blog/rdf-triple-store-vs-labeled-property-graph-difference/>.

features and peculiarities of the system, aimed at improved fruition by end-users and client systems. Another direction for future extension is the inclusion of the possibility to find solutions via Question Answering.

References

1. Abburu, S., Babu, G.S.: Survey on ontology construction tools. *Int. J. Sci. Eng. Res.* **4**, 1748–1752 (2013)
2. Dewey, M.: A classification and subject index for cataloguing and arranging the books and pamphlets of a library, Amherst, Mass (1876)
3. Drakopoulos, G., Kanavos, A., Mylonas, P., Sioutas, S., Tsohis, D.: Towards a framework for tensor ontologies over neo4j: representations and operations. In: 8th International Conference on Information, Intelligence, Systems & Applications, IISA 2017, Larnaca, Cyprus, 27–30 August 2017, pp. 1–6. IEEE (2017)
4. Ferilli, S., Redavid, D.: An ontology and knowledge graph infrastructure for digital library knowledge representation. In: Ceci, M., Ferilli, S., Poggi, A. (eds.) *IRCDL 2020*. CCIS, vol. 1177, pp. 47–61. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-39905-4_6
5. Ferilli, S., De Carolis, B., Buono, P., Di Mauro, N., Angelastro, S., Redavid, D.: Una piattaforma intelligente per la gestione integrata del settore turistico. In: *Primo Convegno Nazionale CINI sull'Intelligenza Artificiale - Workshop on AI for Cultural Heritage*, p. 2 (2019). <http://www.ital-ia.it/submission/163/paper>. (in Italian)
6. Ferilli, S., Redavid, D.: An ontology and a collaborative knowledge base for history of computing. In: *Proceedings of the 1st International Workshop on Open Data and Ontologies for Cultural Heritage (ODOCH-2019)*, at the 31st International Conference on Advanced Information Systems Engineering (CAiSE 2016). Central Europe (CEUR) Workshop Proceedings, vol. 2375, pp. 49–60 (2019)
7. Hartig, O.: Foundations to query labeled property graphs using SPARQL. In: Kaffee, L., et al. (eds.) *Joint Proceedings of the 1st International Workshop On Semantics For Transport and the 1st International Workshop on Approaches for Making Data Interoperable Co-located with 15th Semantics Conference (SEMANTiCS 2019)*, Karlsruhe, Germany, 9 September 2019. CEUR Workshop Proceedings, vol. 2447 (2019). [CEUR-WS.org](http://ceur-ws.org)
8. Heath, T., Bizer, C.: *Linked Data: Evolving the Web into a Global Data Space*. Morgan & Claypool Publishers, Synthesis Lectures on the Semantic Web (2011)
9. ISO/TC 46/SC 4 Technical Committee: Information and documentation - the dublin core metadata element set - part 1: Core elements. Technical report, ISO 15836-1:2017 (2017)
10. Krötzsch, M.: Ontologies for knowledge graphs? In: Artale, A., Glimm, B., Kontchakov, R. (eds.) *Proceedings of the 30th International Workshop on Description Logics*, Montpellier, France, 18–21 July 2017, CEUR Workshop Proceedings, vol. 1879. CEUR-WS.org (2017). <http://ceur-ws.org/Vol-1879/invited2.pdf>
11. Lodi, G., et al.: Semantic web for cultural heritage valorisation. In: Hai-Jew, S. (ed.) *Data Analytics in Digital Humanities*. MSA, pp. 3–37. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-54499-1_1
12. Miller, G.A.: Wordnet: a lexical database for English. *Commun. ACM* **38**, 39–41 (1995)
13. Robinson, I., Webber, J., Eifrem, E.: *Graph Databases*. O'Reilly Media, 2nd edn. (2015)



Mining Exceptional Mediation Models

Florian Lemmerich^(✉), Christoph Kiefer, Benedikt Langenberg,
Jeffrey Cacho Aboukhalil, and Axel Mayer

RWTH Aachen University, Aachen, Germany
florian.lemmerich@gmail.com

Abstract. In statistics, mediation models aim to identify and explain the direct and indirect effects of an independent variable on a dependent variable. In heterogeneous data, the observed effects might vary for parts of the data. In this paper, we develop an approach for identifying interpretable data subgroups that induce exceptionally different effects in a mediation model. For that purpose, we introduce mediation models as a novel model class for the exceptional model mining framework, introduce suitable interestingness measures for several subtasks, and demonstrate the benefits of our approach on synthetic and empirical datasets.

1 Introduction

Mediation analysis is a classical statistical technique that aims for explaining the relationship between an independent variable and an (dependent) outcome variable by integrating a third, so-called mediator variable. The model suggests that the independent variable not only influences the outcome directly, but it also has an effect on the mediator variable, which in turn influences also the outcome. By doing so, it allows for distinguishing between a direct effect of the independent variable on the outcome, and an indirect effect via the mediator. Mediation analysis is a key technique predominantly used in (but not limited to) social and behavioral research, and psychology. However, in its original form, the mediation model assumes constant direct and indirect effects, i.e., it does not take heterogeneity in the data into account.

In this paper, we propose a novel approach that enables exploratory analysis of subgroups with exceptional mediation effects by integrating mediation models as a novel model class into the *exceptional model mining* framework [12]. Extending the well known subgroup discovery task [8, 11], exceptional model mining aims for identifying describable subgroups in the data with significantly different model parameters compared to the other instances in a pre-specified model of a specific class. We introduce mediation models as a novel model class for exceptional model mining, discuss multiple options, how exceptional model mining with mediation models can lead to valuable insights, and present respective interestingness measures that can identify the most interesting subgroups in the data. By doing so, our research allows for the detection of interpretable subgroups, in which the direct, indirect, or total effect in the mediation analysis is specifically strong or weak. In our experimental evaluation, we show that

our approach can successfully recover subgroups with exceptional effects in synthetic data, and demonstrate the benefits of the approach in an example on social survey data.

2 Background and Related Work

Next, we introduce the necessary background on exceptional model mining and mediation analysis, and discuss existing research most closely related to ours.

2.1 Exceptional Model Mining

In this paper, we assume a dataset D to be a multiset of data instances $i \in I$ described by a set of attributes A . In this set, we distinguish between describing attributes $A_D \subset A$ and model attributes $A_M \subset A$. In a given dataset, a *subgroup* is defined by a *subgroup description* $p : D \rightarrow \{true, false\}$ that selects instances from the dataset in a way that can be understood by humans. As it is common practice, we use in this paper conjunctions of selection conditions on single describing attributes, i.e., attribute-value pairs in the case of a nominal attribute, or intervals in the case of numeric attributes (such as $Age < 18 \wedge Gender = Male$). However, this is not a necessary condition. We call the set of all instances $c(p) = \{i \in I | p(i) = true\}$ that are described by a subgroup description p the *subgroup cover* and the set of all instances not covered by the subgroup description as its *subgroup complement*.

In exceptional model mining [12], we are now interested in finding interesting subgroups, i.e., subgroups for which the parameters of the model, which is picked a priori by the analyst, differ substantially depending on the model being fitted on the subgroup cover or subgroup complement instances. For example, consider we could choose in a population survey the simple correlation model class and the two model attributes religiosity and happiness as a model. A (fictitious) finding of exceptional model mining could then be: “*While overall there is a positive correlation between religiosity and happiness ($\rho = 0.1$), the subgroup of women younger than 18 years shows a negative correlation ($\rho = -0.15$)*”. To extract interesting subgroups from the (exponentially) large set of describable candidates, an interestingness measure (quality) q is employed that can assign a score (as a real number) to each candidate based on the statistical properties of the subgroup and the model fitted on its instances. Then, a search algorithm can identify the candidate subgroups with the highest scores. Unfortunately, solving tasks based on new model classes into exceptional model mining requires the development of tailored interestingness measures. As a key contribution, this paper accomplishes this for several use cases related to the mediation model.

2.2 Mediation Model

Mediation analysis is of interest whenever researchers assume that the effect of a putative cause X on an outcome Y of interest might be transmitted through

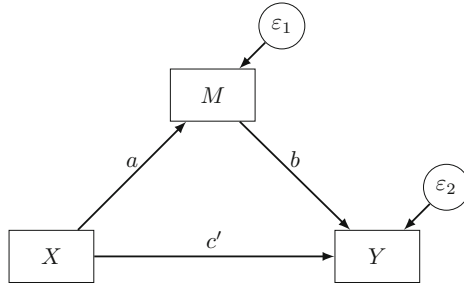


Fig. 1. Basic mediation model

an intermediate variable M [15]. A basic linear mediation analysis, cf. Baron and Kenny [1], involves three steps: First, regressing the outcome variable Y on the cause variable X to compute the total effect c of X . Second, regressing the mediator variable M to the cause variable X to obtain the first part of the indirect effect, a , i.e., the effect of X on M . Third, regressing the outcome variable Y on the cause variable X (i.e., c' , the direct effect) and the mediator variable M (i.e., b , the second part of the indirect effect). Note that in such linear models, the total effect from the first step is given by $c = c' + a \cdot b$, that is, it is decomposed into the direct effect c' and the indirect effect $a \cdot b$ (from the second and third step), see also Sect. 3. Using regression notation for mean centered variables, the basic mediation model is given by the following two equations:

$$M = a \cdot X + \varepsilon_1 \quad (1)$$

$$Y = b \cdot M + c' \cdot X + \varepsilon_2 \quad (2)$$

The two regression equations of a mediation model can be estimated simultaneously in a structural equation modeling (SEM) framework. Based on the parameters of the model, different interpretable effects can be computed as described above. These computations hold for linear models. For extensions to non-linear models and causal assumptions needed in these models, see existing work by Imai et al. [9], Pearl [16], and Mayer et al. [15].

Figure 1 shows a path diagram, which is a graphical representation of the basic mediation model. Structural equation models [2] provide a lot of flexibility to examine structural relations among a variety of variables. For example, covariates and additional mediators can be added. Or measurement models can be added for latent variables. Or the model can be estimated as a multi-group model, where the effects of a categorical variable (e.g., gender, ethnicity) can be investigated by using a distinct model for both males and females allowing for different magnitudes and directions of effects. The latter option will be important for our proposed approach to find interesting subgroups. It is also possible to set equality constraints on specific parameters among groups if needed.

2.3 Related Approaches

Exceptional model mining has been applied in a wide range of applications and for diverse model classes including classification [12], regression [4], and sequential data [14] models. In this context, Duivestijn et al. investigate Bayesian networks for exceptional model mining [5]. While these could in principle also be used to model the dependency structure of mediation, the authors focus on the difference of dependency structures derived from subgroup instances. By contrast, in this paper we study a setting, in which a specific dependency structure (a mediation) is the explicit interest of the research task.

In research on structural equation modeling, van Kesteren and Oberski [10] propose a method for detecting potential mediators from a larger search space in an exploratory way. In the presented research, we consider the mediator as fixed, and identify circumstances with extraordinary effects associated with it. Brandmaier et al. introduce structural equation model trees [3], combining the decision tree paradigm with structural equation models (which also include mediation models). These partition the dataset with respect to different parameterizations of a SEM. While individual paths in those could also be interpreted as rule-like patterns, it differs from the research presented here by the typical, well-explored differences between rule-based and decision tree-based data mining, including a different focus of local patterns vs global models, potential overlap and redundancy in the results, and different modularity of results.

While the basic mediation model assumes constant effects, there exists some extensions for investigating differential effects. In so-called *moderated mediation* analysis [17], the total, direct, and indirect effects may differ depending on categorical variables (e.g., in multigroup models), or depending on values of continuous variables, that is, the effects-of-interest can be modeled as decreasing/increasing depending on the moderator's value.

3 Mining Exceptional Mediation Models

Next, we describe how we can integrate mediation models into the exceptional model mining framework in order to find subgroups with exceptional effects.

3.1 General Approach

We assume that a mediation model with independent, dependent, and mediation variables has been specified and selected by domain experts for a more fine-grained analysis. We then identify subgroups with exceptional effects with exceptional model mining. Overall, we employ a branch-and-bound search over the space of all potential candidate subgroups. For that purpose, we start by forming selection expressions over the describing attributes in our data set and then considering all possible conjunctions over these expressions as candidates. Then, any exhaustive (e.g., search depth-first-search or apriori) or heuristic (e.g., beam search) search algorithm for subgroup discovery can be employed to enumerate and individually evaluate the candidates. The evaluation of a candidate

assigns a score from a user chosen interestingness measure. These interestingness measures are task dependent and will be discussed below. The top-K subgroups with the highest scores are finally returned as the results set.

3.2 Effects of Interest

For many other model classes that are used with exceptional model mining, parameters are symmetrical to each other. For example, in a Markov chain transition model, all parameters specify a probability of a transition from one specific state to another specific state. As a result, interestingness measures in these scenarios are treated all equally, and the exceptionality and thus interestingness of subgroups can be quantified by computing a distance measure (e.g., a Manhattan distance) with equal importance of all parameters. This is not the case for mediation models: Here, all parameters have distinct meanings and are used to determine different types of effects that are interpretable by domain specialists. We can employ exceptional model mining to unveil subgroups, in which a specific effect type (*eff*) is specifically weak or strong. These include:

- The *direct effect* quantifies, by how much the dependent variable changes per unit increase of the independent variable assuming a fixed value of the mediator. It is directly given by the value of the parameter $eff_{dir} = c'$.
- In linear models, the *indirect effect* is computed as the product of the parameters $eff_{indir} = a \cdot b$. It measures how much the dependent variable changes when the mediator increases as much as it would change for a unit increase of the independent variable.
- The *total effect* describes the unconditional dependency of the dependent on the independent variable. In linear models, it is calculated as the sum of the direct and the indirect effect: $eff_{total} = a \cdot b + c'$.

3.3 Interestingness Measures

To find subgroups that are exceptional with respect to a specific effect type, we propose two groups of interestingness measures. In both cases, we determine parameters that maximize the joint likelihood of the model in the subgroup instance and in its complement. Based on that, we compute the respective effect of interest (direct, indirect, or total effect) in the subgroup and in the complement, denoted by eff^{sg} and eff^{compl} .

For the first group of interestingness measures, which allows for a statistically guided selection of subgroups, we then conduct a Wald Test with the null hypothesis that the effect is the same in the subgroup and its complement. We then use the test statistic of this test as the score for subgroup selection, i.e., we aim to find those subgroups, in which the effect is most significantly different between the subgroup and its complement. Note that the resulting p-value of the test could in principle also be used as an alternative, but can quickly lead to arithmetic underflow in large datasets. The Wald Test directly allows to assert

the statistical significance, but the resulting p-values have to be adjusted for multiple comparison, e.g., using Bonferroni corrections.

The second group of interestingness measures is designed for a more exploratory setting. In analogy to popular measures for subgroup discovery and exceptional model mining, cf. [11, 12], we suggest to compute the score as $n^a \cdot (eff^{sg} - eff^{compl})$, where n is the number of instances covered by a subgroup and a an user chosen parameter. This parameter enables the data analyst to trade-off preferences for larger subgroups (in terms of number of instances) and stronger deviations in the effects. Due to this option, this class of interestingness measures is specifically suited for iterative and interactive mining. In its proposed form, the measures identify subgroup with a specifically strong positive effect, extension for weak or negative effects are straightforward.

3.4 Redundancy Reduction

A well-known issue in exceptional model mining is redundancy in the result set, i.e., a high similarity in the top- k found subgroups. For example, if the subgroup A induces an exceptionally strong direct effect and thus achieves a high score, then also the subgroup $A \wedge B$ with any random noise selector B can be expected to also imply an unusual model and receive a high score. To limit these effects, we suggest to use the discussed quality measures with a generalization-aware modification [7]. That is, instead of using the interestingness measures discussed above directly, we compute the score difference between a subgroup and the maximum score in any of its direct generalizations and use it as a final score. For example, assuming the subgroup $A \wedge B$ has a base score of 100, and the subgroups A and B have score 40 and 70 respectively, then the final (adapted) score of $A \wedge B$ is $100 - \max(40, 70) = 30$. Efficient search for this type of interestingness measure can (preferably) be achieved by using a level-wise search strategy such as apriori, or by extensive caching of subgroup scores.

4 Experimental Evaluation

We demonstrate the benefits of the proposed approach in a study with synthetically generated data, and a case study on real world data.

4.1 Setup and Implementation

For our evaluation, we implemented the mediation model class as an extension of the exceptional model mining Python library *pysubgroup* [13]. For fitting the mediation models and calculating Wald Test score we employed the well-known *lavaan* package with the *nlminb* [18] solver in R, which we connected to *pysubgroup* with the *rpy2* R-Python bridge. As a result of connecting these two specialized highly efficient implementations, all discussed experiments could be finished within seconds on a standard desktop machine.¹

¹ Code is available at <http://florian.lemmerich.net/mediation-emm>.

Table 1. Results for the synthetic data with an interestingness measure based on the Wald Test on the direct effect. The p-values p_{WT} are Bonferroni corrected.

Rank	Score	p_{WT}	Description	# Instances	eff_{dir}	eff_{ind}
1	218.32	$< 10^{-14}$	A=A1 AND B=B1	1000	0.049	0.754
2	72.49	$< 10^{-14}$	A=A1	2000	0.558	0.287
3	72.49	$< 10^{-14}$	A=A2	2000	0.859	0.117
4	66.68	$3.30 \cdot 10^{-13}$	B=B1	2000	0.555	0.331
5	66.68	$3.30 \cdot 10^{-13}$	B=B2	2000	0.844	0.091
6	7.58	> 0.5	N17=1	227	0.536	0.248
7	7.58	> 0.5	N17=0	3773	0.733	0.186
8	7.25	> 0.5	N11=1 AND N15=0	20	0.171	0.147

4.2 Experiments on Synthetic Data

To evaluate our interestingness measures, we test if they are able to recover relevant subgroups among noise. For that purpose, we created synthetic data as follows: First, we defined two mediation models with manually chosen parameters. While we tried different settings with equivalent results, we report here on results for one model with a strong mediator effect ($(a, b, c') = (0.9, 0.8, 0.1)$), and one with a strong direct effect ($(a, b, c') = (0.3, 0.4, 0.8)$). The residual variances were fixed at one. Then, we create the model attributes of 1000 instances by the first model, and of 3000 instances by the second model. Next, we generate two binary descriptive variables A and B such that the instances from the first model are assigned ($A = A1, B = B1$), and the instances from the second model in equal parts ($A = A1, B = B2$), ($A = A2, B = B1$), and ($A = A2, B = B2$). Additionally, we generate 20 binary variables with random noise with a random probability $p \in [0, 1]$. The exceptional model mining task then should recover the subgroup $A = A1 \wedge B = B1$ as the one with significantly different effects, while subgroups with noise-based descriptions should not appear in the top subgroups.

Table 1 shows the results, i.e., the top-8 subgroups, of our approach for a representative task with a maximum search depth of two and using the Wald Test for the direct effect as interestingness measure. We can observe that we successfully recovered the correct subgroup $A = A1 \wedge B = B1$ as the one with the most exceptional effect by a wide margin w.r.t to its score. The following subgroups are the generalizations $B = A1$, $B = B2$, $A = A1$, and $A = A2$ of this main pattern, which –by construction– also exhibit an unusual direct effect compared to the subgroup complement. Only starting from position 6, subgroups described by noise attributes follow. Note that the top 5 subgroups can be shown to be highly significant by the Wald test, but all noise-based subgroups are not statistically significant according to this test. Results for other Wald Test-based interestingness measure equally recover the correct subgroup as the most interesting one. We tested these results for several generated datasets with different effect parameterizations and could identify the right subgroups

Table 2. Results for the European social survey data with a wald test-based interest-iness measures.

Rank	Score	p_{WT}	Subgroup description	# Instances	eff_{dir}	eff_{ind}
1	23.538	< .001	hincfel = Living comfortably on present income	12427	0.070	0.043
2	21.903	< .001	edulvla = Tertiary education completed (ISCED 5–6)	12233	0.080	0.041
3	20.986	< .001	edulvla = Less than lower secondary education (ISCED 0–1)	6127	−0.121	0.112
4	12.083	< .001	maritala = Married	23933	0.072	0.047
5	11.899	< .001	28 ≤ agea < 41	9831	0.072	0.040

consistently. Thus, in summary, we show that our approach is in principle capable to recover a subgroup with a different model compared to the remaining data reliably.

4.3 Example: European Social Survey Data

Next, we illustrate the benefits of our approach in an empirical example inspired by a psychological case study [19]. In that work, the authors investigated – roughly speaking – differences between religious and non-religious individuals with respect to social recognition and potential effects on their subjective well-being. For their analysis, the authors analyzed a large-scale dataset from the European Social Survey (third round) [6] and utilized a mediation model between the variables *religiosity* (independent variable), *happiness* (dependent variable), and *social recognition* (mediator variable). Here, we show how the contributions of our paper can be used to study this model with the same data on a more fine-grained level. This demonstration, however, should not be regarded as a full scale empirical case study since some simplifying assumptions have been made, e.g., additional covariates, mediators, and weights on the survey answers have been ignored.

For data preprocessing, we performed the following steps: From the full survey data ($N = 47,099$), we first selected relevant attributes. Then, we performed a two-dimensional confirmatory factor analysis with the latent variables *religiosity* and *social recognition*. Religiosity was measured by two indicators self-identification (*rlgdgr*) and attendance of religious services (*rlgatnd*), and social recognition was measured by three indicators perceived respect, perceived unfair treatment and perceived recognition. If necessary, indicators were reverse-coded for consistency. Based on the confirmatory factor analysis, factor scores were computed and used in the further analysis.

Table 3. Results for the European Social Survey Data with an exploratory interestingness measure $q = sg\ size^{0.1} \cdot |eff_{ind_{sg}} - eff_{ind}|$ and $sg_{size} > 20$, i.e., $a = 0.1$, minimum 20 instances.

Rank	score	p_{WT}	Subgroup description	# Inst.	eff_{dir}	eff_{ind}
1	1.201	0.030	rlgdnm = Islamic \wedge age \geq 65.0	44	-0.389	0.955
2	1.099	0.037	maritala = Never married \wedge edulvla = Other	20	0.092	0.962
3	0.647	0.141	rlgdnm = Islamic \wedge maritala = Widowed	32	0.316	0.593
4	0.636	0.011	trstlgl = 1 \wedge uempli = Marked	60	-0.148	0.625
5	0.560	0.060	agea < 28 \wedge rlgdnm = Eastern rel.	26	-0.547	-0.539

Since the original study had a focus on the indirect effect, we now set out to find circumstances (subgroups), under which the indirect effect in the mediation model was significantly different compared to the complement. As the search space for candidate subgroup description, we used the variables age, marital status (*maritala*), unemployment status (*uempli*, *uempli*), feeling about household’s income (*hincfel*), highest education level (*edulvla*), trust in the legal system (*trstlgl*), religion or denomination at present (*rlgdnm*), ever belonging to a religion or denomination (*rlgblge*) and years living in current country (*livecntr*).

The indirect effect for the model corresponding to the overall data was 0.065. We employed exceptional model mining analysis with maximum search depth of two, the Wald test for the indirect effect as interestingness measure, and the redundancy reduction described above. Results are shown in Table 2. Top subgroups include those living comfortably with their income (rank 1), highly educated people with tertiary education completed (rank 2), lower educated people with less than secondary education completed (rank 3), married people (rank 4), and people in the age range between 28 and 41 years (rank 5). For all these subgroups except lower educated people the indirect effect of religiosity on happiness via social reputation is smaller compared to the complementary group. In contrast, for those with low education (rank 3) this indirect effect is higher. However, while for all these subgroups the change in indirect effect is highly significant due to large subgroup sizes, the changes in effect sizes are only moderate.

To find smaller subgroups with stronger effects, we used in a second exceptional model mining run an exploratory interestingness measure with setting $a = 0.1$ and a subgroup size (number of instances) constraint of 20. As expected, this indeed leads to results, cf. Table 3, with much smaller subgroup sizes, but stronger changes in effects. For example, the top subgroup now describes elderly muslims, for which the size of the indirect effect is substantially increased. Note

that in this case the results can not immediately be confirmed as significant, but can rather be seen as candidates for more detailed investigations in future studies.

In summary, with exceptional model mining on mediation models, we could find several interesting subgroups w.r.t. to the mediation model between religiosity, social reputation, and happiness in the analyzed survey data. We also found indications that the interestingness measures proposed in this paper lead to results with their desired properties.

5 Conclusions

In this paper, we integrated mediation analysis as a novel model class into the exceptional model mining framework. This enabled us to mine for subgroups in the data that induce exceptionally different model effects compared to the rest of the data set. We introduced statistically principled and exploratory interestingness measures for this task and showed its benefits in synthetic and empirical application examples.

In the future, we will investigate the applicability of exceptional model mining towards more complex structural equation models such as latent growth models or confirmatory factor analysis. Furthermore, we look for ways to improve the runtime of the suggested approach, e.g., by introducing pruning into the search.

Acknowledgement. Funded by the Excellence Initiative of the German federal and state governments.

References

1. Baron, R.M., Kenny, D.A.: The moderator-mediator variable distinction in social psychological research: conceptual, strategic and statistical considerations. *J. Pers. Soc. Psychol.* **51**(6), 1173–1182 (1986)
2. Bollen, K.A.: *Structural Equation Modeling with Latent Variables*. Wiley, New York (1989)
3. Brandmaier, A.M., von Oertzen, T., McArdle, J.J., Lindenberger, U.: Structural equation model trees. *Psychol. Methods* **18**(1), 71 (2013)
4. Duivesteijn, W., Feelders, A., Knobbe, A.: Different slopes for different folks: mining for exceptional regression models with cook’s distance. In: *International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 868–876 (2012)
5. Duivesteijn, W., Knobbe, A., Feelders, A., van Leeuwen, M.: Subgroup discovery meets Bayesian networks—an exceptional model mining approach. In: *International Conference on Data Mining (ICDM)*, pp. 158–167 (2010)
6. ESS3: ESS Round 3: European Social Survey Round 3 Data (2006)
7. Grosskreutz, H., Boley, M., Krause-Traudes, M.: Subgroup discovery for election analysis: a case study in descriptive data mining. In: Pfahringer, B., Holmes, G., Hoffmann, A. (eds.) *DS 2010. LNCS (LNAI)*, vol. 6332, pp. 57–71. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16184-1_5
8. Herrera, F., Carmona, C.J., González, P., Del Jesus, M.J.: An overview on subgroup discovery: foundations and applications. *Knowl. Inf. Syst.* **29**(3), 495–525 (2011)

9. Imai, K., Keele, L., Tingley, D.: A general approach to causal mediation analysis. *Psychol. Methods* **15**, 309–334 (2010)
10. van Kesteren, E.J., Oberski, D.L.: Exploratory mediation analysis with many potential mediators. *Structural Equation Modeling*, 1–14 (2019)
11. Klösgen, W.: Explora: a multipattern and multistrategy discovery assistant. In: *Advances in Knowledge Discovery and Data Mining*, pp. 249–271. American Association for Artificial Intelligence (1996)
12. Leman, D., Feelders, A., Knobbe, A.: Exceptional model mining. In: Daelemans, W., Goethals, B., Morik, K. (eds.) *ECML PKDD 2008*. LNCS (LNAI), vol. 5212, pp. 1–16. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87481-2_1
13. Lemmerich, F., Becker, M.: pysubgroup: easy-to-use subgroup discovery in python. In: Brefeld, U., et al. (eds.) *ECML PKDD 2018*. LNCS (LNAI), vol. 11053, pp. 658–662. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-10997-4_46
14. Lemmerich, F., Becker, M., Singer, P., Helic, D., Hotho, A., Strohmaier, M.: Mining subgroups with exceptional transition behavior. In: *International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 965–974 (2016)
15. Mayer, A., Thoemmes, F., Rose, N., Steyer, R., West, S.G.: Theory and analysis of total, direct and indirect causal effects. *Multivar. Behav. Res.* **49**(5), 425–442 (2014)
16. Pearl, J.: Direct and indirect effects. In: *Conference on Uncertainty in Artificial Intelligence*, pp. 411–420. Morgan Kaufmann, San Francisco (2001)
17. Preacher, K.J., Rucker, D.D., Hayes, A.F.: Addressing moderated mediation hypotheses: theory, methods, and prescriptions. *Multivar. Behav. Res.* **42**, 185–227 (2007)
18. Rosseel, Y.: lavaan: an R package for structural equation modeling. *J. Stat. Softw.* **48**(2), 1–36 (2012)
19. Stavrova, O.: *Fitting in and Getting Happy: How Conformity to Societal Norms Affects Subjective Well-Being*, vol. 4. Campus Verlag (2014)

Machine Learning Applications



Multivariate Predictive Clustering Trees for Classification

Tomaz Stepišnik^{1,2}(✉)  and Dragi Kocev^{1,2,3} 

¹ Jožef Stefan Institute, Ljubljana, Slovenia
{tomaz.stepisnik, dragi.kocev}@ijs.si

² Jožef Stefan International Postgraduate School, Ljubljana, Slovenia

³ Bias Variance Labs, d.o.o., Ljubljana, Slovenia

Abstract. Decision trees are well established machine learning models that combined in ensembles produce state-of-the-art predictive performance. Predictive clustering trees are a generalization of standard classification and regression trees towards structured output prediction and semi-supervised learning. Most of the research attention is on univariate decision trees, whereas multivariate decision trees, in which multiple attributes can appear in a test, are less widely used. In this paper, we present a multivariate variant of predictive clustering trees, and experimentally evaluate it on 12 classification tasks. Our method shows good predictive performance and computational efficiency, and we illustrate its potential for performing feature ranking.

Keywords: Predictive clustering trees · Multivariate decision trees · Classification · Multi-label classification

1 Introduction

In predictive modeling, the task is to use a set of learning examples to construct a model that can be used to make accurate predictions for unseen examples. The examples are described with features and associated with a target variable. The model uses the features to predict the value of the target variable. Depending on the type of the target, we differentiate between different predictive modelling tasks. In this paper, we focus on classification, where the target (or targets) has a finite set of possible values. When there are only two possible target values, the task is called binary classification. If there are more than two possible values, it is called multi-class classification. Furthermore, in multi-label classification, a single example can have multiple labels, and in hierarchical multi-label classification, the labels are also organized in a hierarchy (e.g., biological taxonomy).

Decision trees [3] are a well known and widely used machine learning method. However, most of the research focus is on univariate trees, i.e., trees in which the tests for splitting the examples use a single feature. Multivariate decision trees, where the tests in the internal nodes can use more than one attribute, have received much less attention. More flexible splits can lead to smaller trees that

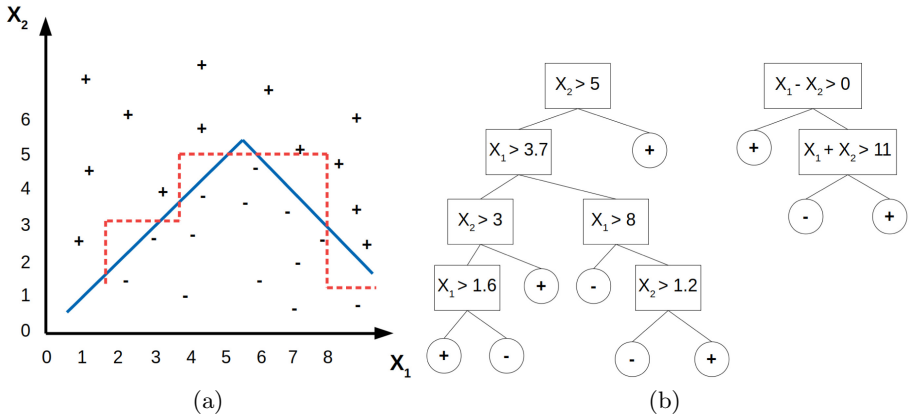


Fig. 1. A toy dataset (a) with drawn decision boundaries learned by a univariate (red, dashed) and multivariate (blue, solid) decision tree (b). (Color figure online)

provide better generalization, but broadening the space of possible splits also complicates the search for the optimal split. Figure 1 demonstrates the advantage of multivariate trees compared to univariate trees on a toy dataset. Some existing multivariate decision trees have been proposed [3, 7–9], however they are computationally inefficient [3, 8] or focused on a very specific task [9].

Predictive clustering trees (PCTs) [6] are a generalization of standard decision trees. When used in a standard classification or regression setting, they work the same as classification or regression trees [3]. However, they support different splitting heuristics and prototype functions in the leaves, and can be used for a wider variety of predictive modeling tasks. When used in ensembles, they offer state-of-the-art performance [6], and they have been successfully applied to a variety of real world problems [4, 10, 11].

In this paper, we present a multivariate variant of predictive clustering trees, that retains the flexibility of the original predictive clustering trees. As in univariate PCTs, splits in multivariate PCTs are optimized to minimize the impurity of the clustering (target) variables on each side of the hyperplane. We engineer a differential criterion function that allows for efficient optimization with gradient descent methods. We experimentally evaluate our method and show that it often outperforms standard PCTs, and is more computationally efficient on datasets with many features or labels.

We first describe our method, then theoretically analyze its time complexity and compare it to the top-down induction of univariate PCTs. We present an experimental evaluation of our method both in single tree and ensemble setting on different classification tasks (binary, multi-class and multi-label). We also perform parameter sensitivity analysis and show how the learned multivariate trees can be interpreted via feature importance scores.

2 Method Description

2.1 Tree Induction Overview

Tree based models, including predictive clustering trees, are most often built using a greedy top-down induction algorithm [3], outlined in Algorithm 1. Examples are recursively partitioned until no acceptable test is found. At that point a leaf is created, where the prototype (majority class or mean value/vector) of the remaining examples is stored for prediction. When searching for a test, one goes through all the possible splits for each individual attribute and finds the one that produces the partitions with the lowest impurity. The impurity can depend on the task, but PCTs by default use the sum of target variances to measure the impurity of a set of examples [6]. The sum can be weighted, if we want to prioritize reducing the impurity of certain targets (e.g., in hierarchical multi-label classification, we can prioritize labels higher in the hierarchy).

The prototype function also depends on the task. Typically, the mean values of the target variables are stored in the leaves. For classification, they are interpreted as the probability that an example in that leaf has the corresponding labels. To make a prediction, an example is passed through the tree according to the tests in internal nodes until it reaches a leaf. In binary and multi-class classification, the label with the highest probability is predicted (majority class in the leaf), whereas in (hierarchical) multi-label classification, all labels with probabilities greater than some user-defined threshold (usually 0.5) are predicted.

Individual trees can be useful, because they are easily interpretable models. However, to achieve state of the art predictive performance, they are most often used in an ensemble setting [1, 2, 6] (bagging ensembles and random forests).

2.2 Learning Multi-variate Splits

Our proposed method follows the greedy top-down induction method, but changes the split search procedure. Instead of only considering univariate splits, we allow any linear combination of features to be a split. This means that split boundaries are no longer only parallel to the coordinate axes, but can be any hyperplane in the feature space (as shown in Fig. 1).

Algorithm 1. Top-down decision tree induction: The inputs are matrices of features $X \in \mathcal{R}^{N \times D}$ and targets $Y \in \mathcal{R}^{N \times T}$.

```

1: procedure GROW_TREE( $X, Y$ )
2:    $test = \text{find\_test}(X, Y)$ 
3:   if  $\text{acceptable}(test)$  then
4:      $X_1, Y_1, X_2, Y_2 = \text{split}(X, Y, test)$ 
5:      $\text{left\_subtree} = \text{grow\_tree}(X_1, Y_1)$ 
6:      $\text{right\_subtree} = \text{grow\_tree}(X_2, Y_2)$ 
7:     return  $\text{Node}(test, \text{left\_subtree}, \text{right\_subtree})$ 
8:   else
9:     return  $\text{Leaf}(\text{prototype}(Y))$ 

```

Let $X \in \mathcal{R}^{N \times D}$ be the matrix containing the D features of the N examples in the learning set. Let $Y \in \{0, 1\}^{N \times L}$ be the binary matrix denoting which of the L labels are present for each of the N examples in the learning set. For example, in binary classification $L = 1$, whereas for multi-class and (hierarchical) multi-label classification L is the number of possible labels. In multi-class classification, each example only has one label, i.e., the sum of each row in Y equals 1, whereas in multi-label classification there can be multiple 1s in each row. Below, M_i will refer to the i -th row of the matrix M and $M_{.i}$ to the i -th column, where M will be either X or Y . We wish to learn a vector of weights $w \in \mathcal{R}^D$ and a bias term $b \in \mathcal{R}$ that define a hyperplane, which splits the learning examples into two subsets. We will refer to the subset where $X_i \cdot w + b \geq 0$ as the positive subset, and the subset where $X_i \cdot w + b < 0$ as the negative subset. We want the examples in the same subset to have similar labels.

First, we calculate the vector $s = \sigma(Xw + b) \in [0, 1]^N$, where the sigmoid function σ is applied component-wise. The vector s contains values from the $[0, 1]$ interval, and we treat it as a fuzzy membership indicator. Specifically, the value s_i tells us how much the i -th example belongs to the positive subset, whereas the value $1 - s_i$ tells us how much it belongs to the negative subset.

To measure the impurity of the positive subset, we calculate the weighted variance of each column (label) in Y , and we weigh each row (example) with its corresponding weight in s . To measure the impurity of the negative subset, we calculate the weighted variances with weights from $1 - s$. Weighted variance of a vector $v \in \mathcal{R}^n$ with weights $a \in \mathcal{R}^n$ is defined as

$$\text{var}(v, a) = \frac{\sum_i^n a_i (v_i - \text{mean}(v, a))^2}{A} = \text{mean}(v^2, a) - \text{mean}(v, a)^2,$$

where $A = \sum_i^n a_i$ is the sum of weights and $\text{mean}(v, a) = \frac{1}{A} \sum_i^n a_i v_i$ is the weighted mean of v .

The final impurity is the weighted sum of weighted variances over all the labels. The impurity of the positive subset is $\text{imp}(Y, p, s) = \sum_j^L p_j \text{var}(Y_{.j}, s)$, and similarly $\text{imp}(Y, p, 1 - s)$ is the impurity of the negative subset. As mentioned above, weights $p \in \mathcal{R}^L$ enable us to give different priorities to different labels. The final objective function we want to minimize is

$$f(w, b) = S * \text{imp}(Y, p, s) + (n - S) * \text{imp}(Y, p, 1 - s),$$

where $s = \sigma(Xw + b)$ and $S = \sum_i^n s_i$. The terms S and $n - S$ represent the sizes of positive and negative subsets, and are added to incentivize balanced splits.

For examples that are not close to the hyperplane, s_i is close to 0 or 1. Weighted variances are therefore approximations of the variances of the subsets, that the hyperplane produces, and the optimization function is almost equivalent to the one used by the original PCTs. The only difference is that we use the weighted variances instead of directly calculating the subset variances. This makes the objective function differentiable and enables us to use the efficient Adam [5] gradient descent optimization method. For each split the w is initialized randomly, and then b is set so that initially the examples are split in half.

Like standard PCTs, multivariate PCTs can also be used in ensemble setting, by using different bootstrapped replicates of the learning sets for different ensemble members (bagging) and/or using random subsets of features for individual splits (random forests).

2.3 Time Complexity Analysis

The time complexity of splitting a node in standard PCTs is $\mathcal{O}(DN \log N) + \mathcal{O}(LDN)$, where N is the number of examples in the node, D the number of features and L the number of labels [6]. The first term is the result of sorting the examples by each feature, when searching for the optimal split by that feature. The second term comes from the evaluation of the splits.

In multivariate trees, we perform iterative optimization of the splits. The most costly parts of each iteration (epoch) are the matrix-vector products of X with w ($\mathcal{O}(ND)$) and Y with s ($\mathcal{O}(NL)$), required to calculate the objective function and its derivative. The time complexity is therefore $\mathcal{O}(EN(D + L))$, where E is the number of optimization epochs. This shows that our approach has an efficiency edge when D and L are large.

3 Evaluation and Discussion

3.1 Experimental Design

To evaluate our method, we compare it to standard PCTs in terms of predictive performance, model size and learning time. For benchmarking, we use 3 datasets for each of binary classification, multi-class classification, multi-label classification and hierarchical multi-label classification. All datasets are publicly available online^{1,2} and their properties are presented in Table 1.

Table 1. Properties of the datasets used for the evaluation: N is the number of examples, D is the number of features and L is the number of labels.

dataset	N	D	L	task	dataset	N	D	L	task
banknote	1372	4	2	binary	birds	645	260	19	multi-label
OVA_breast	1545	10936	2	binary	scene	2407	294	6	multi-label
diabetes	768	8	2	binary	bibtex	7395	1836	159	multi-label
balance	625	4	3	multi-class	diatoms	1098	200	80	hierarchical
imagesegment	2310	19	7	multi-class	clef07d	11006	80	46	hierarchical
gasdrift	13910	128	6	multi-class	enron	1648	1001	56	hierarchical

For standard PCTs, we use their java implementation in the CLUS framework³. We implemented the multivariate PCTs in a python package *spyct* – it is available online⁴.

¹ www.openml.org.

² http://kt.ijs.si/DragiKoccev/PhD/resources/doku.php?id=phd_thesis_datasets.

³ <http://source.ijs.si/ktclus/clus-public/>.

⁴ <https://gitlab.com/TStepi/spyct>.

To measure the performance, we use F1 score for binary and multi-class classification, with macro averaging over the labels for the latter. For (hierarchical) multi-label classification, we use ranking loss [12], which computes the average number of label pairs that are incorrectly ordered (by their predicted probability). In hierarchical multi-label classification we use the default label weighting strategy from CLUS: label has a weight of 0.75^d , where d is the depth of that label in the hierarchy ($d = 0$ for root nodes). We estimate the performance with 10-fold cross validation. We evaluate both single trees and bagging ensembles of 50 trees. The splits in multivariate PCTs are optimized for a maximum of 100 epochs with learning rate 0.1. The optimization is stopped early, if the objective function does not decrease after an epoch. Before learning multivariate PCTs we also standardize each feature to 0 mean and standard deviation of 1. We include the standardization in the times reported below. All experiments were performed on the same computer with Intel i7-6700K processor. Ensembles used 4 cores to learn trees in parallel.

3.2 Results

In this section, we present and discuss the results from the evaluation. Model sizes and learning times are reported only for the ensembles, because they are more reliable than single tree results. Figure 2 presents the results for binary and multi-class datasets. The most notable difference in F1 score is on the balance dataset, where multivariate trees significantly outperform univariate trees. The same holds for bagging ensembles. On the other datasets, the differences in single tree performances are small, and for ensembles they are barely noticeable. In terms of model sizes, multivariate trees are mostly much smaller. This was expected, because multivariate splits are much more expressive than univariate. Multivariate trees are also faster to learn on datasets with many features (OVA_breast, gasdrift) datasets, and slower on datasets with few features. These results support the theoretical time complexity analysis from Sect. 2.3.

Figure 3 presents the results for multi-label and hierarchical datasets. For single trees, our method outperforms standard PCTs on most datasets, and the same is true for ensembles. Interesting cases are bibtex and enron datasets, our method performs poorly in single tree setting, but very well in ensemble setting. This indicates that overfitting, was the reason for poor performance of single trees. Tree sizes are consistently smaller over all datasets, and learning times are also strongly in favor of our method. The reason is that these datasets mostly have many possible labels, and learning time for univariate trees scales with $D \times L$, whereas for our method it only scales with $D + L$.

3.3 Parameter Sensitivity Analysis

We also performed parameter sensitivity analysis. We evaluated our method with different combinations of learning rate and maximum numbers of optimization epochs. The results of single trees on two datasets are presented in Fig. 4.

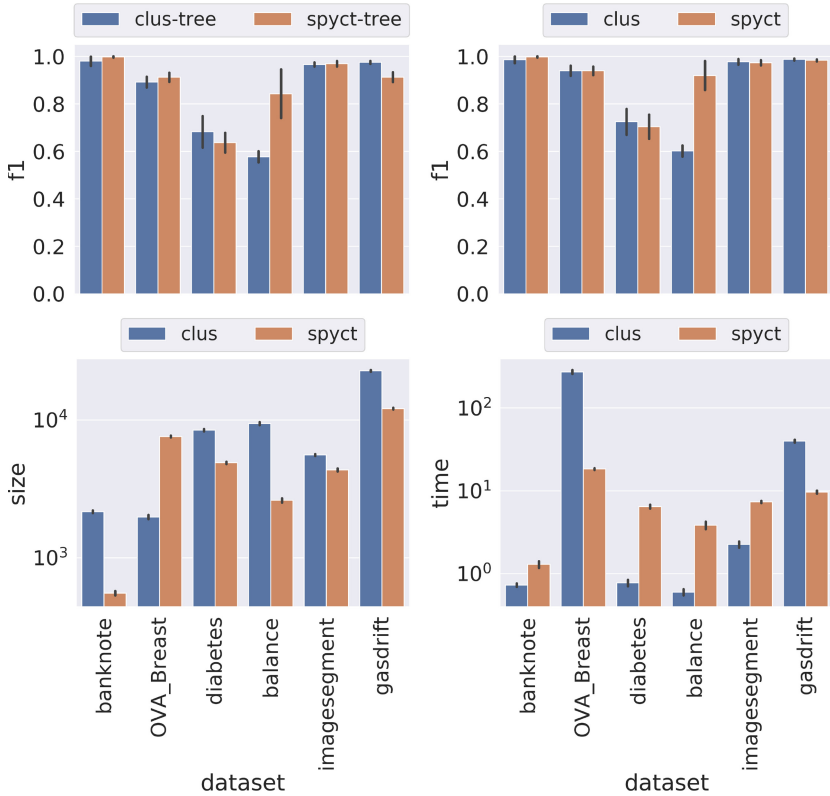


Fig. 2. Experimental results for binary and multi-class datasets. Top row presents the predictive performance (f1, higher is better), bottom row shows efficiency in terms of model size (number of nodes) and learning time (in seconds). Note that times and sizes are on logarithmic scale.

As might be expected, increasing the maximum number of epochs generally leads to better predictive performance, but also comes with greater learning times. On the other hand, there is no general rule for learning rates. While increasing the learning rate might make finding a good split faster (e.g., image-segment dataset), too large learning rate can also prevent the algorithm from finding a good solution (e.g., gasdrift dataset). With higher learning rates, early stopping comes into effect quickly (objective function stops improving), so increasing the maximum number of epochs has little effect on learning time.

Our selection of maximum 100 epochs at 0.1 learning rate is a sensible default, that proved to work well overall. However, it is unlikely to be optimal for any particular dataset. To achieve optimum predictive performance on a given dataset, the parameters (especially learning rate) should be fine-tuned for it specifically.

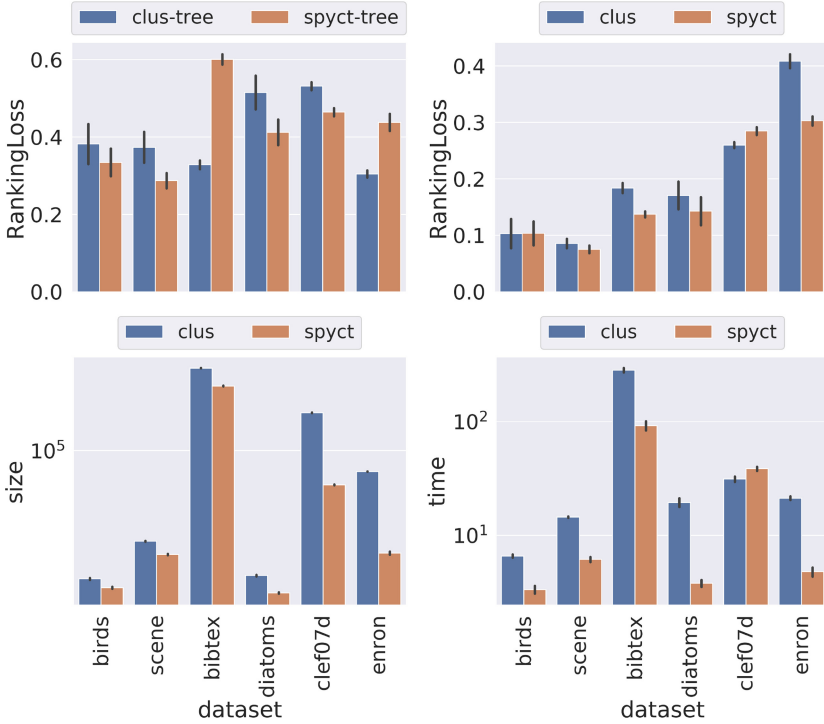


Fig. 3. Experimental results for multi-label and hierarchical multi-label datasets. Top row presents the predictive performance (ranking loss, lower is better), bottom row shows efficiency in terms of model size (number of nodes) and learning time (in seconds). Note that times and sizes are on logarithmic scale.

3.4 Interpretability

One of the advantages of decision trees is their interpretability. Multivariate decision trees are harder to interpret than univariate, especially for datasets with many attributes. However, large decision trees and ensembles thereof are also difficult to interpret. In those cases, feature importance scores are often calculated from the model and used to gain insights into its inner working. We illustrate the ability of our method to produce meaningful feature importance scores.

To obtain the feature importance vector from a tree, we calculate the weighted sum of vectors $|w|$ learned in the split nodes. Each vector w is normalized and weighted by the share of the dataset that was split in that node. This way, splits closer to the top of the tree, that influence more examples, have a bigger impact on feature importance. To get feature importance vector from an ensemble of trees, we simply calculate the average of feature importances of individual trees.

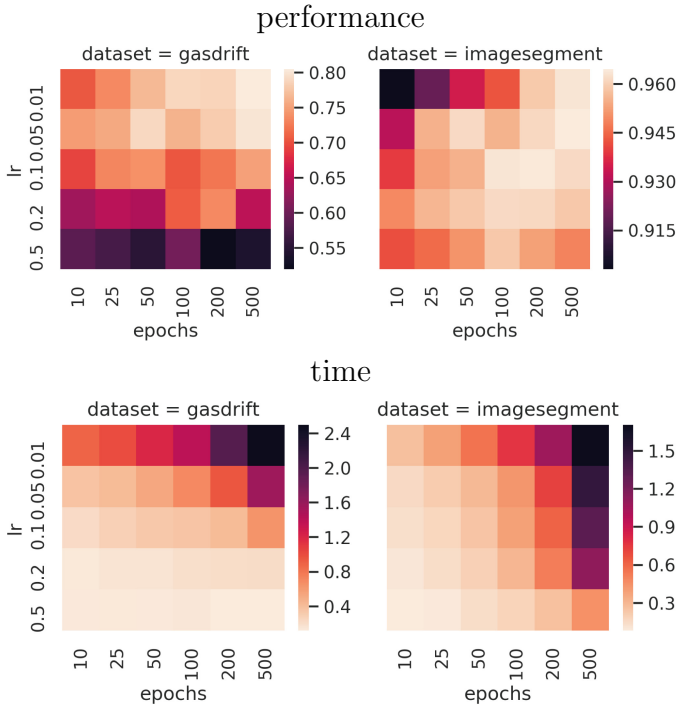


Fig. 4. Heatmaps showing the F1 scores (top row) and learning time (bottom row) depending on the parameter selection on the gasdrift and imagesegment datasets. Lighter color indicates better values.

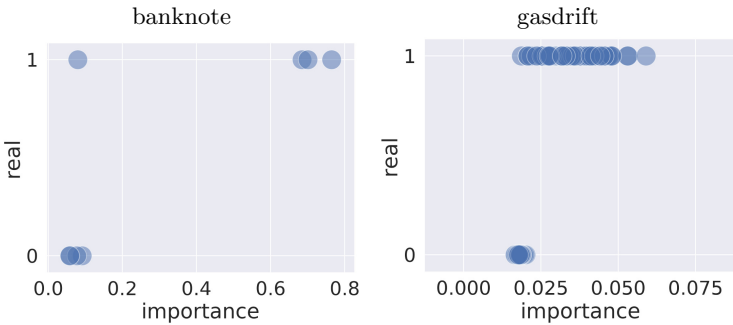


Fig. 5. Feature importances calculated on two datasets. The x-axis shows the importance score, and y-axis indicates whether the feature was real (1) or random (0).

To evaluate our feature importance scoring, we add random features (for each real feature we add one random feature) to a dataset, and compare their feature importance to the importance of real features. Results for two datasets are presented in Fig. 5. We can clearly see that the added random features have very low scores, meaning that the model does not rely on them. There are some real features that also have low scores, but that is to be expected. Datasets often include features that are not useful for predicting the target. This experiment confirms that feature importances obtained with our method are meaningful.

4 Conclusion

We present a multivariate variant of predictive clustering trees, that retains the flexibility of the original framework. We evaluate our proposed method on binary, multi-class, multi-label and hierarchical multi-label classification datasets from various domain. The results show that our method outperforms univariate PCTs and generally produces smaller models. Both theoretical and experimental analyses show that learning time is smaller on datasets with many features or labels. We also perform parameter sensitivity analysis, and demonstrate the ability of our method to produce meaningful feature ranking.

For future work, we plan to evaluate our method on other tasks supported by standard PCTs, such as (multi-target) regression and semi-supervised learning. We plan to evaluate our method in random forest setting and explore its potential advantages in exploiting sparse data using sparse matrix multiplication.

References

1. Breiman, L.: Bagging predictors. *Mach. Learn.* **24**(2), 123–140 (1996)
2. Breiman, L.: Random forests. *Mach. Learn.* **45**(1), 5–32 (2001)
3. Breiman, L., Friedman, J., Olshen, R., Stone, C.J.: *Classification and Regression Trees*. Chapman & Hall/CRC, San Francisco (1984)
4. Debeljak, M., Squire, G.R., Kocev, D., Hawes, C., Young, M.W., Džeroski, S.: Analysis of time series data on agroecosystem vegetation using predictive clustering trees. *Ecol. Model.* **222**(14), 2524–2529 (2011)
5. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014)
6. Kocev, D., Vens, C., Struyf, J., Džeroski, S.: Tree ensembles for predicting structured outputs. *Pattern Recogn.* **46**(3), 817–833 (2013)
7. Menze, B.H., Kelm, B.M., Splitthoff, D.N., Koethe, U., Hamprecht, F.A.: On oblique random forests. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgianis, M. (eds.) *ECML PKDD 2011. LNCS (LNAI)*, vol. 6912, pp. 453–469. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23783-6_29
8. Murthy, S.K., Kasif, S., Salzberg, S.: A system for induction of oblique decision trees. *J. Artif. Intell. Res.* **2**, 1–32 (1994)
9. Prabhu, Y., Varma, M.: FastXML: a fast, accurate and stable tree-classifier for extreme multi-label learning. In: *ACM-Association for Computing Machinery* (2014)

10. Slavkov, I., Gjorgjioski, V., Struyf, J., Džeroski, S.: Finding explained groups of time-course gene expression profiles with predictive clustering trees. *Mol. BioSyst.* **6**(4), 729–740 (2010)
11. Struyf, J., Džeroski, S., Blockeel, H., Clare, A.: Hierarchical multi-classification with predictive clustering trees in functional genomics. In: Bento, C., Cardoso, A., Dias, G. (eds.) *EPIA 2005. LNCS (LNAI)*, vol. 3808, pp. 272–283. Springer, Heidelberg (2005). https://doi.org/10.1007/11595014_27
12. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. In: Maimon, O., Rokach, L. (eds.) *Data Mining and Knowledge Discovery Handbook*, pp. 667–685. Springer, Boston (2010). https://doi.org/10.1007/978-0-387-09823-4_34



Comparison of Machine Learning Methods to Detect Anomalies in the Activity of Dairy Cows

Nicolas Wagner^{1,2}(✉), Violaine Antoine¹, Jonas Koko¹, Marie-Madeleine Mialon², Romain Lardy², and Isabelle Veissier²

¹ UCA, LIMOS, UMR 6158, CNRS, Clermont-Ferrand, France
nicolas.wagner@uca.fr

² UCA, INRAE, UMR Herbivores, 63122 Saint-Genès-Champanelle, France

Abstract. Farmers need to detect any anomaly in animals as soon as possible for production efficiency (e.g. detection of estrus) and animal welfare (e.g. detection of diseases). The number of animals per farm is however increasing, making it difficult to detect anomalies. To help solving this problem, we undertook a study on dairy cows, in which their activity was captured by an indoor tracking system and considered as time series. The state of cows (diseases, estrus, no problem) was manually labelled by animal caretakers or by a sensor for ruminal pH (acidosis). In the present study, we propose a new Fourier based method (FBAT) to detect anomalies in time series. We compare FBAT with the best machine learning methods for time series classification in the current literature (BOSS, Hive-Cote, DTW, FCN and ResNet). It follows that BOSS, FBAT and deep learning methods yield the best performance but with different characteristics.

Keywords: Machine learning · Deep learning · Time series classification · Detection of anomalies · Precision livestock farming

1 Introduction

Precision livestock farming is based on the use of smart technologies (mainly sensors) to monitor closely the animals or their environment. The aim is to optimize the production and reduce farmers work load. The increase in computers storage capacity and in the precision of sensors makes possible to record a high quantity of data which requires automatic processing to be used by farmers. Machine learning tools is beginning to be employed to extract relevant information from these massive data. For example, machine learning has been used to determine grass growth from satellite and weather data [10] or to predict the quantity of manure to be spread on pastures or crops as fertilizer [11].

Farmers need to detect any anomaly in animals as soon as possible both for milk production efficiency and animal welfare. Such a detection seems possible

through the analysis of the animals' activities. For instance, [15] found that dairy cows' activity varies according to a circadian cycle which significantly changes if the cow is about to be sick or in estrus.

Furthermore, time series classification (TSC) or anomaly detection are among the most challenging problem in machine learning [6, 12, 18] and are present in many fields of science like sensor-based human activity recognition [16], credit card fraud detection [1], electroencephalogram and electrocardiogram analysis [3], geo-distributed networks [5], etc. TSC differs from classical machine learning problems since it deals with data listed in time order. Some algorithms were developed for TSC like Dynamic Time Warping (DTW) [4], Bag of SFA Symbols (BOSS) [14] or Hive-Cote [8]. Most recently, deep learning neural networks as FCN and ResNet were also used and found to outperform the other algorithms [6, 17].

In this paper, we employed algorithms of time series classification (BOSS, DTW, Hive-Cote FCN and ResNet) considered as the best ones. In addition, because the activity of cows follows a circadian cycle, we proposed and tested a new method based on Fourier transformations (Fourier Based Approximation with Thresholding or FBAT).

The first section describes the most popular TSC classifier. The Sect. 2 details our new FBAT method. Section 3 first describes the data set, i.e. time series of activities of dairy cows, then explains the experimental protocol and finally presents the results. Perspectives of the work are given in a conclusion.

2 Time Series Classifier

This section presents the current best algorithms for TSC [2, 6, 17] used as baseline to compare the FBAT method.

2.1 Dynamic Time Warping

DTW [4] is a method that measures the similarity between two time series. It is often used as a distance with the one Nearest Neighbor algorithm (1-NN). Although combining 1-NN and DTW gives good results in practice, however, DTW is not a distance function. Indeed, it does not respect all mathematical properties of a distance especially the triangle inequality [13].

The difference between DTW and standard distance measures is the following: standard distances assume that the i^{th} of a series is aligned with the i^{th} point of an other series while DTW is designed to minimize the effects of shifting and distortion in time series. The DTW method have many advantages. It is easy to employ, it can be used with many algorithms like k-NN and when combined with 1-NN it is one of the best algorithms for time series classification [2]. This makes it an interesting baseline. Its quadratic time complexity is a disadvantage but it remains faster than other algorithms like Hive-Cote.

2.2 Hive-Cote

Hive-Cote [8] is an improved version of the algorithm Cote (or Flat-Cote). Flat-Cote consists in using 35 classifiers of time series classification. Each classifier produces a result and the final decision is based on a vote of all classifiers. The vote is weighted by the training accuracy of each classifier. One problem with Flat-Cote is the flat architecture. It means that all classifiers vote independently. However, some algorithms pertain to the same category and probably give similar results. To solve this problem, Hive-Cote gathers the algorithms into groups called modules. Each module computes the probability for each class to be the solution. This probability is computed with the weighted results of the algorithms for each module. Then, the final solution is the class that has the highest probability across all module's outputs.

Hive-cote is composed of five modules: elastic ensemble, shapelet transform ensemble, BOSS, time series forest and random interval features.

2.3 Fully Convolutional Networks

Fully Convolutional Networks (FCNs) [9] are similar to Convolutional Neural Networks (CNNs) excepted that they contain local pooling layer so as to keep the same dimensionality input through the convolutional layers. In addition, a standard CNN generally ends by a Fully Connected (FC) layer that is replaced by a Global Average Pooling (GAP) in the FCNs. The architecture used in this paper was proposed by [17]. It consists of three parts that are composed of a convolution layer, a Batch Normalization (BN) layer and a ReLu activation layer. These three parts are followed by a GAP layer and a classical softmax layer. The three convolution blocks contain 128, 256 and 128 filters with a filter length of respectively 8, 5 and 3. The stride is set to one with a zero padding that enables to preserve the same length of time series across the network. This architecture has the advantage to remain stable according to the length of the time series (excepted for the last softmax layer). This allows us to use exactly the same network used in [6, 17]. FCN is the best deep learning algorithm on the 44 data sets analyzed in [17].

2.4 ResNet

A Residual Network [7] is close to CNN. The difference lies in shortcuts added from the input of convolution blocks to their output. These shortcuts inject the information that may be lost by the convolutional block. The ResNet proposed by [6, 17] is composed of three convolutional blocks. All blocks are composed of three convolutional layers with respectively a filter's length set to 8, 5 and 3. Each convolutional layer is followed by a BN and Relu layer. The convolutional layers of the first block are composed of 64 filters and the convolutional layers of the second and last block are composed of 128 filters. The three blocks are followed by a global pooling and a softmax layer. This architecture has the same advantage as FCN: it does not vary with the length of the time series. ResNet is the best deep learning algorithm on the 85 data sets tested in [6].

2.5 Bag of SFA Symbols

BOSS [14] is a method that combines the advantage of the Fourier transform and the bag of words model. It allows to reduce noise and to handle variable lengths.

First, a sliding window of size w with a step of 1 is applied over each time series. The obtained time windows are converted into sequences (or words) of symbols of length l with an alphabet size of c using the Symbolic Fourier Approximation (SFA) algorithm. A time series is then represented by the sequences of each window. Finally, a histogram is built using sequences as modal class. The last step consists in using 1-NN algorithm with the BOSS distance function. Given two histograms B_1 and B_2 , the formula of the BOSS distance function is:

$$dist(B_1, B_2) = \sum_{a \in B_1; B_1(a) > 0} [B_1(a) - B_2(a)]^2, \quad (1)$$

where a is a word and $B_i(a)$ the number of occurrences of a in the i^{th} histogram.

3 Fourier Based Approximation with Thresholding

We propose a Fourier Based Approximation with Thresholding (FBAT) method to classify time series by measuring the variations of the cyclic components. It is made to classify time series as normal or abnormal by assuming that an abnormal series includes a break on the cycle. Thus, if the variations of the cyclic components are high, the algorithm classifies the time series S_i as abnormal. The algorithm starts by extracting two sub-series A and B of size p and delayed of q from the input series with $p < |S_i|$ and $p + q < |S_i|$. A Fourier transform is applied on both sub-series to extract their harmonic decomposition. With these harmonics, a new model $m(t)$ is computed for each sub-series following the formula:

$$m(t) = \sum_{f=-z}^z |h_f| \cos(2\pi f \frac{t}{p} + \arg(h_f)), \quad z = 0 \dots \lceil \frac{p-1}{2} \rceil, \quad (2)$$

where h_f is the harmonic corresponding to the frequency f and z is the number of harmonics to keep in the model. Note that h_f is a complex number with $|h_f|$ its modulus and $\arg(h_f)$ its argument. Moreover, the two sub-series A and B are delayed by q . As a consequence, it is necessary to synchronize the models of A and B by applying a temporal shift to the model of B . This shift is performed by adding a delay $-\frac{q}{p}2\pi$ in the formula of the model of B .

A L2-norm distance d_{L2} is then computed between the two models. This distance reflects the variation of the cyclic component of the input time series. A high distance means a high variation and vice versa.

To classify the input time series as normal or abnormal, the algorithm needs to compute a threshold τ for the distance. If $d_{L2} > \tau$, the time series is classified as abnormal. To compute this threshold, all distances d_{L2} are computed for

each time series that belongs to the training set. Then, s samples are computed between the minimum and the maximum obtained distances. The accuracy of the training set is computed for each sample and the sample that yields the best accuracy is chosen to be the threshold.

4 Experiments and Results

4.1 Data Set

The data were collected on 28 Holstein cows during a two month experimentation in which a subacute ruminal acidosis, a metabolic disease common in ruminants, was induced.

Data Construction. The raw data consist of the record of the location of each cow every second with an indoor tracking system (CowView, GEA Farm Technologies, Bönen, Germany). Three activities were identified: *eating* if the cow was located next to the trough, *resting* if it was in a cubicle (resting place) and *in alleys* if the cow was in an alley. These activities were aggregated in a new variable called level of activity. The procedure is described in [15]. We thus obtained for our study time series consisting in the evolution over time of the level of activity of each cow estimated per hour. All anomalies, such as acidosis, oestrus, etc. were noted. The acidosis was detected by a sensor that measured the pH in the rumen.

A set of 28 time series, corresponding to the 28 cows for two months was available. To build the data set, a sliding window of 36 h was applied on each cow to extract sub-series (see justification in next section). The obtained data set was divided into two parts: one for the training and one for test. Half of series labelled as *abnormal* were used for the training set and the other half for the test set, except for the series related to acidosis that were all placed in the test set. Indeed, this specific anomaly was induced by experimenters. The idea is to avoid the perturbation of a classifier during the learning phase with an unnatural anomaly. Finally, to balance the training data set, a reduction of the *normal* series was performed by randomly selecting few ones. The same number of normal series were randomly chosen for the test data set. The training data set is composed of 1088 *normal* series and 972 *abnormal* series. The test set is composed of 1408 *normal* series and 4212 *abnormal* series (including 3180 series with acidosis).

Data Properties. The first property of this data set is that each cow has its own natural daily rhythm based on a circadian cycle with a low activity during the night and a higher activity during the day [15]. This change of activity between nights and days can be modified if the cow is sick or under stress. We decided to work with series of 36 h in order to observe a cycle of more than one day and to be able to detect anomalies with precision (e.g. a normal cycle of 24 h followed by 12 h abnormal).

Figure 1 illustrates the activity of two cows during two normal days. Both cows are more active during the day than at night, in line with the fact that the rhythm is circadian. This figure also illustrates that the rhythm of normal activity can differ between cows and days. Figure 2 illustrates the level of activity of a cow under lame during three days. This shows how the activity of a cow can be modified by an anomaly. Given that there exists normal variations between days (as illustrated by the Cow b, Fig. 1), the difficulty lies in discriminating between changes due to an anomaly and those due to spontaneous variations.

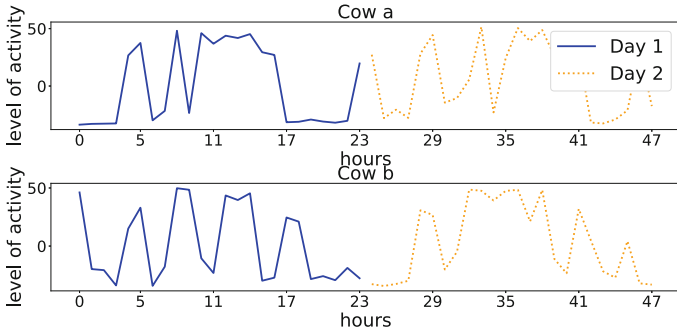


Fig. 1. 48 h of normal activity for two different cows.

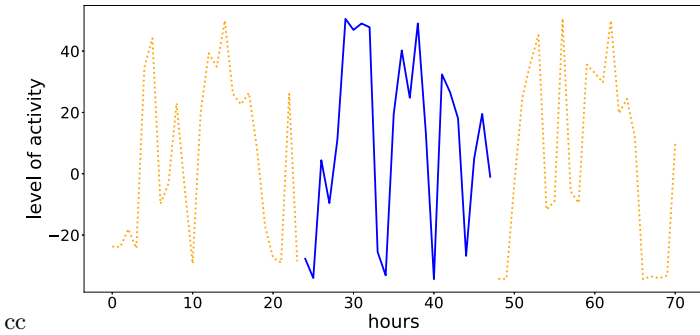


Fig. 2. Level of activity of a cow during three consecutive days: the solid line represents the day detected as lame.

4.2 Experiments

We compare the algorithms described in Sect. 2 with the FBAT method presented Sect. 3. The code of the five methods comes from the github repository

of the original authors [2, 6]. The code of the FBAT method is available on the github repository <https://github.com/nicolas-wagner/FBAT>.

For BOSS we use the same parameters as in [2]: the word length $l = [8, 10, 12, 14, 16]$, the alphabet size $c = 4$ and the window size, $w = [10, 12, \dots, 36]$.

For FBAT we set the time window p to 24, the delay q to 12 and the number of samples s to 10000. We test all possibilities of the number of harmonics z , i.e. from 0 to 12 harmonics.

As in [6], the deep learning methods were run 10 times to train them with 10 different initializations of parameters. The results presented in this paper are the average over these 10 runs.

We use the same train and test data set for all methods tested. The train data set is composed of 1088 time series labelled as *normal* (negative) and 972 time series labelled as *abnormal* (positive). The test set is composed of 1408 *normal* time series and 4212 *abnormal*.

We define normal label as the negative class and abnormal label as the positive class. For each classifier it is possible to count the number of True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). We then calculated the overall accuracy as well as the precision and the recall for positive and the negative classes as follows:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \quad (3)$$

$$precision_- = \frac{TN}{TN + FN} \quad \text{and} \quad recall_- = \frac{TN}{TN + FP}, \quad (4)$$

$$precision_+ = \frac{TP}{TP + FP} \quad \text{and} \quad recall_+ = \frac{TP}{TP + FN}. \quad (5)$$

The accuracy is used as a single value to measure and compare the performance of the methods. The precision and the recall measured for each class help to understand the behavior of each classifier in detail. A high $recall_-$ (resp. $precision_-$) means that the majority of time series labelled as normal (resp. classified as normal) are classified as normal (resp. labelled as normal) and inversely for a high $recall_+$ (resp. $precision_+$).

The CPU time (in hour) is also retrieved from the experiments (training + test time). For the deep learning methods, a GPU mode is available so, FCN and ResNet were run on CPU and GPU. The first machine were composed of CPUs Intel Xeon 2.4GHz with 80 cores and 1 TB of RAM. The second were composed of CPUs Intel Xeon 2.4GHz with 10 cores and 62.5 GB of RAM with a GPU NVIDIA Quadro P5000 (16 GB of GDDR5 memory and 2560 cores). All algorithms were run in a sequential mode using only one core.

4.3 Results

The performances of all methods are summarized Table 1.

All of these methods are intended to be used by livestock farmers with a personal computer. Consequently, the CPU time is an important characteristic

Table 1. Results of all classifiers

		DTW	Hive-Cote	BOSS	FBAT	FCN	ResNet
Accuracy		0.54	0.63	0.72	0.60	0.66	0.67
Precision ₋		0.31	0.38	0.43	0.38	0.40	0.40
Recall ₋		0.68	0.73	0.36	0.90	0.73	0.71
Precision ₊		0.82	0.87	0.80	0.94	0.88	0.87
Recall ₊		0.49	0.60	0.84	0.50	0.64	0.65
Time (h)	CPU	1 h10	28 h	0 h38	0 h06	19 h28	16 h36
	GPU	–	–	–	–	1 h16	2 h13

to take into account and we notice that Hive-Cote has a too large CPU time (28 h) to be used in real conditions.

DTW obtains the worst performance in terms of accuracy (0.54) and its CPU time is rather high. DTW is faster than Hive-Cote but slower than BOSS and FBAT and similar to the GPU time of the deep learning models. Therefore, we estimate that DTW does not obtain enough satisfying results to be kept as a solution for this problem.

The performances of the neural networks are similar, excepted in terms of GPU time where FCN is almost two times faster than ResNet. They are a compromise between BOSS and FBAT in terms of $recall_+$ and $recall_-$. However, FCN and ResNet are considered as expensive solutions for livestock farmers since they need a GPU to be used in a real application.

BOSS produces the best accuracy results and obtains a low recall of the negative class. This means that among all time series labelled as *normal*, most of them are incorrectly classified as *abnormal*. If a farmer decides to use BOSS as a tool for detecting anomalies in dairy cows, he/she will then receive a high number of false alerts. These wrong detections may overshadow the correct ones and the method can become worthless. On the opposite, FBAT has the higher recall for the negative class. This would lead to a low number of false alerts for the farmer. The recall of abnormal days ($recall_+$) metric however decreases from 0.84 for BOSS to 0.50 for FBAT. As a matter of fact, a low $recall_+$ score may be due to the data set construction. Indeed, thanks to previous observations [15], we chose to label all time series included between two days before and one day after an anomaly as abnormal because the behavior of an animal can be disturbed shortly before and after clinical symptoms are detected. But all anomalies may not last for four consecutive days and this can lower the $recall_+$. We checked if for each anomaly, at least one of the four days is detected as abnormal by FBAT. The FBAT method detected at least one day among the four consecutive ones labelled as abnormal in 83% of the lameness cases, 61% of the acidosis and 100% of the estrus. These results seem adequate for an on-farm use and a test by farmers is necessary to decide if the $recall_-$ is satisfactory.

Another advantage of the FBAT method in a farm application is the threshold. Indeed, we proposed a solution to automatically set the threshold between

normal vs. abnormal time series. Nevertheless, the threshold can be adjusted. If a farmer thinks that the method does not detect enough anomalies, the threshold can be decreased. On the opposite, if the method detects too much false positive series, the threshold can be increased. Moreover, a threshold can be defined for each cow. If a cow is particularly insensitive to anomalies, its threshold can be decreased and inversely for a cow with a higher sensitivity.

The last advantage of FBAT is its CPU time, which is the best one of our experiments with 6 min. We also tested FBAT on personal computers (instead of big and expensive servers) and the CPU time didn't exceed 30 min.

5 Conclusion

The early detection of anomalies is very important for a farmer. Thanks to tools developed for precision livestock farming, it is possible to collect data in real time that can be analyzed by machine learning methods. In this study, we proposed a method based on Fourier transforms (FBAT) that we tested with the best algorithms and deep learning models available in the current literature for time series classification (BOSS, Hive-Cote, DTW, FCN and ResNet). The results showed that FBAT and BOSS are the two best solutions to solve the problem of anomaly detection in dairy cow activity. BOSS gives the best recall in the negative class whereas FBAT gives the best recall in positive class. They both obtain the best CPU time and they are both easy to implement. FBAT has the advantage to employ a threshold that can be adjusted to each cow. Testing these methods in real conditions, that is by farmers themselves, should help to choose the best method for the purpose. As other perspective, we propose to consider the labels as fuzzy. Indeed, the anomalies are detected when clinical signs are well visible, but it is reasonable to assume that anomalies gradually appear and disappear. We expect to increase the performances of the classifiers by better defining the labels. Finally, we plan to study the robustness of the algorithms to noisy labels. Indeed, label noise often occurs when humans are involved. In our application, caretakers are detecting and labeling anomalies but they easily have imperfect evidence.

Acknowledgment. This collaborative work was made possible thanks to the French Government IDEX-ISITE initiative 16-IDEX-0001 (CAP 20–25). The PhD grant for N. Wagner was provided by INRA and Université Clermont Auvergne. We thank the HERBIPOLE staff, B. Meunier, Y. Gaudron and M. Silberberg for data.

References

1. Adewumi, A.O., Akinyelu, A.A.: A survey of machine-learning and nature-inspired based credit card fraud detection techniques. *Int. J. Syst. Assurance Eng. Manag.* **8**(2), 937–953 (2017)
2. Bagnall, Anthony., Lines, Jason., Bostrom, Aaron., Large, James, Keogh, Eamonn: The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Min. Knowl. Disc.* **31**(3), 606–660 (2016). <https://doi.org/10.1007/s10618-016-0483-9>

3. Berkaya, S.K., et al.: A survey on ECG analysis. *Biomed. Signal Process. Control* **43**, 216–235 (2018)
4. Berndt, D.J., Clifford, J.: Using dynamic time warping to find patterns in time series. In: *KDD Workshop*, Seattle, WA, vol. 10, pp. 359–370 (1994)
5. Corizzo, R., Ceci, M., Japkowicz, N.: Anomaly detection and repair for accurate predictions in geo-distributed big data. *Big Data Res.* **16**, 18–35 (2019)
6. Fawaz, H.I., et al.: Deep learning for time series classification: a review. *Data Mining Knowl. Discov.* **33**(4), 917–963 (2019)
7. He et al.: Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778 (2016)
8. Lines, J., Taylor, S., Bagnall, A.: Hive-cote: the hierarchical vote collective of transformation-based ensembles for time series classification. In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 1041–1046. IEEE (2016)
9. 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 (2015)
10. Marwah, R., Cawkwell, F., Hennessy, D., Green, S.: Improved estimation of grassland biomass using machine learning and satellite data. In: *9th ECPLF 2019*, pp. 174–179. Teagasc (2019)
11. Mollenhors, H., de Haan, M., Oenema, J., Hoving-Bolink, A., Veerkamp, R., Kamphuis, C.: Machine learning to realize phosphate equilibrium at field level in dairy farming. In: *9th ECPLF 2019*, pp. 41–44. Teagasc (2019)
12. Munir, M., Siddiqui, S.A., Dengel, A., Ahmed, S.: Deepant: a deep learning approach for unsupervised anomaly detection in time series. *IEEE Access* **7**, 1991–2005 (2018)
13. Ruiz, E.V., Nolla, F.C., Segovia, H.R.: Is the DTW “distance” really a metric? An algorithm reducing the number of DTW comparisons in isolated word recognition. *Speech Commun.* **4**(4), 333–344 (1985)
14. Schäfer, P.: The BOSS is concerned with time series classification in the presence of noise. *Data Min. Knowl. Disc.* **29**(6), 1505–1530 (2015)
15. Veissier, I., Mialon, M.M., Sloth, K.H.: Early modification of the circadian organization of cow activity in relation to disease or estrus. *J. Dairy Sci.* **100**(5), 3969–3974 (2017)
16. Wang, J., Chen, Y., Hao, S., Peng, X., Hu, L.: Deep learning for sensor-based activity recognition: a survey. *Pattern Recogn. Lett.* **119**, 3–11 (2019)
17. Wang, Z., Yan, W., Oates, T.: Time series classification from scratch with deep neural networks: a strong baseline. In: *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 1578–1585. IEEE (2017)
18. Yang, Q., Wu, X.: 10 challenging problems in data mining research. *Int. J. Inf. Technol. Decision Making* **5**(04), 597–604 (2006)



Clustering Algorithm Consistency in Fixed Dimensional Spaces

Mieczysław Alojzy Kłopotek¹  and Robert Albert Kłopotek² 

¹ Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland
mieczyslaw.kłopotek@ipipan.waw.pl

² Faculty of Mathematics and Natural Sciences, School of Exact Sciences,
Cardinal Stefan Wyszyński University in Warsaw, Warsaw, Poland
r.kłopotek@uksw.edu.pl

Abstract. Kleinberg introduced an axiomatic system for clustering functions. Out of three axioms, he proposed two (scale invariance and consistency) are concerned with data transformations that should produce the same clustering under the same clustering function. The so-called consistency axiom provides the broadest range of transformations of the data set. Kleinberg claims that one of the most popular clustering algorithms, k -means does not have the property of consistency. We challenge this claim by pointing at invalid assumptions of his proof (infinite dimensionality) and show that in one dimension in Euclidean space the k -means algorithm has the consistency property. We also prove that in higher dimensional space, k -means is in fact inconsistent. This result is of practical importance when choosing testbeds for implementation of clustering algorithms while it tells under which circumstances clustering after consistency transformation shall return the same clusters.

Keywords: Cluster analysis · Consistency axiom · Consistency transformation · Fixed dimensional euclidean space consistency · k -Means algorithm

1 Introduction

In his heavily cited paper [4], Kleinberg introduced an axiomatic system for clustering functions. Out of three axioms, he proposed two are concerned with data transformations that should produce the same clustering under the same clustering function. We can speak here about “clustering preserving transformations” induced by these axioms. The so-called *consistency axiom*, mentioned below, shall be of interest to us here as it provides the broadest range of transformations.

Property 1. Let Γ be a partition of S , and d and d' two distance functions on S . We say that d' is a Γ -transformation of d if (a) for all $i, j \in S$ belonging to the same cluster of Γ , we have $d'(i, j) \leq d(i, j)$ and (b) for all $i, j \in S$ belonging to different clusters of Γ , we have $d'(i, j) \geq d(i, j)$. The clustering function f has

the consistency property if for each distance function d and its Γ -transformation d' the following holds: if $f(d) = \Gamma$, then $f(d') = \Gamma$

The validity or no-validity of any clustering preserving axiom for a given clustering function is of vital practical importance, as it may serve as a foundation for a testbed of the correctness of the function. Any modern software developing firm creates tests for its software in order to ensure its proper quality. Generators providing versatile test data are therefore of significance because they may detect errors unforeseen by the developers. Thus the consistency axiom may be used to generate new test data from existent one knowing a priori what the true result of clustering should be.

Note that Kleinberg [4, Section 2] defines clustering function as:

Definition 1. A clustering function is a function f that takes a distance function d on [set] S [of size $n \geq 2$] and returns a partition Γ of S . The sets in Γ will be called its clusters.

where the distance is understood by him as

Definition 2. With the set $S = \{1, 2, \dots, n\}$ [...] we define a distance function to be any function $d : S \times S \rightarrow \mathbb{R}$ such that for distinct $i, j \in S$ we have $d(i, j) \geq 0$, $d(i, j) = 0$ if and only if $i = j$, and $d(i, j) = d(j, i)$.

Note that his distance definition is not a Euclidean one and not even metric, as he stresses. This is of vital importance because based on this he formulates and proves a theorem (his Theorem 4.1)

Theorem 1. Theorem 4.1 from [4]. For every $k \geq 2$ and every function g [...] and for [data set size] n sufficiently large relative to k , the $(k; g)$ -centroid clustering function [this term encompassing k -means]¹ does not satisfy the Consistency property.

which we claim is wrong with respect to k -means for a number of reasons as we will show below. The reasons are:

- The objective function underlying k -means clustering is *not* obtained by setting $g(d) = d^2$ contrary to Kleinberg’s assumption (k -medoid is obtained).
- k -means always works in fixed-dimensional space while his proof relies on unlimited dimensional space.
- Unlimited dimensionality implies a serious software testing problem because the correctness of the algorithm cannot be established by testing as the number of tests is too vast.

¹ Kleinberg defined the centroid function as follows: for any natural number $k \geq 2$, and any continuous, non-decreasing, and unbounded function $g : \mathbb{R}^+ \rightarrow \mathbb{R}^+$, the $(k; g)$ -centroid clustering consists of: (1) choosing the set of k centroid points $T \subseteq S$ for which the objective function $\Delta_g^k(T) = \sum_{i \in S} g(d(i, T))$ is minimized, where $d(i, T) = \min_{j \in T} d(i, j)$. (2) a partition of S into k clusters is obtained by assigning each point to the element of T closest to it. He claims that the objective function underlying k -means clustering is obtained by setting $g(d) = d^2$.

- The consistency property holds for k -means in one-dimensional space.

The last result opens the problem of whether or not the consistency also holds for higher dimensions.

2 k -Means Algorithm

The popular clustering algorithm, k -means [5] strives to minimize the partition quality function (called also *partition cost function*)

$$J(U, M) = \sum_{i=1}^m \sum_{j=1}^k u_{ij} \|\mathbf{x}_i - \boldsymbol{\mu}_j\|^2 \quad (1)$$

where \mathbf{x}_i , $i = 1, \dots, m$ are the data points, M is the matrix of cluster centers $\boldsymbol{\mu}_j$, $j = 1, \dots, k$, and U is the cluster membership indicator matrix, consisting of entries u_{ij} , where u_{ij} is equal to 1 if among all of cluster (gravity) centers $\boldsymbol{\mu}_j$ is the closest to \mathbf{x}_i , and is 0 otherwise.

It can be rewritten in various ways while the following are of interest to us here. Let the partition $\Gamma = \{C_1, \dots, C_k\}$ be a partition of the data set onto k clusters C_1, \dots, C_k . Then

$$J(\Gamma) = \sum_{j=1}^k \sum_{\mathbf{x}_i \in C_j} \|\mathbf{x}_i - \boldsymbol{\mu}(C_j)\|^2 \quad (2)$$

where $\boldsymbol{\mu}(C) = \frac{1}{|C|} \sum_{\mathbf{x}_i \in C} \mathbf{x}_i$ is the gravity center of the cluster C . The above can be presented also as

$$J(\Gamma) = \frac{1}{2} \sum_{j=1}^k \frac{1}{|C_j|} \sum_{\mathbf{x}_i \in C_j} \sum_{\mathbf{x}_l \in C_j} \|\mathbf{x}_i - \mathbf{x}_l\|^2 \quad (3)$$

The problem of seeking the pair (U, M) minimizing J from Eq. (1) is called *k-means-problem*. This problem is known as NP-hard. We will call *k-means-ideal* such an algorithm that finds a pair (U, M) minimizing J from equation (1). Practical implementations of k -means usually find some local minima of $J()$. There exist various variants of this algorithm. For an overview of many of them, see e.g., [8]. An algorithm is said to be from the k -means family if it has the structure described by Algorithm 1. We will use a version with random initialization (randomly chosen initial seeds) as well as an artificial one initialized close to the true cluster center, which mimics k -means-ideal.

3 Kleinberg's Proof of Theorem 1 and Its Unlimited Dimensionality Deficiency

Kleinberg's proof, delimited to the case of $k = 2$ only, runs as follows: Consider a set of points $S = X \cup Y$ where X, Y are disjoint and $|X| = m$, $|Y| = \gamma m$, where

Data: the data points $\mathbf{x}_i, i = 1, \dots, m$, the required number of clusters k

Result: μ_1, \dots, μ_k

[1] Initialize k cluster gravity centers μ_1, \dots, μ_k ;

while a stop criterion (no change of cluster membership, or no sufficient improvement of the objective function, or exceeding some maximum number of iterations, or some other criterion) not reached **do**

[2] Assign each data element \mathbf{x}_i to the cluster C_j identified by the closest μ_j

;

[3] Update μ_j of each cluster C_j as the gravity center of the data elements in C_j ;

end

Algorithm 1: Structure of a practical algorithm from k -means-family

$\gamma > 0$ is “small”. $\forall_{i,j \in X} d(i,j) = r, \forall_{i,j \in Y} d(i,j) = \epsilon < r, \forall_{i \in X, j \in Y} d(i,j) = r + \delta$ where $\delta > 0$ and δ is “small”. By choosing $\gamma, \epsilon, r, \delta$ appropriately, the optimal choice of $k = 2$ centroids will consist of one point from X and one from Y . The resulting partition is $\Gamma = \{X, Y\}$. Let divide X into $X = X_0 \cup X_1$ with X_0, X_1 of equal cardinality. Reduce the distances so that $\forall_{c=1,2} \forall_{i,j \in X_c} d'(i,j) = r' < r$ and $d' = d$ otherwise. If r' is “sufficiently small”, then the optimal choice of two centroids will now consist of one point from each X_c , yielding a different partition of S . But d' is a Γ -transform of d so that a violation of consistency occurs. So far the proof of Kleinberg of the Theorem 1.

The proof cited above is a bit excentric because the clusters are heavily unbalanced (k -means tends to produce rather balanced clusters). Furthermore, the distance function is awkward because Kleinberg’s counter-example would require an embedding in a very high dimensional space, non-typical for k -means applications.

We claim in brief:

Theorem 2. *Kleinberg’s proof of [4, Theorem 4.1] that k -means ($k = 2$) is not consistent, is not valid in \mathbb{R}^p for data sets of cardinality $n > 2(p + 1)$.*

Proof. In terms of the concepts used in the Kleinberg’s proof, either the set X or the set Y is of cardinality $p + 2$ or higher. Kleinberg requires that distances between $p + 2$ points are all identical which is impossible in \mathbb{R}^p (only up to $p + 1$ points may be equidistant).

Furthermore Kleinberg’s minimized target function $\Delta_d^g(T) = \sum_{i \in S} g(d(i, T))$, where $d(i, T) = \min_{j \in T} d(i, j)$, differs significantly from the formula (3). For the original set X , the formula (3) would return $\frac{1}{2}(m - 1)r^2$, while Kleinberg’s would produce $(m - 1)r^2$. For a combination of a elements from X and b elements from Y in one cluster we get $\frac{a(a-1)r^2/2 + b(b-1)\epsilon^2/2 + ab(r+\delta)^2}{a+b}$ from (2) or the minimum of $(a-1)r^2 + b(r+\delta)^2$ and $(b-1)\epsilon^2 + a(r+\delta)^2$ for Kleinberg’s $\Delta_d^g(T)$. The discrepancy between these formulas is shown in Fig. 1.

We see immediately that

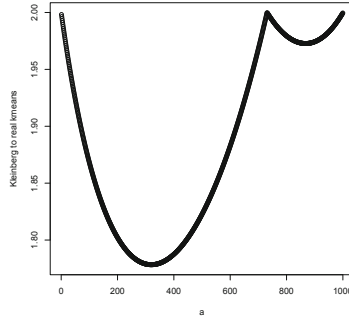


Fig. 1. Quotient of Kleinberg’s k-means target and the real k -means target.

Theorem 3. *Kleinberg’s target function does not match the real k -means target function.*

4 The Impact of Dimensionality of Consistency Property

Theorem 4. *k -means is consistent in one dimensional Euclidean space.*

Proof. Consider two alternative partitions in one dimensional space:

- the partition $\Gamma_1 = \{C_{1.}, \dots, C_{k.}\}$ which will be base for the Γ -transform
- and the competing partition $\Gamma_2 = \{C_{.1}, \dots, C_{.k'}\}$.

Due to the nature of k -means let each cluster of each partition after Γ -transform be represented as an interval not intersecting with any other cluster of the same partition. For Γ_1 , it holds before the transform; therefore, it holds afterward. Γ_2 shall be the competing optimal transform; therefore, it holds for sure afterward. We intend to demonstrate that under the Γ transformation, assuming that the actual partition is Γ_1 , the target function of k -means for Γ_1 will decrease not less than that for Γ_2 . For simplicity, assume that the indices of clusters grow with the growing value of the cluster center.

For this purpose assume that $C_{ij} = C_i \cap C_j$ are non-empty intersections of clusters $C_i \in \Gamma_1, C_j \in \Gamma_2$, of both partitions. Define $minind(C_i)$, resp. $maxind(C_i)$ as the minimal/maximal index j such that C_{ij} is not empty. The $Q(\Gamma_1)$ will be the sum of centered sums of squares over all C_{ij} plus the squared distances of centers of all C_{ij} to the center of C_i . times cardinality of C_{ij} .

$$\begin{aligned}
 Q(\Gamma_1) &= \sum_{C_i \in \Gamma_1} \sum_{j; C_{ij} \neq \emptyset} \left(|C_{ij}|(\mu(C_{ij}) - \mu(C_i))^2 + \sum_{x \in C_{ij}} (x - \mu(C_{ij}))^2 \right) \\
 &= \left(\sum_{i,j; C_{ij} \neq \emptyset} \sum_{x \in C_{ij}} (x - \mu(C_{ij}))^2 \right) + \left(\sum_{C_i \in \Gamma_1} \sum_{j; C_{ij} \neq \emptyset} |C_{ij}|(\mu(C_{ij}) - \mu(C_i))^2 \right)
 \end{aligned}$$

Please note that

$$\begin{aligned}
 \sum_{j;C_{ij}\neq\emptyset} |C_{ij}|(\mu(C_{ij}) - \mu(C_i))^2 &= \sum_{j;C_{ij}\neq\emptyset} |C_{ij}|(\mu(C_{ij}) - \sum_{j';C_{ij'}\neq\emptyset} \frac{|C_{ij'}|}{|C_i|} \mu(C_{ij'}))^2 \\
 &= \sum_{j;C_{ij}\neq\emptyset} |C_{ij}|(\sum_{j';C_{ij'}\neq\emptyset} (\frac{|C_{ij}|}{|C_i|} \mu(C_{ij}) - \frac{|C_{ij'}|}{|C_i|} \mu(C_{ij'})))^2 \\
 &= \sum_{j;C_{ij}\neq\emptyset} |C_{ij}|(\sum_{j';C_{ij'}\neq\emptyset} (\frac{|C_{ij'}|}{|C_i|} \mu(C_{ij}) - \mu(C_{ij'})))^2 \\
 &= 0.5 \sum_{j;C_{ij}\neq\emptyset} \sum_{j';C_{ij'}\neq\emptyset} \frac{|C_{ij}| \cdot |C_{ij'}|}{|C_i|} (\mu(C_{ij}) - \mu(C_{ij'}))^2
 \end{aligned}$$

The $Q(\Gamma_2)$ can be computed analogously, but let us follow a bit distinct path.

$$\begin{aligned}
 Q(\Gamma_2) &= \sum_{C_j \in \Gamma_2} \frac{1}{|C_j|} \sum_{x \in C_j} \sum_{y \in C_j} (x - y)^2 \\
 &= \sum_{C_j \in \Gamma_2} \frac{1}{|C_j|} \left(\left(\sum_{i;C_{ij}\neq\emptyset} \sum_{x \in C_{ij}} \sum_{y \in C_{ij}} (x - y)^2 \right) \right. \\
 &\quad \left. + \left(\sum_{i',i'';C_{i'j}\neq\emptyset, C_{i''j}\neq\emptyset} \sum_{x \in C_{i'j}, y \in C_{i''j}} (x - y)^2 \right) \right) \\
 &= \sum_{C_j \in \Gamma_2} \frac{1}{|C_j|} \left(\left(\sum_{i;C_{ij}\neq\emptyset} |C_{ij}| \sum_{x \in C_{ij}} (x - \mu(C_{ij}))^2 \right) \right. \\
 &\quad \left. + \left(\sum_{i',i'';C_{i'j}\neq\emptyset, C_{i''j}\neq\emptyset} \sum_{x \in C_{i'j}, y \in C_{i''j}} (x - y)^2 \right) \right) \\
 &= \left(\sum_{i,j;C_{ij}\neq\emptyset} \frac{|C_{ij}|}{|C_j|} \sum_{x \in C_{ij}} (x - \mu(C_{ij}))^2 \right) \\
 &\quad + \sum_{C_j \in \Gamma_2} \frac{1}{|C_j|} \left(\sum_{i',i'';i' \neq i'', C_{i'j}\neq\emptyset, C_{i''j}\neq\emptyset} \sum_{x \in C_{i'j}, y \in C_{i''j}} (x - y)^2 \right)
 \end{aligned}$$

Both summands of $Q(\Gamma_1)$, that is $\left(\sum_{i,j;C_{ij}\neq\emptyset} \sum_{x \in C_{ij}} (x - \mu(C_{ij}))^2 \right)$ and $\left(\sum_{C_i \in \Gamma_1} \sum_{j;C_{ij}\neq\emptyset} (|C_{ij}|(\mu(C_{ij}) - \mu(C_i))^2) \right)$ will decrease upon Γ_1 based consistency transformation. $(x - \mu(C_{ij}))^2$ decreases because the distance to each of elements of C_{ij} decreases as they are all in the same cluster C_i . Each $(\mu(C_{ij}) - \mu(C_{ij'}))^2$ decreases because all the elements constituting C_{ij} and $C_{ij'}$

belong to the same cluster C_i . Hereby there is always an extreme data point $P_{ij} \in C_{ij}$ separating it from $C_{ij'}$. As the points of both C_{ij} and $C_{ij'}$ get closer to P_{ij} consistency transformation, so the centers of both C_{ij} and $C_{ij'}$ will get closer to P_{ij} , so that they will move closer to each other. As summands of $Q(\Gamma_2)$ are concerned, the first, will also decrease upon consistency transformation. $\left(\sum_{i,j;C_{ij} \neq \emptyset} \frac{|C_{ij}|}{|C_{.j}|} \sum_{x \in C_{ij}} (x - \mu(C_{ij}))^2\right)$ but not by the same absolute value as $\left(\sum_{i,j;C_{ij} \neq \emptyset} \sum_{x \in C_{ij}} (x - \mu(C_{ij}))^2\right)$, because always $|C_{ij}| \leq |C_{.j}|$. But $\sum_{C_{.j} \in \Gamma_2} \frac{1}{|C_{.j}|} \left(\sum_{i',i'';i' \neq i'', C_{i'j} \neq \emptyset, C_{i''j} \neq \emptyset} \sum_{x \in C_{i'j}, y \in C_{i''j}} (x - y)^2\right)$ will increase because x, y stem from different clusters of Γ_1 . Therefore, if Γ_1 was the optimal clustering for k -means cost function prior to consistency transformation, it will remain so afterward.

But what about higher dimensions? Let us illustrate by a realistic example (balanced, in Euclidean space) that inconsistency of k -means in \mathbb{R}^m is a real problem - see Theorems 5 and 6.

Theorem 5. *k -means in 3D is not consistent.*

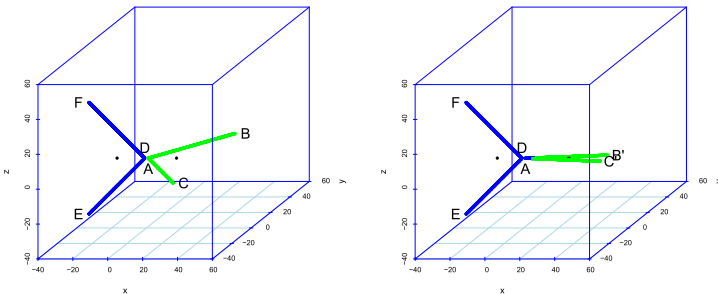


Fig. 2. Inconsistency of k -means in 3D Euclidean space. Left picture - data partition before consistency transform. Right picture - data partition after consistency transform.

Proof. Let A, B, C, D, E, F be points in three-dimensional space (see Fig. 2) with coordinates: $A(1, 0, 0)$, $B(33, 32, 0)$, $C(33, -32, 0)$, $D(-1, 0, 0)$, $E(-33, 0, -32)$, $F(-33, 0, 32)$. Let $S_{AB}, S_{AC}, S_{DE}, S_{DF}$ be sets of say 1000 points randomly uniformly distributed over line segments (except for endpoints) AB, AC, DE, EF resp. Let $X = S_{AB} \cup S_{AC} \cup S_{DE} \cup S_{EF}$. k -means with $k = 2$ applied to X yields a partition $\{S_{AB} \cup S_{AC}, S_{DE} \cup S_{DF}\}$. Let us perform a Γ transformation consisting in rotating line segments AB, BC around the point A in the plane spread by the first two coordinates towards the first coordinate axis so that the angle between

this axis and AB' and AC' is say one degree. Now the k -means with $k = 2$ yields a different partition, splitting line segments AB' and AC' .²

With this example not only *consistency violation* is shown, but also *refinement-consistency violation*. Not only in 3D, but also in higher dimensions. So what about the case of two dimensions - 2D?

Theorem 6. *k -means in 2D is not consistent.*

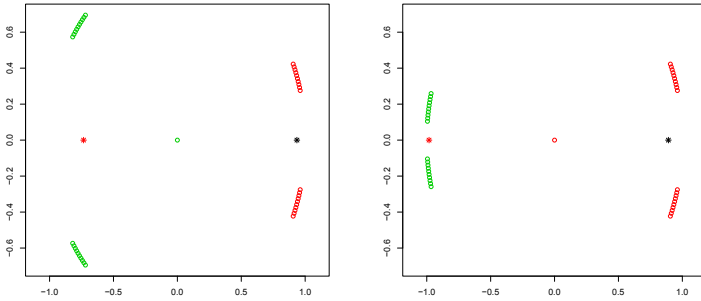


Fig. 3. Inconsistency of k -means in 2D Euclidean space. Left picture - data partition before consistency transform. Right picture - data partition after consistency transform. Cluster elements are marked with blue and green. Red points indicate cluster centers.

Proof. The proof of Theorem 6 uses a less realistic example than in Theorem 5, hence Theorem 5 was worthy considering in spite of the fact that it is implied by Theorem 6. Imagine a unit circle with data points arranged as follows (Fig. 3 left): one data point in the center, and the remaining points arranged on the circle with the following angular positions with respect to the circle center. Set $A = \{16^\circ, 17^\circ, \dots, 25^\circ, -16^\circ, -17^\circ, \dots, -25^\circ\}$. Set $B = \{136^\circ, 137^\circ, \dots, 145^\circ, -136^\circ, -137^\circ, \dots, -145^\circ\}$. k -means with $k = 2$ will merge points of the set B and the circle middle point as one cluster, and the set A as the other cluster. After a consistency-transformation (Fig. 3 right) let A turn to A' identical with A and let B change to B' $B' = \{165^\circ, 166^\circ, \dots, 174^\circ, -165^\circ, -166^\circ, \dots, -174^\circ\}$, while the point in the center of the circle remains in its position. Now k -means with $k = 2$ yields one cluster consisting of points of the set B' and the second cluster consisting of the circle middle point and the set A' . The center point of the circle switches the clusters upon consistency transformation.

² In a test run with 100 restarts, in the first case we got clusters of equal sizes, with cluster centers at $(17, 0, 0)$ and $(-17, 0, 0)$, (between_SS/total_SS = 40%) whereas after rotation we got clusters of sizes 1800, 2200 with centers at $(26, 0, 0)$, $(-15, 0, 0)$ (between_SS/total_SS = 59%).

5 Experiments

Experiments have been performed to check whether or not the Theorem 4 that denies Kleinberg’s findings for one-dimensional space really holds. Samples were generated from uniform distribution (sample size 100, 200, 400, 1000, 2000, 4000, 10000) for each $k = 2, \dots, \text{floor}(\sqrt{\text{samplesize}})$. Then the respective sample was clustered into k clusters ($k = 2, \dots, \text{floor}(\sqrt{\text{samplesize}})$) and k -means clustering (R package) was performed with $100k$ restarts. Subsequently, Γ transformation was performed where the distances within a cluster were decreased by a randomly chosen factor (a separate factor for each pair of neighboring data points), and at the same time the clusters were moved away so that the distance between cluster elements of distinct clusters is not decreased. Then k -means clustering was performed with $100k$ restarts in two variants. The first variant was with random initialization. The second variant was with the initialization of the midpoint of the original (rescaled) cluster interval. Additionally, for control purposes, the original samples were reclustered. The number of partitions was counted for which errors in restoring the original clustering was observed. Experiments were repeated ten times. Table 1 presents the average results obtained.

Table 1. Validation of the Theorem 4.

Sample size	100	200	400	1000	2000	4000	10000
Max. k	10	14	20	31	44	63	100
Errors variant 1	0	0	0	2.4	10.2	21.5	62.4
Errors variant 2	0	0	0	0	0	0.5	2.0
Errors reclustered	0	0	2.3	14.1	30.2	49.5	86.2

In this table, looking at the errors for variant 1, we see that with the increasing sample size (and hence increasing maximum of k), more errors are committed. This contrasts with the variant 2 where the number of errors is negligible. The second variant differs from the first in that seeds are distributed so that there is one in each intrinsic cluster.

Clearly the Theorem 4 holds (as visible from the variant 2). At the same time, however, the table shows that k -means with random initialization is unable to initialize properly for a larger number k of clusters in spite of a large number of restarts (variant 1). This is confirmed by the experiments with reclustered original data.

This study also shows how a test data generator may work when comparing variants of k -means algorithm (for one-dimensional data).

6 Conclusions

In this paper, we have provided a definite answer to the problem of whether or not k -means algorithm possesses the consistency property. The answer is negative

except for one-dimensional space. Settling this problem was necessary because the proof of Kleinberg of this property was inappropriate for real application areas of k -means that it is a fixed-dimensional Euclidean space. The result precludes usage of consistency axiom as a generator of test examples for k -means clustering function (except for one-dimensional data) and implies the need to seek alternatives.

Kleinberg's consistency was subject of strong criticism and new variants were proposed like Monotonic consistency [6] or MST-consistency [9]. See also criticism in [2, 3]. The mentioned new definitions of consistency are apparently restrictions of Γ -consistency, and therefore the Theorem 4 would be valid. The Monotonic consistency seems not to impose restrictions on Kleinberg's proof on k -means violating consistency. Therefore in those cases, the consistency of k -means under higher dimensionality needs to be investigated. Note that we have also challenged the result [7], who claims that Kleinberg's consistency may be achieved by k -means with random initialization (see our Theorem 5). The shift of axioms from clustering function to quality measure [1] was suggested to the problems with consistency, but this approach fails to tell what the outcome of clustering should be, which is not useful for the mentioned test generator application.

References

1. Ben-David, S., Ackerman, M.: Measures of clustering quality: a working set of axioms for clustering. In: Proceedings Advances in Neural Information Processing Systems, vol. 21, pp. 121–128 (2008)
2. Carlsson, G., Mémoli, F.: Characterization, stability and convergence of hierarchical clustering methods. *J. Mach. Learn. Res.* **11**, 1425–1470 (2010)
3. Correa-Morrison, J.: An indication of unification for different clustering approaches. *Pattern Recogn.* **46**, 2548–2561 (2013)
4. Kleinberg, J.: An impossibility theorem for clustering. In: Proceedings NIPS, vol. 2002, pp. 446–453 (2002). <http://books.nips.cc/papers/files/nips15/LT17.pdf>
5. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press (1967)
6. Strazzeri, F., Sánchez-García, R.J.: Morse theory and an impossibility theorem for graph clustering (2018). <https://arxiv.org/abs/1806.06142>
7. Wei, J.H.: Two examples to show how k -means reaches richness and consistency. *DEStech Trans. Comput. Sci. Eng.* (2017). <https://doi.org/10.12783/dtcse/aita2017/16001>
8. Wierzchoń, S., Kłopotek, M.: Modern clustering algorithms. *Stud. Big Data* **34** (2018)
9. Zadeh, R.: Towards a principled theory of clustering (2010). <http://stanford.edu/rezab/papers/principled.pdf>



Estimating the Importance of Relational Features by Using Gradient Boosting

Matej Petković^{1,2} , Michelangelo Ceci^{1,3} , Kristian Kersting⁴ ,
and Sašo Džeroski^{1,2} 

¹ Jozef Stefan Institute, Jamova 39, Ljubljana, Slovenia

² Jozef Stefan Postgraduate School, Jamova 39, Ljubljana, Slovenia

{matej.petkovic,saso.dzeroski}@ijs.si

³ Università degli Studi di Bari Aldo Moro, via E. Orabona 4, Bari, Italy

michelangelo.ceci@uniba.it

⁴ CS Department, TU Darmstadt, Hochschulstrasse 1, Darmstadt, Germany

kersting@cs.tu-darmstadt.de

Abstract. With data becoming more and more complex, the standard tabular data format often does not suffice to represent datasets. Richer representations, such as relational ones, are needed. However, a relational representation opens a much larger space of possible descriptors (features) of the examples that are to be classified. Consequently, it is important to assess which features are relevant (and to what extent) for predicting the target. In this work, we propose a novel relational feature ranking method that is based on our novel version of gradient-boosted relational trees and extends the Genie3 score towards relational data. By running the algorithm on six well-known benchmark problems, we show that it yields meaningful feature rankings, provided that the underlying classifier can learn the target concept successfully.

Keywords: Feature ranking · Relational trees · Gradient boosting

1 Introduction

One of the most frequently addressed tasks of machine learning is predictive modeling, where the goal is to build a model by using a set of known examples, which are given as pairs of target values and vectors of feature values. The model should generalize well to previously unseen combinations of feature values and accurately predict the corresponding target value. In this paper, we limit ourselves to classification, i.e., to the case when the target can take one of finitely many nominal values. For example, when modeling the genre of a movie, the possible values might be **thriller**, **drama**, **comedy** and **action**.

In the simplest and most common case, the data comes as a single table where rows correspond to examples and columns to features, including the target.

This is financially supported by the Slovenian Research Agency (grants P2-0103, N2-0128, and a young researcher grant to MP).

© Springer Nature Switzerland AG 2020

D. Helic et al. (Eds.): ISMIS 2020, LNAI 12117, pp. 362–371, 2020.

https://doi.org/10.1007/978-3-030-59491-6_34

However, when predicting the genre of a movie, it might be beneficial to not only know the properties of the movie (i.e., the feature values), but also to have access to the properties of ratings of the movie (e.g., number of stars and date) on some website. As a consequence, the data is now represented by two tables: one describing the movies and the other one describing the ratings. A link between them might then be the relation *belongsTo* (rating, movie) that tells which movie a given rating rates. The actual `movies` dataset used in our experiments (shown in Fig. 1) contains five additional tables and four relations among them.

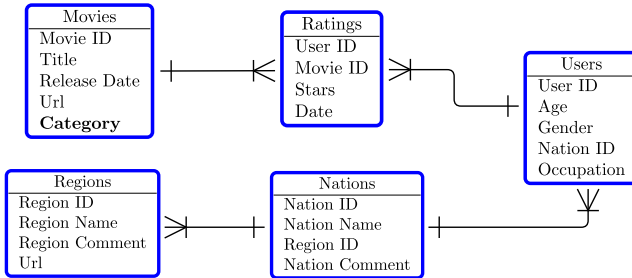


Fig. 1. The `movies` dataset consists of 5 tables and 4 relations among them. Single and multiple ends on the connections among the tables represent one-to- (or -to-one) and many-to- (or -to-many) relations, respectively. The task at hand is to predict the property `Category` (genre) of a movie (shown in bold).

In the era of abundance of data, such complex datasets are more and more frequent in various domains, e.g., sports statistics, businesses, epidemiology [14], among others. In biology, for instance, protein networks [18] encode complex data about proteins and their functions, together with different protein-protein interactions. Concrete examples are given in Sect. 5. In all these cases, relational data are very much of interest because they offer a powerful representation language where a broad spectrum of predictive modeling tasks can be applied. A notable example is link prediction where the goal is to predict whether two examples are in a given relation or not [4], say, two researchers are co-authors of a paper [16]. It is also possible to predict links of multiple types [3]. Another prominent task is classification. For example, movies can be categorized into different genres. Again, predicting multiple targets is possible [12, 14].

The main focus of the present paper, however, is not predictive modeling, but rather feature ranking. In the simplest case when the data is given as a single table, the task of feature ranking is to estimate the importance of each descriptive feature when predicting the target value. The features are then ranked with respect to their importance. A possible motivation for performing feature ranking is dimensionality reduction. Discarding the less relevant features from the dataset (before proceeding to predictive modeling) results in lower time and space complexity of learning the models, which may also be more accurate since a lower number of features reduces overfitting. Moreover, feature ranking can

explain black box models such as ensembles of decision trees [1, 5] or neural networks, which may be of crucial importance in domains such as medicine.

Feature ranking on relational data, however, is not a trivial task. In fact, with two tables or more (and relations between them) at hand, the notion of a feature is a more general concept. It can refer to a relation between tables or to a descriptive attribute, which is part of the description of the objects in a given table. Furthermore, the existing relations and descriptive attributes can be combined into new ones. For example, the schema in Fig. 1 implicitly defines a feature whose value for a given movie is the name of the nation which contributes the highest number of ratings for the movie. Consequently, the feature space can be extremely large, and discovering only the most relevant features for a given predictive modeling task might have a high value. A feature ranking algorithm can estimate the importance of the features explicitly given (such as release date in the `movies` dataset), or it can use some heuristic to construct new features from the ones existing in the descriptive relations.

In this paper, we propose a feature ranking algorithm that focuses on the second approach, i.e., heuristically introduces new features. The novel heuristic for building relational trees starts the search for relevant features at those which are explicitly given, and gradually proceeds towards more and more complex ones if needed. To improve the quality (and stability) of feature rankings, we use gradient boosting ensembles [5] instead of a single tree. Once the ensemble is built, feature ranking is seamlessly computed out of it by using the adaptation of the Genie3 score [9].

The remainder of the paper is organized as follows. Section 2 explains the necessary background and notation, Sect. 3 reviews related work, Sect. 4 introduces our extension of gradient boosting to relational data and our novel feature ranking algorithm, Sect. 5 gives the experimental setup, while Sect. 6 the results are discussed. Finally, Sect. 7 concludes the work.

2 Background on Relational Predictive Modeling

Let \mathcal{X} be a generic domain that identifies an object type and x be an example (instance) of such type. Every object $x \in \mathcal{X}$ is represented in terms of its ID (object identifier) and a list of attribute values. A relation r of arity $t \geq 1$ is defined as a subset of a Cartesian product of the domains \mathcal{X}_i of relation's arguments X_i , $1 \leq i \leq t$. The fact that $x_1 \in \mathcal{X}_1, \dots, x_t \in \mathcal{X}_t$ are in a relation r , is denoted either as $(x_1, \dots, x_t) \in r$ or $r(x_1, \dots, x_t)$.

If $t = 1$ the relation describes a property of a single object, e.g., *isMale*(user). If $t = 2$, the relation is said to be binary. A binary relation is one-to-many if each $x_1 \in \mathcal{X}_1$ is in a relation with *at least* one $x_2 \in \mathcal{X}_2$, and each x_2 is in a relation with *exactly* one x_1 . The other three cases (one-to-one, many-to-one and many-to-many) are defined analogously.

The goal of the relational classification task is thus to learn a model that uses descriptive relations to predict the target relation $r(X_0, Y)$ where the first component corresponds to the example to be classified and the last component

corresponds to the target values. For the link prediction task, the target relation is of form $r(X_0^1, X_0^2, Y)$ where both X_0^1 and X_0^2 correspond to examples and the last component corresponds to the target values. Thus, a relational representation of the data elegantly unifies different classification tasks into the same framework. However, in this work, we focus on the $r(X_0, Y)$ target relations only.

A dataset is typically stored as a group of tables and relations between them. For example, the `movies` dataset in Fig. 1 contains five tables that describe five different types of objects: movies, ratings, users, nations, and regions. Every movie is given as a 5-tuple, consisting of movie ID, title, release date, url, and category. Similarly, ratings are given as 4-tuples of user ID (the author of the rating), movie ID (the movie that rating refers to), stars and date.

Since both the first component of movies and the second component of ratings are of the same type (movie ID), the dataset also implicitly defines the one-to-many relation via which we can find, for example, all ratings of a given movie. Similar links exist between users and ratings (connected via user ID columns), between users and nations, and between nations and regions. Should there be a relation among users, e.g., *isARelative*, we would need an additional table, e.g., *relatives*, that would contain two columns (both of the type user ID) and would explicitly list all the pairs of relatives.

Prior to applying our algorithm (and also the majority of those discussed in the related work), the data at hand should be converted into a pure relational representation, coherent with the formal description above, where all facts are given as relation elements $r(x_1, \dots, x_t)$, e.g., *gender*(Ana, female).

3 Related Work

Since our feature ranking is embedded into a classifier [7], it is closely related to relational ensemble techniques. Gradient boosting of relational trees is proposed in [11]. It takes relations (given as sets of tuples) as the input and builds gradient-boosted regression trees. As usual, multi-class problems demand 1-hot encoding of the target variable, which converts the original dataset into a series of 1-versus-all classification problems, and an ensemble is built for each of them separately. In turn, a tree induction in an ensemble bases on the TILDE learner [17] and its predecessor FOIL [13], which results in two possible limitations of the method.

First, it allows only for the nominal descriptive features, thus the numeric ones (e.g., age of a user) should first be discretized into bins which results in the loss of ordering of values. The second possible limitation are candidate tests in the internal nodes of the trees. Without loss of generality, we assume that the variable X_0 that corresponds to the example ID whose target values is to be predicted, always appears as the first component of a relation r . In this case, the candidate splits are of two types. First, a split can be a conjunction of predicates

$$r_1(X_0, x_1^2, \dots) \wedge \dots \wedge r_j(X_0, x_j^2, \dots) \wedge r_{j+1}(x_{j+1}^1, x_{j+1}^2, \dots) \wedge \dots \wedge r_\ell(x_\ell^1, x_\ell^2, \dots)$$

where $\ell \geq 1$, and x_i^k is the value of the variable at position k for relation r_i . Second, some of the variables X_i^k may not be grounded yet (i.e., their value is

not determined). In that case, a split is of the form

$$\exists X_{i_1}^{k_1} \dots \exists X_{i_n}^{k_n} : r_1(X_0, x_1^2, \dots) \wedge \dots \wedge r_\ell(x_\ell^1, x_\ell^2, \dots) \quad (1)$$

where n is the number of non-grounded variables. When the actual example ID x_0 reaches a split, X_0 takes its value, the split is evaluated, and the example follows the YES or NO branch accordingly. That means that having splits like *Is the average age of users that rated a movie, larger than 60?* is not possible. That was overcome by introducing aggregates into TILDE [17]. There, also constrained aggregation is possible, e.g., *Is the average age of users that have contributed at least 5 ratings in total, and have rated the given movie, larger than 60?* For the exact formulation of the possible split tests, we refer the reader to [17] where the extension of the method to relational random forests is described.

Regarding relational feature ranking methods, there is only the FARS method [8], which belongs to the group of filters [7], i.e., no predictive model is needed for computing the ranking. As such, it cannot be used for explaining the decisions of classifiers. It is suitable for classification datasets and is based on propagation of the class values from the table that contains the target attribute to the other tables. The method supports neither estimation of numeric attributes nor the estimation of implicitly defined features.

4 Our Method

Here, we describe our two contributions. We first present the proposed test split generation for relational trees (and boosting ensembles). Afterward, the proposed feature ranking approach is introduced.

4.1 Relational Gradient Boosting with Aggregates

Relational trees are built with the standard top-down induction procedure [2] whose main part is greedily finding the optimal test (according to a heuristic h that is based on an impurity measure, e.g., Gini index [2]) that splits the data into two subgroups. The point at which relational tree induction differs from the standard one is how the candidate splits are created. Indeed, one option is using TILDE with aggregates. However, also from the feature ranking perspective, we find the following feature-value back-propagation splits more appropriate, since by doing so (in contrast to TILDE), we can directly link the values of a given feature to the given example ID, no matter which table is the feature present in.

Consider Fig. 2, which depicts our candidate test generation. Each test is generated in two stages. First (as shown in Fig. 2a), we start from an example ID x_0 , and follow any relation r_1 where x_0 can appear in. The group g_1 of all tuples $\mathbf{x}_1^i \in r_1$, which x_0 is part of, is thus found. Then, for each of the tuples \mathbf{x}_1^i , we recursively repeat the search from \mathbf{x}_1^i , thus finding the group of examples g_2^i by following any relation r_2 that shares at least one input domain \mathcal{X} with relation r_1 . The search is finished after at most ℓ steps which is a user-defined

parameter. In Fig. 2a, we have $\ell = 2$. For example, following the schema of the movie data set in Fig. 1, we might start from $x_0 = \text{titanic}$, and find the group g_1 of all pairs $\mathbf{x}_1^i = (\text{titanic}, \text{ratingID}^i)$. For each such pair, a (singleton) group g_2^i of all pairs $(\text{ratingID}^i, \text{stars})$ is found.

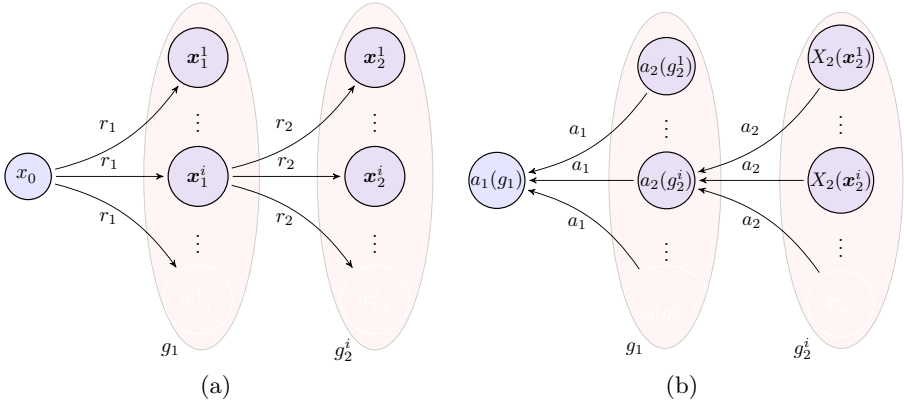


Fig. 2. Candidate test generation: finding related tuples by following descriptive relations (a), and back-propagation of the feature values by aggregation (b).

After the search is finished, the back-propagation of feature values by aggregation starts, by choosing one of the variables that were introduced in the last step of the search. Let this be X_ℓ , i.e., X_2 . Its type defines possible starting aggregates. We can always use *count* or *countUnique*. Additionally, we can use *max*, *min*, *mean*, and *sum* if X_2 is numeric, and *mode* if X_2 is nominal. The data type returned by the first aggregate in turn defines the possible options for the next one in the chain. In general, we proceed from right to left (Fig. 2b).

By using aggregate a_2 , we aggregate every group g_2^i over X_2 . Following the example above, the variable at hand would be the number of stars. Then, the tuple $\mathbf{x}_1^i \in g_1$ is effectively replaced by the aggregated value $a_2(g_2^i)$. After this is done for all tuples \mathbf{x}_1^i , the group g_1 is similarly aggregated into the final value $a_1(g_1)$. If we set a_2 to *mean* and a_1 to *max*, in the above example, we compute the maximal rating of a movie (the mean of a single value is the value itself).

Finally, the procedure for generating candidate splits proceeds to finding the optimal threshold ϑ the aggregated values are compared against, and chooses the best test among the candidates. This covers also the existentially quantified split tests in Eq. (1) if $\vartheta = 0$ and the aggregates are set to *count*. This motivates the idea to allow the algorithm to continue the search for a good split in the YES-child at any of the steps from its parent node.

Therefore, the evaluation of the tests becomes more time-consuming deeper in the tree, so gradient boosting [5] where the trees are shallower is more efficient than, e.g., bagging [1] where bias-variance decomposition of the error reveals that trees should be grown to a full depth.

Using aggregates is necessary since the preliminary experiments (not part of this work) show that this increases the expressiveness of the splits which reflects in substantially improved predictive power of the models.

4.2 Feature Ranking

Now, we are ready to introduce our novel relational feature ranking. Let (R, A, ϑ) be a triplet denoting a test in a node \mathcal{N} in a tree \mathcal{T} , where R and A are lists of relations and aggregates used in the test. The size s of a test is defined as $s = |R|$ ($= |A|$). Let $E(\mathcal{N})$ be the set of examples that reach the node \mathcal{N} , and $h(\mathcal{N})$ the heuristic value of the split in \mathcal{N} [2]. We write $r \in \mathcal{N}$ if $r \in R$. Then, a natural extension of the Genie3 score [9] for the already existing relations r is

$$\text{importance}(r) = \sum_{\mathcal{T}} \sum_{\mathcal{N} \in \mathcal{T}} \frac{\mathbf{1}[r \in \mathcal{N}]}{s(\mathcal{N})} h(\mathcal{N}) |E(\mathcal{N})|, \quad (2)$$

where $\mathbf{1}$ is the indicator function. The definitions says that all relations that appear in a given node are rewarded equally and proportionally to the heuristic value. The term $|E(\mathcal{N})|$ assures that relations that appear higher in a tree (and influence more examples) receive bigger award.

Please note that Eq. (2) is naturally extended to (parts of) lists R of relations by summing up the importances of their atomic parts. Note also that the relations and combinations thereof that do not appear in the ensemble, have the importance score with the value 0.

5 Experimental Setting

In order to investigate the performance of our relational feature ranking methods, we computed feature rankings for six well-known datasets. In addition to the `movie` dataset (shown in Fig. 1), these were: `basket` [10] (basketball players, coaches, teams, etc.), `IMDB` [6] (movies, actors, directors, etc.), `Stack` [15] (user posts, users and comments), and `Yelp` [19] (different businesses, their reviews, users etc.). Additional statistics for the data are given in Table 1.

In our experiments, the quality of the underlying predictive model will be used as a proxy for the quality of the ranking, thus 10-fold cross-validation (CV) is performed. For each training set, internal 3-fold CV was performed to tune the boosting parameters via grid-search. The parameters (and their possible values) that were optimized are shrinkage ($\{0.05, 0.2, 0.4, 0.6\}$), proportion of chosen examples ($\{0.6, 0.8, 1.0\}$), proportion of the evaluated tests in a node ($\{0.2, 0.4, \dots, 1.0, \text{sqrt}\}$), and maximal depth of trees ($\{2, 4, 6, 8\}$). Ensemble size was set to 50 and the size of splits was $\ell \leq 2$.

6 Results and Discussion

The experimental results are summarized in Fig. 3. It shows the feature importances, averaged over the 10-folds of CV, and the three most relevant features,

for each dataset. We observe two qualitatively different results: for the datasets **basket**, **Stack** and **UWCSE** (and to some extent **Yelp**), it is evident that feature importance scores have mostly converged to their final values, meaning that the Gini heuristic in the splits goes down to 0.0 and the trees with higher indices do not influence the ranking or predictions of the model. This is confirmed by the accuracy of the corresponding models: 0.98 (**basket**), 0.95 (**Stack**), 0.83 (**basket**) and 0.88 (**Yelp**). Since accuracy on the training sets are even higher, this means that only a few training examples are being miss-classified and the target values at 50-th iteration are mostly close to 0.0.

The two most prominent members of the second class of results are **IMDB** and **movie** where the feature importance scores are still noticeably growing. On the other hand, the order of the features is mostly fixed, so trees are similar to each other, but unable to fully solve the predictive problem. Indeed, the accuracy of the corresponding two models is 0.63 (**IMDB**) and 0.57 (**movie**) which is closer to the default accuracy than in the previous cases (see Table 1).

The next observation is that for all six datasets, a group of 1–3 most important features is established. The difference between this group and the other features is most notable for the **UWCSE** dataset. Here, the goal is to predict, which discipline a given course belongs to, and the most important relation is *taughtBy*(course, person, session). This is not surprising as it is also the link to the *advisedByDiscipline*(person, person, discipline), ranked second. Since professors typically work only in one discipline, the discipline a professor advises someone in, is likely equal to a discipline that the professor teaches.

A similar difference between the top feature and the others is visible also in the cases of the **Yelp** dataset (as well as for **basketball**). In the case of **Yelp**, the goal is to predict the category of a business (e.g., restaurant, Health&Medical etc.), and counting in how many tuples of the *attributes* relation a business appears, is a good indication of the target value. For example, restaurants tend to have a lot of attributes such as classy, hipster, romantic, dessert etc. whereas Health&Medical places are sometimes described only by **ByAppointmentOnly**.

Table 1. Data characteristics: number of tables, number of relations in the final representation, sum of relation sizes (number of descriptive facts), number of target facts, number of classes, and the proportion of the majority class.

Dataset	Tables	Relations	Descriptive facts	Target facts	Classes	Majority class
basket	9	118	630038	95	2	0.70
IMDB	21	57	614662	8816	4	0.58
movie	5	16	183469	1422	4	0.51
Stack	7	52	383040	5855	5	0.36
UWCSE	12	15	1961	115	5	0.25
Yelp	9	51	3348181	24959	4	0.57

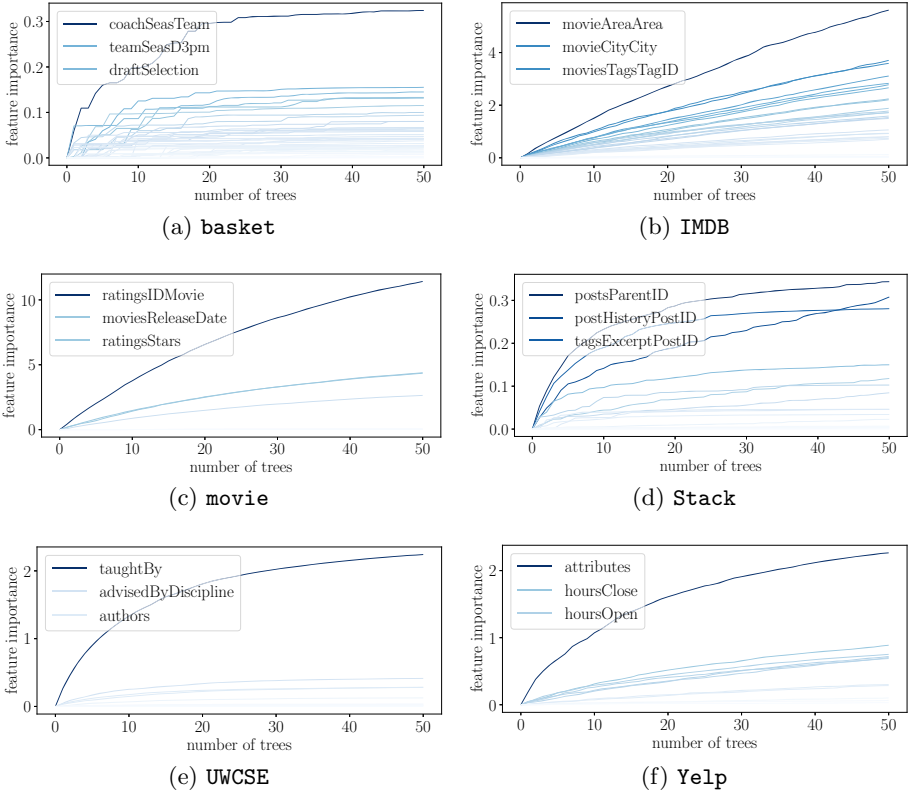


Fig. 3. Development of feature importance values with the number of trees, for different datasets. Every line corresponds to a feature that is present in an ensemble. It’s color corresponds to the final feature importance at 50 trees. Additionally, the three most relevant features are listed.

In the cases where the models are not that accurate, feature ranking is a way of seeing whether the model overfits and if some relations should be excluded from the descriptive space. This happens in the case of the IMDB dataset where the goal is to predict the genre of a movie but the most relevant feature is the area where the movie was taken.

7 Conclusions and Future Work

We have proposed an adaptation of the Genie3 feature ranking to relational data, using our adaptation of gradient boosting as the underlying ensemble, and evaluated its appropriateness empirically. The main motivation for choosing boosting was that the trees learned in boosting can be quite shallow as compared to those learned in bagging. However, parameter-tuning for boosted relational trees takes a considerable amount of time, so we plan to extend the proposed

approach to other ensemble methods, such as bagging and random forests (and parallelize them). Also, the feature-ranking-motivated definition of the possible split tests will be evaluated in a predictive modeling scenario. Finally, we plan to compare the different relational ensembles against each other and against other learners.

References

1. Breiman, L.: Bagging predictors. *Mach. Learn.* **24**(2), 123–140 (1996)
2. Breiman, L., Friedman, J., Olshen, R., Stone, C.J.: *Classification and Regression Trees*. Chapman & Hall/CRC, Boca Raton (1984)
3. Davis, D., Lichtenwalter, R., Chawla, N.V.: Multi-relational link prediction in heterogeneous information networks. In: 2011 International Conference on Advances in Social Networks Analysis and Mining, pp. 281–288 (2011)
4. Dong, Y., et al.: Link prediction and recommendation across heterogeneous social networks. In: 2012 IEEE 12th International Conference on Data Mining, pp. 181–190 (2012)
5. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. *Ann. Stat.* **29**(5), 1189–1232 (2001)
6. GroupLens Research: Imdb dataset. <https://grouplens.org/datasets/hetrec-2011/>
7. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. *J. Mach. Learn. Res.* **3**, 1157–1182 (2003)
8. He, J., Liu, H., Hu, B., Du, X., Wang, P.: Selecting effective features and relations for efficient multi-relational classification. *Comput. Intell.* **26**, 258–281 (2010)
9. Huynh-Thu, V.A., Irrthum, A., Wehenkel, L., Geurts, P.: Inferring regulatory networks from expression data using tree-based methods. *PLOS ONE* **5**(9), 1–10 (2010). <https://doi.org/10.1371/journal.pone.0012776>
10. Moore, A.W.: Basket dataset. <http://www.cs.cmu.edu/~awm/10701/project/data.html>
11. Natarajan, S., Kersting, K., Khot, T., Shavlik, J.: *Boosted Statistical Relational Learners*. SCS. Springer, Cham (2014). <https://doi.org/10.1007/978-3-319-13644-8>
12. Pio, G., Serafino, F., Malerba, D., Ceci, M.: Multi-type clustering and classification from heterogeneous networks. *Inf. Sci.* **425**, 107–126 (2018)
13. Quinlan, J.R.: Boosting first-order learning. In: Arikawa, S., Sharma, A.K. (eds.) ALT 1996. LNCS, vol. 1160, pp. 143–155. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-61863-5_42
14. Serafino, F., Pio, G., Ceci, M.: Ensemble learning for multi-type classification in heterogeneous networks. *IEEE Trans. Knowl. Data Eng.* **30**(12), 2326–2339 (2018)
15. Stack Exchange: Stack dataset. <https://archive.org/details/stackexchange>
16. Sun, Y., Barber, R., Gupta, M., Aggarwal, C.C., Han, J.: Co-author relationship prediction in heterogeneous bibliographic networks. In: 2011 International Conference on Advances in Social Networks Analysis and Mining, pp. 121–128 (2011)
17. Vens, C.: Complex aggregates in relational learning. Ph.D. thesis, Faculteit Ingenieurswetenschappen, Katholieke Universiteit Leuven (2007)
18. Škrlj, B., Kralj, J., Lavrač, N.: Targeted end-to-end knowledge graph decomposition. In: Riguzzi, F., Bellodi, E., Zese, R. (eds.) ILP 2018. LNCS (LNAI), vol. 11105, pp. 157–171. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99960-9_10
19. Yelp: Yelp dataset. www.yelp.com/dataset_challenge



Multi-objective Discrete Moth-Flame Optimization for Complex Network Clustering

Xingjian Liu¹, Fan Zhang¹, Xianghua Li¹, Chao Gao¹, and Jiming Liu²(✉)

¹ College of Computer and Information Science, Southwest University, Chongqing, China

² Department of Computer Science, Hong Kong Baptist University, Kowloon, Hong Kong
jiming@comp.hkbu.edu.hk

Abstract. Complex network clustering has been extensively studied in recent years, mostly through optimization approaches. In such approaches, the multi-objective optimization methods have been shown to be capable of overcoming the limitations (e.g., instability) of the single-objective methods. Nevertheless, such methods suffer from the shortcoming of incapability of maintaining a good tradeoff between exploration and exploitation, that is, to find better solutions based on the good ones obtained so far. In this paper, we present a new nature-inspired heuristic optimization method, called multi-objective discrete moth-flame optimization (DMFO) method, which achieves such a tradeoff. We describe the detailed algorithm of DMFO that utilizes the Tchebycheff decomposition approach with an l_2 -norm constraint on the direction vector (2-Tch). Furthermore, we show the experimental results on synthetic and several real-world networks that verify that the proposed DMFO and the algorithm are both effective and promising for tackling the task of complex network clustering.

Keywords: Complex network clustering · Multi-objective optimization · Discrete moth-flame optimization · Decomposition

1 Introduction

Complex networks are ubiquitous in the real world, some examples of which include biological networks [1], social networks [2], and transportation networks [3]. Computationally speaking, a complex network can be viewed as a graph in which nodes and links correspond, respectively, to the objects and the relationships between them in a certain complex system. Of great interest in analyzing a complex network is to detect its intrinsic cluster structure, such as those found in social networks [2]. Here, a cluster refers to a subset of closely related or linked nodes, while their connections with the nodes in other communities are sparse [4]. The network clustering aims to discover such community structures

in a network, which can also be viewed as a task of dividing the network into various subsets of nodes according to their characteristics.

Network clustering, in essence, can be modeled as an optimization problem [5]. Several optimization algorithms for solving this problem have been proposed in recent years. Traditional single-objective methods first select an optimal objective (such as the modularity Q [6] or the community score [7]) and apply different strategies to optimize such an objective. For example, GANet optimizes the community score using a genetic algorithm for network clustering in social networks [7]. DCRO optimizes the modularity by simulating a chemical reaction process in order to achieve a better clustering result [8]. Generally, one of the key short-comings in the above-mentioned methods lies in that they consider only a single structural attribute in the formulation of their objective functions, even though network clustering is required to address multiple underlying structural attributes. To overcome this limitation, multi-objective methods that mine multiple attributes of a network simultaneously would be desired so as to improve the accuracy and efficiency of network clustering [9–12]. Along this direction, several multi-objective optimization methods for network clustering have been developed that simultaneously make tradeoffs among multiple contradictory objectives. For instance, a multi-objective network clustering method, called MOGA-Net, applies the non-dominated sorting genetic algorithm-II (NSGA-II) to optimize both the community score and community fitness simultaneously [13]. Two contradictory optimization functions, kernel K-means (KKM) and ratio cut (RC), are used in developing the discrete multi-objective particle swarm optimization (MODPSO) method for network clustering [9].

While they represent an improvement over the single-objective methods, the existing multi-objective optimization methods also have certain limitations, the most notable of which is known as the problem of instability when performing network clustering, that is, the processes of exploration and exploitation cannot be well balanced. Such an imbalance may cause inefficient search of the solution space, and as a result, some solutions may fall into local optima [14].

The moth-flame optimization (MFO) algorithm is a new stochastic optimization algorithm [15]. Combined with the simple flame generation (SFG) and the spiral flight search (SFS) strategy, the MFO is capable of achieving a good balance between exploration and exploitation. By incorporating these characteristics, MFO has been successfully utilized to deal with many real-world scientific and engineering optimization problems. However, the existing MFO variants cannot be applied to network clustering problems well. In this paper, we aim to further develop a novel multi-objective discrete moth-flame optimization algorithm (DMFO) for network clustering. On the basis of the label-based representation of discrete individuals, a discrete MFO variant is first extended by redesigning the SFG and SFS strategies. Moreover, a multi-objective optimization framework based on the two contradictory optimization functions (i.e., KKM and RC) is presented for network clustering. Experimental results show that our DMFO method obtains the outstanding performance.

The rest of this paper is organized as follows. The problem formulation is given in Sect. 2. Section 3 presents the proposed method for network clustering. Extensive experiments are carried out to show the performance of DMFO in Sect. 4. Finally, basic concluding remarks are discussed in Sect. 5.

2 Problem Formulation

A complex network can be modeled by an undirected graph $G = (V, E)$, where V and E represent the sets of nodes and links, respectively. A community of complex network is defined as a group of nodes, which have more intra-links than inter-links [4]. The task of network clustering in a complex network is to obtain densely connected subgraphs $G_i (i = 1, \dots, k)$ in G , where G obeys $\cup_{1 \leq i \leq k} G_i = G$ and $\cap_{1 \leq i \leq k} G_i = \emptyset$.

The process of network clustering can be formulated as a multi-objective optimization problem (MOP) [9] in which several criteria are proposed to measure the densities of intra-links and inter-links. In this paper, we select the kernel K-means (KKM) and the ratio cut (RC) as two conflicting objective functions to maximize the internal connections and to minimize the external connections, respectively [9]. Given a network G consisting of n nodes and m links, the adjacent matrix of G is $A = (A_{ij})_{n \times n}$. If G has a cluster partition $C = \{C_1, \dots, C_k\}$ in which C_i is the node set of subgraph $G_i (i = 1, \dots, k)$, the multi-objective optimization problem can be formulated as Eq. (1).

$$\min \begin{cases} f_1 = KKM = 2(n - k) - \sum_{i=1}^k \frac{L(C_i, C_i)}{|C_i|} \\ f_2 = RC = \sum_{i=1}^k \frac{L(C_i, \bar{C}_i)}{|C_i|} \end{cases} \quad (1)$$

where $L(C_i, C_i) = \sum_{i \in C_i, j \in C_i} A_{ij}$ and $L(C_i, \bar{C}_i) = \sum_{i \in C_i, j \in \bar{C}_i} A_{ij}$ mean the densities of internal and external links for nodes in $C_i (i = 1, \dots, k)$, respectively.

To evaluate the quality of network partitions, we use the modularity Q [6] and the normalized mutual information (NMI) [16] as evaluation metrics, which are defined in Eq. (2) and Eq. (3), respectively. If the real community partition of a network is known, we evaluate the performance of the algorithm using both NMI and Q . Otherwise, only Q is used to estimate the clustering results.

$$Q = \sum_{s=1}^{N_c} \left(\frac{l_s}{2m} - \left(\frac{d_s}{2m} \right)^2 \right) \quad (2)$$

where N_c is the number of detected clusters in a network. l_s and d_s denote the total number of links and degrees in the cluster s , respectively. Specifically, the larger the Q value is, the higher the clustering quality of the algorithm is.

$$NMI = \frac{-2 \sum_{i=1}^{c_A} \sum_{j=1}^{c_B} C_{ij} \log(C_{ij}n / C_i \cdot C_j)}{\sum_{i=1}^{c_A} C_i \log(C_i/n) + \sum_{j=1}^{c_B} C_j \log(C_j/n)} \quad (3)$$

where $c_A(c_B)$ denotes the number of clusters in partition $A(B)$. C represents the confusion matrix. $C_i \cdot (C_j)$ means the sum of the elements of C in the i^{th} row

(j^{th} column). The larger the NMI value is, the closer the obtained division is to the intrinsic partition. The value of NMI is between 0 and 1.

3 The Proposed Method

In this section, we first introduce the representation and initialization of discrete individuals. Then, we redesign two strategies of DMFO (i.e., SFG and SFS) to achieve a good balance of exploration and exploitation in the clustering process. Finally, an improved Tchebycheff decomposition method is applied to optimize the multi-objective network clustering problem.

3.1 Representation and Initialization of Discrete Individuals

The representation problem of discrete position is essential for MFO-based optimization. It is a key step for an algorithm to encode the scheme of a solution. This paper adopts the label-based representation strategy whose label denotes a cluster value of a node [9]. Although there are two populations of moths and flames in MFO, their positions, in essence, represent a division of a network. Therefore, we adopt the same representation strategy. More specifically, if there are D nodes in a network, the positions of the i^{th} moth and the i^{th} flame are redefined based on Eq. (4).

$$M_i = \{m_{i1}, \dots, m_{ij}, \dots, m_{iD}\}, F_i = \{f_{i1}, \dots, f_{ij}, \dots, f_{iD}\} \quad (4)$$

where $m_{ij}(f_{ij})$ denotes the group value of the j^{th} node in the i^{th} moth (flame). j is a random integer in the range of $[1, D]$. If $m_{ij}(f_{ij})$ and $m_{ik}(f_{ik})$ are equal, then the j^{th} and k^{th} node of the i^{th} moth (flame) belong to the same cluster.

Figure 1 presents an illustration of the mechanism for moth (of flame) encoding and decoding, based on its discrete position representation. To speed up the convergence of the proposed DMFO, we apply a label propagation strategy, as defined in [9], to initialize the moth population randomly.

3.2 SFG Strategy of DMFO

In the proposed DMFO method, the aim of SFG strategy is redesigned to avoid falling into local optima during the clustering process. More specifically, we first select the best individuals in the current two generations as flames to guide the moths to find the optimal solution. Since the flames play a vital role in the discrete variant, the neighborhood-based two-way crossover and neighbor-based mutation (NBM) [9] are adopted in the SFG strategy in order to obtain the well-performed flames while avoiding the flames falling into local optima. Then, the obtained flames are merged with the moths obtained in this iteration, and the well-performed individuals are further selected as the next-generation flames based on the modularity Q .

The core of neighborhood-based two-way crossover is as follows. For the i^{th} individual $F_i (i = 1, \dots, N)$, an individual $F_j (j = 1, \dots, H)$ is first selected

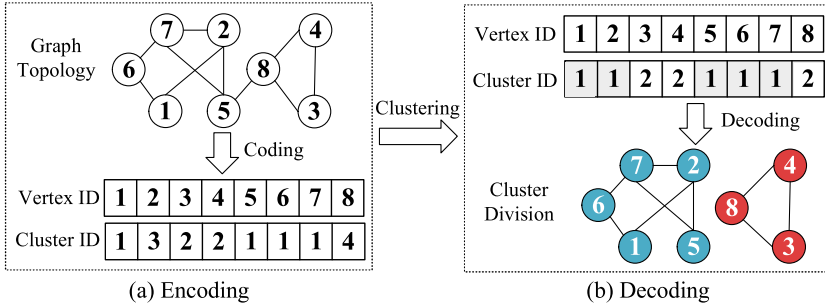


Fig. 1. An illustration of discrete position representation and the mechanism for a discrete individual encoding and decoding. (a) shows the process of encoding a graph into a discrete individual based on a label-based strategy. (b) shows how to decode a discrete individual into cluster label after clustering a discrete individual using the network clustering method. The Cluster ID denotes the cluster label of each node.

randomly in its neighborhood H . The neighborhood where each moth selects the crossover object is chosen by the distances among their aggregation multi-objective weight vectors. Then, F_i and F_j will serve as both the target individual and source individual of the crossover. Finally, two new individuals will be generated by the two-way crossover simultaneously. F'_i represents the better of the two new individuals. If F'_i is better than F_i , replace F_i , otherwise, keep it unchanged.

3.3 SFS Strategy of DMFO

The SFS strategy is redesigned in order to efficiently and extensively search for better results in a solution space. Specifically, the distance U moves when the position of a moth is updated based on Eq. (5).

$$U_i = R_i \cdot \text{Sig}(|e^{bt} \cdot \cos 2\pi t|) \tag{5}$$

where b is a constant and t represents a random number in the range of $[-1, 1]$. $R_i = M_i \oplus F_j$ represents the distance between the moth M_i and the flame F_j . Specifically, \oplus is a XOR operator. If the corresponding node cluster value between M_i and F_j is the same, R_i is 0, otherwise 1. Suppose $Y = (y_1, \dots, y_D)$, $X = (x_1, \dots, x_D)$. In Eq. (5), the function $Y = \text{Sig}(X)$ is defined in Eq. (6).

$$y_i = \begin{cases} 1 & \text{for } \text{rand}(0,1) < \text{sigmoid}(x_i) \\ 0 & \text{for } \text{rand}(0,1) \geq \text{sigmoid}(x_i) \end{cases} \quad (i = 1, \dots, D) \tag{6}$$

where $\text{sigmoid}(x) = 1/(1 + e^{-x})$ [9].

The redefined U function in Eq. (5) is used to determine if the moth needs to be updated. Here, we assume that the k^{th} and the $(k + 1)^{\text{th}}$ generation moth are $M_i(k) = (m_{i1}^k, m_{i2}^k, \dots, m_{iD}^k)$ and $M_i(k + 1) = (m_{i1}^{k+1}, m_{i2}^{k+1}, \dots, m_{iD}^{k+1})$, respectively, and $U_i = (u_1, u_2, \dots, u_D)$. The update rule of moth position is

redefined as $M_i(k + 1) = U_i \odot M_i(k)$. k is the number of iterations and the operator \odot is the specific update strategy of the moth based on Eq. (7).

$$m_{id}^{k+1} = \begin{cases} m_{id}^k & \text{if } u_d = 0 \\ Nbest_d^k & \text{else} \end{cases} \tag{7}$$

where $Nbest_d^k$ represents the cluster value owned by the most of neighbors of the d^{th} node of a moth.

In addition, in the redefined SFS strategy, the number of flames automatically decrease with the iterative process. That is, $F_{no} = \text{round} (N - (k(N - 1))/K)$, where the $\text{round}(\cdot)$ operation is used to obtain the nearest integer of \cdot . N is the maximum number of flames. K is the iterative maximum number. F_{no} denotes the adaptive number of flame. Figure 2 plots the position update process of a moth according to the redefined SFS strategy.

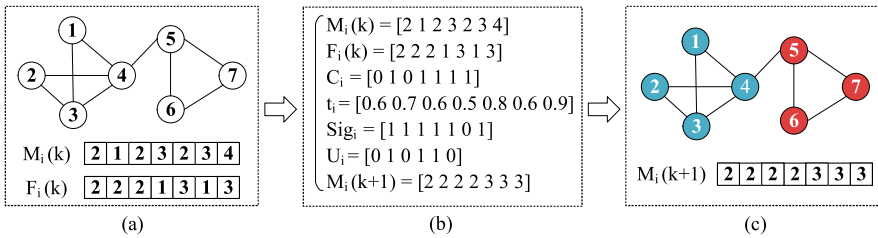


Fig. 2. An illustration of a moth using redefined SFS strategy for position updates. (a) The discrete representation of the i^{th} moth and flame (i.e., M_i and F_i) at the k^{th} generation. (b) The regeneration process of the i^{th} moth. (c) A new i^{th} moth and its graph structure.

3.4 The l_2 -norm Constraint

In our DMFO method, an improved Tehebycheff decomposition method with l_2 -norm constraint on direction vectors (2-Tch) [17] is used to decompose the multi-objective optimization problem into a set of scalar optimization sub-problems. The 2-Tch method is defined in Eq. (8).

$$\min_{\mathbf{x} \in \Omega} g^{\text{ptch}}(\mathbf{F}(\mathbf{x})|\mathbf{w}, \mathbf{z}^*) = \max_{1 \leq i \leq q} \left\{ \frac{f_i(\mathbf{x}) - z_i^*}{w_i} \right\} \tag{8}$$

where $w = (w_1, \dots, w_q)$ with $\|w\|_2 = 1$ and $w_1, \dots, w_q \geq 0$ is a weight vector of a subproblem. q is the number of objective functions and $z_i^* = (z_1^*, \dots, z_q^*)$ is an ideal objective vector with $z_i^* = \min\{f_i(x)|x \in \Omega\}$ ($i = 1, \dots, q$). Algorithm 1 presents the DMFO method framework for network clustering.

Algorithm 1: Framework of the proposed DMFO

Input: The adjacency matrix A of a network, max generation: $Max.iter$, moths or flames size: $popsize$, mutation probability: pm , crossover probability: pc , neighborhood size: $niche$.

Output: Pareto front solutions: PF , each solution represents a cluster division.

```

1 Initialization:
2 Initialize  $M(0)$ ;
3 Generate a uniformly distributed weight vector  $w = (w_1, \dots, w_{popsize})$ ;
4 Initialize the neighborhood of each moth(flame) based on the Euclidean
  distance of weight vectors;
5 Initialize reference point  $z^*$ .
6 Main loop:
7 For  $k = 1 : Max.iter$  do
8    $F(k) \leftarrow$  execute SFG strategy;
9    $M(k+1) \leftarrow$  execute SFS strategy;
10  If  $rand(0, 1) < pm$  then
11    Execute the mutation operation for each  $M_i(k+1)$ ;
12    Evaluate all moths  $M(k+1)$ ;
13    If  $M_i(k+1)$  is better than  $M_i(k)$  then
14       $M_i(k+1) \leftarrow M_i(k+1)$ ;
15    Else
16       $M_i(k+1) \leftarrow M_i(k)$ ;
17    Update reference point  $z^*$ ;

```

4 Experiments

4.1 Datasets and Parameters

Synthetic Datasets. The extended Girvan-Newman (GN) benchmark networks [6] are selected to verify the performance of proposed DMFO method. These networks contain 128 nodes, which are equally divided into four clusters with 32 nodes, and the average degree of each node is 16. μ denotes a mixing parameter in the range of $[0, 1]$. More specifically, a fraction of $(1 - \mu)$ links of each node is shared with the nodes in the same cluster and the rest of μ links are shared with the other nodes in other clusters. In general, as μ increases, the community structure gradually becomes less evident. When the mixing parameter $\mu < 0.5$, the number of neighbors of each node belonging to its cluster is more than the number of other neighbors in other clusters. Thus, the networks have a clear community structure. When $\mu > 0.5$, the community structure will be very blurred. Therefore, the mixing parameter μ ranges from 0.0 to 0.5, with a spacing of 0.05, resulting in a total of 11 synthetic networks in our experiments. This paper uses Q as the clustering evaluation metrics for the synthetic networks.

Real-World Datasets. Four real-world networks are used to test the performance of DMFO, which include the Zachary's Karate Club network (G_1) [18],

the Bottlenose Dolphin network (G_2) [13], the American College Football Network (G_3) [6], and the Books about US Politics network (G_4) [9]. Specifically, G_1 is a social network of friendships among 34 members of a karate club. This network is divided into two groups. G_2 is an animal social network consisting of 62 bottlenose dolphins. G_3 is a football network, which consists of 115 nodes and 616 links representing the football teams and the regular-season matches between different teams, respectively. The whole network is divided into 12 clusters. G_4 is a book network, consisting of 105 nodes and 441 links. We utilize Q and NMI simultaneously to evaluate the clustering performance of the algorithm on four real-world datasets.

Parameter Setting. In DMFO, the number of maximum of iteration for all datasets is set to 80, and the population size is customized according to the size of datasets. Specifically, the size of population for G_1 and G_2 is 150, and the rest is set to 100. In addition, moths are mainly updated according to the flames in DMFO, and it is easy to fall into local optima. Therefore, we increase the mutation rate and crossover rate, i.e., $pm = 0.3$, $pc = 0.9$, for avoiding trapping in the local optimization.

To evaluate the performance of different methods, we select six state-of-the-art network clustering methods, namely, GA-Net [7], MOGA-Net [13], MODPSO [9], QDM-PSO [10], MOPSO-Net [11], MOCD-ACO [12], to compare with the proposed DMFO. The parameter settings for other algorithms are based on their corresponding references. Besides, the averaged results are obtained by running 10 times independently for each experiment in order to obtain more reliable and accurate results.

4.2 Experimental Results

Comparison Results on Synthetic Networks. Figure 3 provides the statistical results of the NMI for each method on synthetic networks with varying mixing parameters from 0 to 0.5. More specifically, DMFO can also detect the true clustering structures even when the mixing parameter μ is 0.45 in which the community structure is blurred. GA and MOGA-Net methods only detect the true cluster division of networks when μ is below 0.15. When $\mu > 0.35$, NMI of GA-Net is 0 which means that such method cannot cluster a network effectively. For the MODPSO, QDMPSO, and MOPSO-Net, the performance of three methods is similar. All of them can discover the real division of a network when $\mu < 0.45$. In summary, our DMFO method achieves outstanding performance for clustering the extended GN benchmark datasets.

Comparison Results on Real-World Networks. Figure 4 illustrates the histogram of the comparison results in terms of Q and NMI on four real-world datasets. As we can see from Fig. 4(a), the optimal average value of Q can be obtained by our DMFO method on four real-world networks. It can be seen from Fig. 4(b) that the proposed DMFO can obtain $NMI = 1$ on both G_1 and G_2

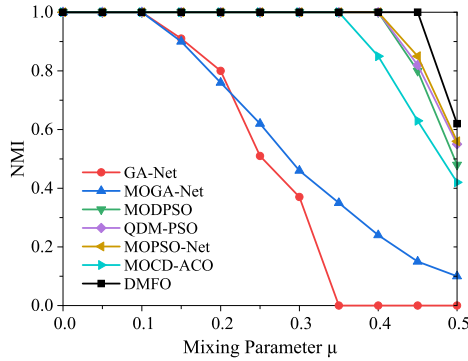


Fig. 3. Comparison results in terms of NMI on the GN extended benchmark datasets. The figure shows that our DMFO outperforms other methods on synthetic datasets.

networks, which means that DMFO can obtain true cluster partition results. In addition, on G_3 and G_4 , our DMFO method can also get the best average value of NMI . Therefore, we can conclude that the proposed DMFO can effectively find the optimal cluster partition results compared with other algorithms.

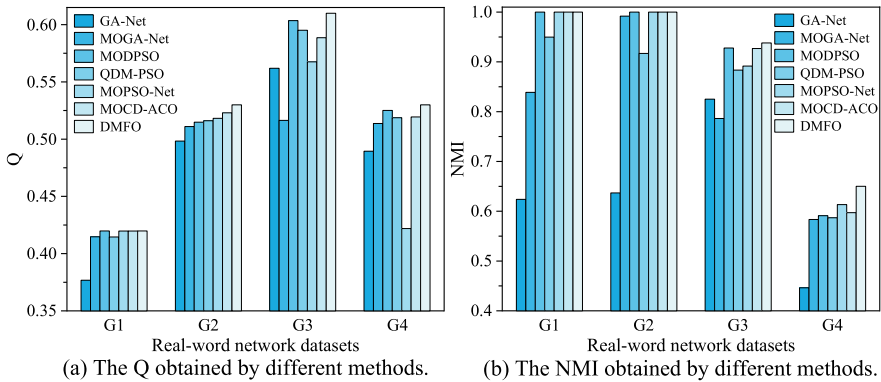


Fig. 4. Comparison results in terms of Q and NMI on the four real-world datasets. From the figure we can obtain that our DMFO method can achieve the optimal network clustering results on four real-world datasets.

5 Conclusion

In this article, a decomposition-based multi-objective discrete moth-flame optimization (DMFO) method is developed for network clustering. More specifically, the novelty of our proposed method is to achieve a good balance between exploration and exploitation in the clustering process by redefining the two update

strategies of MFO algorithm, which allow for better clustering to be obtained. Moreover, the improved Tchebycheff method (2-Tch) is used in the proposed method to decompose the network clustering problem into a family of sub-problems. Experimental results in synthetic and real-world datasets show that our proposed method can obtain the outstanding solutions than other comparison algorithms.

Acknowledgment. This work was supported by the National Natural Science Foundation of China (Nos. 61976181, 11931015) and Natural Science Foundation of Chongqing (Nos. cstc2018jcyjAX0274, cstc2019jcyj-zdxmX0025).

References

1. Chiti, F., Dobson, C.M.: Protein misfolding, amyloid formation, and human disease: a summary of progress over the last decade. *Annu. Rev. Biochem.* **86**, 27–68 (2017)
2. He, K., Li, Y., Soundarajan, S., Hopcroft, J.E.: Hidden community detection in social networks. *Inf. Sci.* **425**, 92–106 (2018)
3. Strano, E., Viana, M.P., Sorichetta, A.: Mapping road network communities for guiding disease surveillance and control strategies. *Sci. Rep.* **8**(1), 1–9 (2018). Article no. 4744
4. Radicchi, F., Castellano, C., Cecconi, F.: Defining and identifying communities in networks. *Proc. Natl. Acad. Sci.* **101**(9), 2658–2663 (2004)
5. Newman, M.E.J.: Fast algorithm for detecting community structure in networks. *Phys. Rev. E* **69**(6), 066133 (2004)
6. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. *Proc. Natl. Acad. Sci.* **99**(12), 7821–7826 (2002)
7. Pizzuti, C.: GA-Net: a genetic algorithm for community detection in social networks. In: Rudolph, G., Jansen, T., Beume, N., Lucas, S., Poloni, C. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 1081–1090. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87700-4_107
8. Chang, H., Feng, Z., Ren, Z.: Community detection using dual-representation chemical reaction optimization. *IEEE Trans. Cybern.* **47**(12), 4328–4341 (2017)
9. Gong, M., Cai, Q., Chen, X., Ma, L.: Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition. *IEEE Trans. Evol. Comput.* **18**(1), 82–97 (2014)
10. Li, L., Jiao, L., Zhao, J., Shang, R., Gong, M.: Quantum-behaved discrete multi-objective particle swarm optimization for complex network clustering. *Pattern Recogn.* **63**, 1–14 (2017)
11. Rahimi, S., Abdollahpouri, A., Moradi, P.: A multi-objective particle swarm optimization algorithm for community detection in complex networks. *Swarm Evol. Comput.* **39**, 297–309 (2018)
12. Ji, P., Zhang, S., Zhou, Z.P.: A decomposition-based ant colony optimization algorithm for the multi-objective community detection. *J. Ambient Intell. Humaniz. Comput.* **11**(1), 173–188 (2019). <https://doi.org/10.1007/s12652-019-01241-1>
13. Pizzuti, C.: A multiobjective genetic algorithm to find communities in complex networks. *IEEE Trans. Evol. Comput.* **16**(3), 418–430 (2012)
14. Chen, X., et al.: A new evolutionary multiobjective model for traveling salesman problem. *IEEE Access* **7**, 66964–66979 (2019)

15. Mirjalili, S.: Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. *Knowl.-Based Syst.* **89**, 228–249 (2015)
16. Danon, L., Díaz-Guilera, A., Duch, J., Arenas, A.: Comparing community structure identification. *J. Stat. Mech. Theory Exp.* **2005**(09), P09008 (2005)
17. Ma, X., Zhang, Q., Tian, G., Yang, J., Zhu, Z.: On Tchebycheff decomposition approaches for multiobjective evolutionary optimization. *IEEE Trans. Evol. Comput.* **22**(2), 226–244 (2018)
18. Duch, J., Arenas, A.: Community detection in complex networks using extremal optimization. *Phys. Rev. E* **72**(2), 027–104 (2005)



Predicting Associations Between Proteins and Multiple Diseases

Martin Breskvar^{1,2} and Sašo Džeroski^{1,2}

¹ Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia

² Jožef Stefan International Postgraduate School, Ljubljana, Slovenia

{Martin.Breskvar,Saso.Dzeroski}@ijs.si

Abstract. We formulate the task of predicting protein-disease associations as a multi-label classification task. We apply both problem transformation (binary relevance), i.e., local approaches, and algorithm adaptation methods (predictive clustering trees), i.e., global approaches. In both cases, methods for learning individual trees and tree ensembles (random forests) are used. We compare the predictive performance of the local and global approaches on one hand and different feature sets used to represent the proteins on the other.

Keywords: Protein-disease associations · Multi-label classification · Predictive clustering trees · Random forests · Network embeddings

1 Introduction

Mutations and other alterations in many genes and gene products (proteins) in the cell can manifest themselves as distinct disease prototypes [10]. Since proteins interact with each other through protein-protein interaction networks (PPIs), the impact of altering one protein can spread along the links of the PPI and affect neighboring proteins. The sets of proteins associated with a given disease and their interactions are known as disease pathways.

Studying a single disease protein in isolation cannot fully explain most human diseases. On one hand, each disease can be associated with many proteins. Computational methods have thus been developed to predict which proteins are associated with a given disease and to bring them together into pathways. Typically, a binary classification problem is formulated for the disease at hand, with each protein corresponding to one instance/example. On the other hand, a protein can be associated with more than one disease. In this context, each disease gives rise to a binary classification problem. Considering the association between proteins and multiple diseases simultaneously corresponds to a multi-label classification (MLC) problem.

Diseases can be divided and subdivided into categories and subcategories, following the Human Disease Ontology [16]. This means that the space of labels (diseases) is organized into a hierarchy. Given that each gene can be associated to

multiple diseases (labels) and that the diseases are organized into a hierarchy, the task at hand becomes a task of hierarchical MLC, i.e., predicting the association between proteins and multiple diseases simultaneously, while taking into account the Disease Ontology.

Proteins (and the genes that encode them) can be described in many different ways. Here we choose to use three different sources of information about the proteins/genes. First, we use the protein-protein interaction network. In particular, we use features from the embedding of the nodes in the PPI network, as proposed by Agrawal et al. [1]. We also use functional annotations of the genes/proteins from the Gene Ontology (GO) [2, 7, 9]. Finally, we use information on the known pathways in which genes/proteins are involved from the Kyoto Encyclopedia of Genes and Genomes (KEGG) [12].

In this paper, we apply machine learning methods to the task of learning to predict associations between proteins and multiple diseases. We consider this task as a MLC task and apply both problem transformation (binary relevance), i.e., local approaches, and algorithm adaptation methods (predictive clustering trees), i.e., global approaches. In both cases, methods for learning individual trees and tree ensembles (random forests) are used. On one hand, we compare the predictive performance of the local and global approaches. On the other hand, we compare the different feature sets used to represent proteins.

The remainder of the paper is organized as follows. Section 2 describes the different datasets for predicting associations between proteins and multiple diseases, including the used data sources and the feature engineering process. Section 3 briefly summarizes the applied machine learning methods. Section 4 describes the experimental setup. In Sect. 5, we present the results of applying the machine learning methods to the prepared MLC datasets. Section 6 concludes with a discussion and an outline of directions for further work.

2 Data

We use five different sources of information to construct the MLC datasets we consider, three of which have been used by Agrawal et al. [1]. Examples/instances in our datasets correspond to proteins and labels/targets to diseases. To construct the label part of the data, we use DisGeNET [15] and the Human Disease Ontology. For the feature part of the data, we use a human PPI network [8, 14], the Gene Ontology and the Kyoto Encyclopedia of Genes and Genomes.

2.1 Data Sources

To derive the label part of our datasets, we use a database of protein-disease associations, which contains pairs $a(p, d)$, indicating that alteration of protein p is linked to disease d . Protein-disease associations were taken from Agrawal et al. [1], who in turn took them from DisGeNET [15]. Agrawal et al. [1] consider 519 diseases that each have at least 10 disease proteins.

Of the 519 diseases, we consider the 302 which can be found in the Disease Ontology. Namely, for each of the 519 diseases, represented with UMLS (Unified Medical Language System) [5] codes, only 302 diseases map to the codes in the Disease Ontology.

We use the human PPI network compiled by Menche et al. [14] and Chatr-Aryamontri et al. [8], as provided by Agrawal et al. [1]. The network contains physical interactions between proteins experimentally confirmed in humans. This includes metabolic enzyme-coupled interactions and signaling interactions. The network is unweighted and undirected with $n = 21,557$ proteins and $m = 342,353$ experimentally confirmed interactions. Proteins are mapped to genes and the largest connected component of the PPI network is used.

The Gene Ontology (GO) knowledge base is the world's largest source of information on the functions of genes. Terms in GO describe the molecular functions, cellular locations, and processes gene products may carry out. Genes (and proteins) are annotated with terms from GO and we use these annotations to generate features describing proteins.

The Kyoto Encyclopedia of Genes and Genomes (KEGG) is a resource for understanding high-level functions and utilities of biological systems, such as the cell and the organism, from genomic and molecular-level information. KEGG is an integrated database resource consisting of eighteen databases. In this paper, we use one of them, namely the KEGG PATHWAY database.

2.2 Feature Engineering

The first set of features that we used to describe proteins was derived from the PPI network mentioned above. In particular, an embedding of the nodes in the network was generated by using the *node2vec* [11] approach: This is a neural embedding approach that learns a vector representation for each protein using a single-layer neural network and random walks. The default parameter values of *node2vec* were used, generating an embedding of 128 dimensions (features).

An alternative representation of the proteins is obtained by using functional annotations from the Gene Ontology (GO). In principle, each term from the GO gives rise to one binary feature, which is true for a given gene/protein if the latter is annotated with the term at hand. We filter the resulting features by excluding those terms that are not assigned to any of the considered proteins (e.g., are false throughout). The total number of generated GO features is 11,695.

In a similar manner, we use KEGG to derive new features describing the proteins. Each term from KEGG corresponds to one binary feature. The feature is true for a given protein if the protein is annotated with the term, i.e., is known to belong to the pathway corresponding to the KEGG term. The total number of generated KEGG features is 222.

2.3 Disease Groups

It is known that a single protein can influence several diseases and that a single disease can be affected by several proteins. Therefore it may be beneficial to

Table 1. Properties of the generated datasets. The datasets share a common set of features (embeddings and GO/KEGG terms) and have different numbers of labels (diseases) and examples. The features from network embeddings are numeric, whereas the GO/KEGG features are binary. The column l_c denotes label cardinality, i.e., average number of labels per example.

	Examples	Features	Labels	l_c
Embeddings				
All diseases	5,255	128	302	2.398
Cardiovascular system diseases	951		33	1.221
Cancer	2,214		68	1.486
Immune system diseases	1,055		21	1.144
Nervous system diseases	1,063		44	1.277
GO/KEGG				
All diseases	5,199	11,917	302	2.410
Cardiovascular system diseases	946		33	1.222
Cancer	2,203		68	1.488
Immune system diseases	1,049		21	1.145
Nervous system diseases	1,051		44	1.279

narrow the set of considered diseases by focusing the analysis and model learning on specific subsets of it. We study four groups of diseases, defined by the Disease Ontology. We generate subsets of the original datasets by examining the second-level of the ontology. This level consists of several categories. Here, we focus only on the following four: cardiovascular system diseases (33), various types of cancers (68), immune system diseases (21) and nervous system diseases (44).

For each of the considered disease groups, new variants of the original datasets are constructed, containing features generated from neural embeddings as well as GO/KEGG features. This results in 8 new variants of the original two datasets. Considering only a subset of the diseases can result in unlabeled examples. This problem is less prominent when examples in the dataset are densely labeled. However, in our case, the label space is particularly sparse. In both original datasets, on average, each example is labeled only with less than 3 out of the 302 possible labels. The resulting 8 generated variants consequently contain considerably fewer examples than the original two datasets. Table 1 summarizes all generated datasets, which are available online¹. The entry “all diseases” refers to the two original/complete datasets.

3 Methods

Local vs. Global Approaches to MLC. The tasks we address in this paper are tasks of MLC. Such tasks can generally be approached in two ways: locally or

¹ http://kt.ijs.si/martin_breskvar/data/ismis2020.

globally. The local approaches learn a model for each label separately, whereas the global approaches learn one model for all labels simultaneously. In this paper, we use both.

Local Approach. The local approach, known under the name of binary relevance, addresses the MLC task by splitting it up into many binary classification tasks. This approach is often used and performs well. Its advantage is the ability to use any readily available binary classification method. This approach yields as many models as there are labels. Its disadvantage is its inability to capture potential relations in the target space.

Global Approach. The global approaches require specially crafted methods in order to capture the potential relations in the target space. Such approaches result in one model, which is able to make predictions for all considered labels. Such approaches have been shown to reduce overfitting and computational effort. The disadvantage of using global approaches is the need to use specially designed methods that can handle more complex target spaces.

Single Trees and Tree Ensembles for MLC. In this paper we build tree models and ensembles thereof. In particular, we use predictive clustering trees (PCTs) [4] as individual models and Random forests of PCTs [13] as model ensembles to address the MLC task.

Single Trees. A PCT is a generalized decision tree, obtained with the standard top-down induction (TDI) algorithm for decision trees. The induction algorithm for PCTs is general in the sense that it allows for different impurity and prototype functions. The impurity function is used to calculate the homogeneity of a given set of data instances, whereas the prototype function is applied to calculate representative target values for the set that can be used as predictions. With this, PCTs can address many different tasks, including regression and classification, (hierarchical) multi-target regression and (hierarchical) multi-label classification, and time series prediction. The CLUS software package² contains implementations of all mentioned approaches.

To address the MLC task, the PCT induction algorithm uses the *impurity function* $IM(E) = \frac{1}{|T|} \sum_{a \in T} \text{Gini}_a(E)$, where T is the set of all considered labels and Gini is the Gini index. The *prototype function* is defined as a vector \hat{y} , where the components (\hat{y}_a) represents the predictions for target attributes (labels) a . The presence of a label in a leaf node is predicted if the majority of training data instances, that arrive to that node, are labeled with it.

Tree ensembles. An ensemble is a set of models, called base predictive models, where each model contributes to the overall prediction. It is a common practice to use ensembles to lift the predictive performance of the base predictive models and thus obtain better predictive performance. Ensembles make predictions by combining the predictions of their base models (we use probability distribution voting [3] that has been extended towards MLC). We build our ensembles

² <http://source.ijs.si/ktclus/clus-public/>.

with the Random forest algorithm [6] that has been extended to multi-target prediction [13], including MLC, and uses PCTs as base predictive models.

4 Experiments

Experimental Setup. We consider two datasets, with the same examples (proteins) and targets (disease associations), composed as previously described. They differ in the set of features (attributes/independent variables). The first dataset comprises the features obtained via the neural embedding (NE) of the nodes of the PPI. The second dataset comprises the features obtained by using the GO and the KEGG databases. In Table 1, these two datasets are marked with “all diseases”. In addition to these two datasets, we also consider their variants, generated according to the disease groups described in Sect. 2.3.

We apply PCTs and random forests thereof in four different ways. We learn global single trees, global ensembles, local single trees (one tree per label) and local ensembles (one ensemble per label). When building single trees (global or local), we apply F -test pruning, with the optimal value of the F parameter determined by internal 3-fold cross-validation.

Method Parameters. For single trees, we search for the optimal F -test value considering the following values: 0.125, 0.1, 0.05, 0.01, 0.005, 0.001. Random forests use 100 base predictive learners (PCTs), which are left unpruned and evaluate the candidate splits for 25% of all input attributes.

Evaluation Measures. We evaluate our models by calculating two measures of predictive performance: the area under the average receiver operating characteristic (AUROC) and the area under the average precision-recall curve (AUPRC). Standard AUROC and AUPRC values are calculated for each target (disease) separately and then averaged across all targets (302 diseases, in case of “all diseases” datasets). We selected to monitor these two measures because they are threshold independent (unlike the accuracy, precision and recall measures). Although AUROC is widely used, it can give overly optimistic evaluation scores in cases where a large skew in the class distribution is present. This happens because correctly predicting the absence of a label is rewarded. To address this issue, we also report AUPRC, which handles this deficiency appropriately.

We estimated the predictive performance of the obtained models by using 10-fold cross-validation. The results which we report were obtained on the testing folds. In addition to the estimated predictive performance, we also report the *overfitting score*: a measure of how much a model overfits the training data. We calculate the overfitting score (OS) for $\overline{\text{AUROC}}$ as follows:

$$\text{OS} = (\overline{\text{AUROC}}_{\text{train}} - \overline{\text{AUROC}}_{\text{test}}) / \overline{\text{AUROC}}_{\text{train}}.$$

The OS for $\overline{\text{AUPRC}}$ is calculated analogously. Smaller values are better, because they indicate less overfitting.

5 Results

Table 2 gives the results, i.e., estimates of predictive performance on unseen data, in terms of AUROC and AUPRC scores, respectively. Table 3 gives the overfitting scores, calculated for the AUROC and AUPRC metrics. We investigate the predictive performance along the following dimensions:

1. Which approach performs better (local or global)?
2. Which method performs better (single tree or ensembles)?
3. Which input data is more informative?
4. Which models overfit less?

Table 2. AUROC and AUPRC results for local and global models as well as single trees and ensembles for the considered datasets.

	AUROC		AUPRC					
	Embeddings	GO/KEGG	Local	Global	Local	Global	Local	Global
	All diseases							
Single tree	0.405	0.335	0.469	0.332	0.013	0.006	0.053	0.006
Random forest	0.633	0.622	0.730	0.712	0.042	0.042	0.121	0.092
	Cancer							
Single tree	0.461	0.348	0.487	0.346	0.032	0.017	0.081	0.017
Random forest	0.623	0.607	0.705	0.696	0.077	0.072	0.150	0.119
	Cardiovascular system diseases							
Single tree	0.473	0.315	0.483	0.307	0.050	0.029	0.165	0.029
Random forest	0.637	0.619	0.715	0.732	0.107	0.118	0.267	0.212
	Nervous system diseases							
Single tree	0.506	0.320	0.532	0.313	0.061	0.021	0.146	0.021
Random forest	0.640	0.640	0.741	0.721	0.132	0.129	0.240	0.189
	Immune system diseases							
Single tree	0.491	0.358	0.504	0.347	0.059	0.044	0.114	0.044
Random forest	0.607	0.578	0.714	0.711	0.097	0.089	0.199	0.177

The results suggest that the single tree models, obtained with the local approach, outperform the global single tree models for both the dataset that contains neural embeddings as input features and the dataset constructed from GO terms and KEGG pathways. The differences in performance are substantial, although the overall performances of both local and global single tree models do not seem encouraging. For ensemble models, the differences in performance are much smaller and performances can be considered comparable.

Table 3. Overfitting scores calculated from $\overline{\text{AUROC}}$ and $\overline{\text{AUPRC}}$ for local and global models as well as single trees and ensembles for the considered datasets.

	$\overline{\text{AUROC}}$				$\overline{\text{AUPRC}}$			
	Embeddings		GO/KEGG		Embeddings		GO/KEGG	
	Local	Global	Local	Global	Local	Global	Local	Global
	All diseases							
Single tree	0.363	0.330	0.304	0.336	0.948	0.229	0.837	0.232
Random forest	0.367	0.378	0.270	0.288	0.958	0.958	0.879	0.908
	Cancer							
Single tree	0.437	0.304	0.322	0.308	0.947	0.212	0.814	0.229
Random forest	0.377	0.393	0.295	0.303	0.923	0.928	0.850	0.881
	Cardiovascular system diseases							
Single tree	0.439	0.370	0.363	0.385	0.926	0.220	0.696	0.223
Random forest	0.363	0.381	0.285	0.268	0.893	0.882	0.733	0.788
	Nervous system diseases							
Single tree	0.419	0.360	0.302	0.374	0.915	0.261	0.728	0.277
Random forest	0.360	0.360	0.259	0.279	0.868	0.871	0.759	0.810
	Immune system diseases							
Single tree	0.466	0.284	0.347	0.305	0.926	0.185	0.792	0.200
Random forest	0.393	0.422	0.286	0.289	0.903	0.911	0.801	0.823

Although predictive superiority of ensembles was expected, the poor performance of single trees was somewhat surprising. It turned out that the low performance of single trees was due to extensive F -test pruning. The results also suggest that the GO/KEGG input features carry more information as compared to the input features obtained with neural embeddings from the PPI network. The performance scores show substantial improvements when using GO/KEGG instead of neural embeddings.

In terms of overfitting, the results are as follows. When using the neural embeddings as input features, single tree global models tend to overfit less (in case of $\overline{\text{AUPRC}}$, the difference is substantial) w.r.t. local single trees. However, in the ensemble setting, global ensembles overfit *slightly more* w.r.t. local ensembles in terms of $\overline{\text{AUROC}}$ and no difference is visible in terms of $\overline{\text{AUPRC}}$. Similar observations can be made for the GO/KEGG dataset. Global single trees overfit more than the local single trees in terms of $\overline{\text{AUROC}}$ and considerably less in terms of $\overline{\text{AUPRC}}$.

Models built for the described disease groups exhibit improved predictive performance in terms of $\overline{\text{AUPRC}}$ as compared to the models built on datasets that contain all diseases. Improvement of predictive performance on the same datasets is comparable in terms of $\overline{\text{AUROC}}$. The predictive performance advantage of using GO/KEGG features over neural embeddings is retained on datasets

that contain only labels of a specific disease group. Global models do not improve over local ones and perform comparably as on the datasets with all diseases.

In terms of AUPRC, the disease group models overfit slightly less than the models for all diseases. The overfitting scores, in terms of AUROC, seem comparable to the ones obtained with models built on datasets with all diseases.

6 Conclusions and Further Work

In this paper, we have presented results of learning predictive models for associating proteins with 302 diseases. We consider two groups of datasets, where each group consists of five datasets: a dataset with all diseases and 4 variants of it, one for each considered disease group. The first group contains features obtained by using neural embeddings on a PPI network. The second group contains features constructed from Gene Ontology and KEGG pathways. We have presented our results in terms of two evaluation measures, namely AUROC and AUPRC. We have also reported overfitting scores which were calculated based on the aforementioned evaluation measures. We were learning local and global models of single predictive clustering trees and ensembles thereof. We show that the GO/KEGG datasets contain more informative features for predicting the considered diseases.

Further work is possible along several directions. We will consider another group of datasets, with both feature sets (neural embeddings and GO/KEGG features). Along this direction, we will also generate embeddings of higher dimensions (i.e., vectors of 256, 512, 1024 features). We also believe that using a hierarchy in the output space can be beneficial. In particular, we will reformulate the learning task into a hierarchical MLC task and take advantage of the Disease Ontology in order to obtain better predictive models. Given the low label cardinality of all considered datasets, we believe it would be beneficial to apply the hierarchical single-label classification (HSC) approach [17]. Feature ranking will also be performed. It is expected that many of the GO/KEGG features will be deemed uninformative: This will speed up the learning process and potentially improve predictive performance. We will also consider evaluation measures only for the top N predicted labels, ordered descending by the predicted probabilities. Examples of such measures are Recall-at-100 and Precision-at-100.

Acknowledgements. We acknowledge the support of the Slovenian Research Agency (grants P2-0103 and N2-0128), the European Commission (grant HBP, The Human Brain Project SGA2), and the ERDF (Interreg Slovenia-Italy project TRAIN). The computational experiments were executed on the computing infrastructure of the Slovenian Grid (SLING) initiative.

References

1. Agrawal, M., Žitnik, M., Leskovec, J.: Large-scale analysis of disease pathways in the human interactome. *Pac. Symp. Biocomput.* **23**, 111–122 (2018)

2. Ashburner, M., et al.: Gene ontology: tool for the unification of biology. *Nat. Genet.* **25**(1), 25 (2000)
3. Bauer, E., Kohavi, R.: An empirical comparison of voting classification algorithms: bagging, boosting, and variants. *Mach. Learn.* **36**(1), 105–139 (1999)
4. Blockeel, H., Raedt, L.D., Ramon, J.: Top-down induction of clustering trees. In: *Proceedings of the 15th International Conference on Machine Learning*, pp. 55–63. Morgan Kaufmann (1998)
5. Bodenreider, O.: The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic Acids Res.* **32**(suppl-1), D267–D270 (2004)
6. Breiman, L.: Random forests. *Mach. Learn.* **45**(1), 5–32 (2001). <https://doi.org/10.1023/a:1010933404324>
7. Carbon, S., et al.: Amigo: online access to ontology and annotation data. *Bioinformatics* **25**(2), 288–289 (2008)
8. Chatr-Aryamontri, A., et al.: The biogrid interaction database: 2015 update. *Nucleic Acids Res.* **43**(D1), D470–D478 (2014)
9. Consortium, G.O.: The gene ontology resource: 20 years and still going strong. *Nucleic Acids Res.* **47**(D1), D330–D338 (2018)
10. Creixell, P., et al.: Pathway and network analysis of cancer genomes. *Nat. Methods* **12**(7), 615 (2015)
11. 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*, pp. 855–864. ACM (2016)
12. Kanehisa, M., Furumichi, M., Tanabe, M., Sato, Y., Morishima, K.: Kegg: new perspectives on genomes, pathways, diseases and drugs. *Nucleic Acids Res.* **45**(D1), D353–D361 (2016)
13. Kocev, D., Vens, C., Struyf, J., Džeroski, S.: Tree ensembles for predicting structured outputs. *Pattern Recogn.* **46**(3), 817–833 (2013). <https://doi.org/10.1016/j.patcog.2012.09.023>
14. Menche, J., et al.: Uncovering disease-disease relationships through the incomplete interactome. *Science* **347**(6224), 1257601 (2015)
15. Piñero, J., et al.: Disgenet: a discovery platform for the dynamical exploration of human diseases and their genes. *Database* **2015** (2015)
16. Schriml, L.M., et al.: Human disease ontology 2018 update: classification, content and workflow expansion. *Nucleic Acids Res.* **47**(D1), D955–D962 (2018). <https://doi.org/10.1093/nar/gky1032>
17. Vens, C., Struyf, J., Schietgat, L., Džeroski, S., Blockeel, H.: Decision trees for hierarchical multi-label classification. *Mach. Learn.* **73**(2), 185 (2008)

Short Papers



Exploiting Answer Set Programming for Building explainable Recommendations

Erich Teppan¹(✉) and Markus Zanker²(✉)

¹ Universität Klagenfurt, Klagenfurt, Austria
erich@ifit.uni-klu.ac.at

² Free University of Bolzano, Bolzano, Italy
mzanker@unibz.it

Abstract. The capability of a recommendation system to justify its proposals becomes an ever more important aspect in light of recent legislation and skeptic users. Answer Set Programming (ASP) is a logic programming paradigm aiming at expressing complex problems in a succinct and declarative manner. Due to its rich set of high level language constructs it turns out that ASP is also perfectly suitable for realizing knowledge and/or utility-based recommendation applications, since every aspect of such a utility-based recommendation capable of producing explanations can be specified within ASP. In this paper we give an introduction to the concepts of ASP and how they can be applied in the domain of recommender systems. Based on a small excerpt of a real life recommender database we exemplify how utility based recommendation engines can be implemented with just some few lines of code and show how meaningful explanations can be derived out of the box.

Keywords: Recommender systems · Explanations · Answer set programming

1 Introduction

The ability of a recommendation system to explain to users why a specific item is recommended has reached considerable research momentum, supported also by recent legislation like GDPR¹ that even codifies a right for explanations of the outcomes of algorithmic decision makers. Although knowledge-based recommendation strategies [9] are a niche topic compared to predominant machine learning based approaches, they are nevertheless in actual use for high-involvement product domains like financial services [3] or consumer products where many variables and aspects are typically considered during decision making processes. Utility-based recommender systems can be considered to be a specific variant of knowledge-based systems, where the matching between user preferences or needs and item properties or features is realized via utility functions. The recent

¹ See <https://eugdpr.org/> for reference.

revived attention for knowledge-based approaches [1] is actually based on three pillars: the wide availability of structured and unstructured content that can be exploited for knowledge extraction, the promise of achieving beyond accuracy goals [11] like transparency, validity and explainability as well as the development of ever more efficient computational mechanisms for processing declarative knowledge. The answer set programming (ASP) paradigm under the stable model semantics [7] must be seen in the context of the latter pillar. It is a successful logic programming (LP) paradigm that has evolved from an academic discipline rooted in deductive databases to an approach that is practically applicable in many different domains [6]. It turns out that ASP, due to its rich set of different language constructs, seems to be perfectly appropriate for also expressing recommendation problems, particularly those ones that can be naturally modeled based on a knowledge or utility-based recommendation paradigm. One big advantage of encoding a recommendation engine in ASP is the compact high level representation that eases maintenance. Another advantage is that explanations for recommendations can be derived quite naturally as a side product of recommendation calculation based on logic rules.

In this paper, we introduce for the first time how the ASP mechanism can be used to build the complete logic of a utility-based recommender system. Hereby, we focus particularly on how to automatically derive meaningful explanations for recommendations. As a first proof of concept we use an excerpt from a real world knowledge base from the financial service domain [4] and show how a corresponding recommendation engine, in particular calculations of item utilities and explanations, can be expressed in ASP in a succinct way.

2 Related Work

To the best of our knowledge there exist only two papers employing ASP in the context of recommender systems. In [10] the authors propose dynamic logic programming, an extension of ASP, in order to provide a user the means to specify changes of their user profile in order to support the recommendation engine in producing better recommendations. Clearly, the focus of this approach is totally different as it is not concerned with building any aspect of the recommendation engine itself.

Not so for the second paper identified. The system described in [8] consists of two ASP based components. The first component is used for automatic extraction of relevant information from touristic offers contained in leaflets produced by tour operators. This information is in turn added to a tourism ontology, i.e. the second component. The second component is responsible for answering the question which touristic offers match a certain customer profile that is also added to the ontology. Thus, the system described in [8] can be seen as a raw content-based recommender. In contrast to that, the approach discussed herein is different in several ways. First, we do not build upon ontologies, but directly harness the ASP knowledge representation. Second, beyond making solely the binary choice, whether an item is matching the user preferences or not, we calculate fine grained utilities to rank items. Third, we demonstrate meaningful

explanations following the taxonomy of [5], where explanations actually reason on all three information categories: user preferences, item properties as well as comparisons with alternatives.

3 Working Example

In order to illustrate the applicability of ASP for knowledge-based recommendation approaches we develop a motivating example. It builds on a utility-based recommendation scenario, where the domain knowledge is encoded by relating user needs and item properties via utility values on different dimensions based on multi-attribute utility theory (MAUT) [2]. Due to the clarity of the dependencies we build on a published example [4] from the domain of financial services that actually constitutes a small excerpt from a real world recommender system in use by a European financial services provider.

The core of MAUT specifications are utility dimensions. These dimensions are abstract concepts which contribute to the overall utility of an item as perceived by a user. In our working example there are two dimensions: *profit*, i.e. the financial return of an investment, and *availability*, i.e. how quickly an investment can be converted back to cash. Note, that these utility dimensions are abstract concepts capable to model and map also indirect relationships between user and/or product attributes. In our example, the simplified user profiles contain just two attributes (or explicit preferences), namely the *duration* of an investment and the personal *goal* of a user with respect to some investment. For the investment *duration* let us assume three distinct values *long*, *medium* and *short* term. Analogously, for the *goal* description we again assume three values: *savings* for a rainy day, profitable *growth* and *venture*. Similarly, properties also describe product items. Let us assume that financial products can be described based on the portions of *shares* a product contains (0%, $\leq 30\%$, $\leq 60\%$, $\leq 80\%$, $> 80\%$) as well as expected price *fluctuation* (*low*, *medium* and *high*).

The connection between utility dimensions and attribute values can be given by so called scoring values (or just scores for short) that specify how much user preferences and product properties contribute to the utility dimensions. The higher the score for a dimension based on a given attribute value the more this attribute contributes to fulfilling this dimension. In this example we use scores between 0 and 10 where a score of 0 signifies that an attribute value does not positively contribute to a dimension. Figure 1 depicts the scores for the user attribute values and resulting utility dimension weights for an example user profile. Conforming to MAUT, the weight that is given to a dimension is estimated as the sum of user attribute value scores on that dimension. Figure 2 depicts the scores for the product attribute values, attribute values for a small example set of four financial products and the resulting product contributions. For a given product, its contribution to a dimension is calculated as the sum of its attribute value scores on that dimension. Finally, the overall utility of each product for a specific user is defined by the sum of weighted dimension contributions. The computations for our toy example are given in Fig. 3.

Hence, for this example scenario, the top ranked product for user *Simon* would be *bond BX* followed by *mutual fund AXP*. These overall utilities can be utilized already for a very basic justification for each recommendation, such as ‘Item X promises the highest overall utility u_x ’. A closer look to the dimensional contributions and weights provides us means for further argumentation, for instance, ‘The top ranked *bond BX* scores also strongest in terms of *availability*, which is also considered most relevant given your preferences p_u ’. On the other hand, *mutual fund AXP* scores second best on both dimensions, i.e. it can be justified as being a good compromise. Although *mutual fund CXP* is the worst overall it shows the best contributions to *profit*. For *bond RX* there is, for instance, a clear *negative* explanation why it must not be recommended since it is dominated by *mutual fund AXP* that offers the same utility score in terms of *availability*, but even more in terms of *profit*. Thus, there is no rational reason for opting towards *bond RX* [12–14].

<u>duration</u>	<u>profit</u>	<u>availability</u>	<u>goal</u>	<u>profit</u>	<u>availability</u>
long	10	2	savings	2	8
medium	7	6	growth	6	4
short	3	10	venture	10	2

<u>profile</u>	<u>duration</u>	<u>goal</u>	<u>profile</u>	<u>profit</u>	<u>availability</u>
Simon	medium	savings	Simon	7+2=9	6+8=14

Fig. 1. User scores, example user profile and resulting dimension weights

<u>shares</u>	<u>profit</u>	<u>availability</u>	<u>fluctuation</u>	<u>profit</u>	<u>availability</u>
0%	2	7	high	7	4
1 – 30%	4	6	medium	5	6
31 – 60%	5	5	low	1	8
61 – 80%	8	2			
81 – 100%	10	1			

<u>product</u>	<u>shares</u>	<u>fluctuation</u>	<u>product</u>	<u>profit</u>	<u>availability</u>
mutual fund AXP	31-60%	medium	mutual fund AXP	5+5=10	5+6=11
bond BX	0%	medium	bond BX	2+5=7	7+6=13
bond RX	0%	high	bond RX	2+7=9	7+4=11
mutual fund CXP	61-80%	high	mutual fund CXP	8+7=15	2+4=6

Fig. 2. Product scores, attribute values and resulting utility contributions

4 ASP for Recommenders

In this section we show how utility based recommendation and explanation can be expressed based on the answer set programming (ASP) paradigm in form of

user	product	utility
Simon	mutual fund AXP	$9*10+14*11=244$
	bond BX	$9*7+14*13=245$
	bond RX	$9*9+14*11=235$
	mutual fund CXP	$9*15+14*6=219$

Fig. 3. Product utilities for the example user profile

first order logic facts and rules. Since it is not possible at this point to give a complete introduction to ASP, we explain the ASP code snippets on an intuitive level².

Listing 1.1 shows how the user profile data from Fig. 1, i.e. the actual user requirements, is encoded by ASP logic. Analogously, Listing 1.2 depicts how the scoring and product data in Fig. 2 can be expressed by logic facts.

```
profile("Simon", duration, medium).
profile("Simon", goal, savings).
```

Listing 1.1. User profile given as logic facts

```
user_score(duration, long, profit, 10).
user_score(duration, medium, profit, 7).
user_score(duration, short, profit, 3).
user_score(duration, long, availability, 2).
user_score(duration, medium, availability, 6).
user_score(duration, short, availability, 10).
user_score(goal, savings, profit, 2).
user_score(goal, growth, profit, 6).
user_score(goal, venture, profit, 10).
user_score(goal, savings, availability, 8).
user_score(goal, growth, availability, 4).
user_score(goal, venture, availability, 2).
product_score(shares, "0", profit, 2).
product_score(shares, "1-30", profit, 4).
product_score(shares, "31-60", profit, 5).
product_score(shares, "61-80", profit, 8).
product_score(shares, "81-100", profit, 10).
product_score(shares, "0", availability, 7).
product_score(shares, "1-30", availability, 6).
product_score(shares, "31-60", availability, 5).
product_score(shares, "61-80", availability, 2).
product_score(shares, "81-100", availability, 1).
product_score(fluctuation, high, profit, 7).
product_score(fluctuation, medium, profit, 5).
product_score(fluctuation, low, profit, 1).
product_score(fluctuation, high, availability, 4).
product_score(fluctuation, medium, availability, 6).
product_score(fluctuation, low, availability, 8).
product("mutual fund AXP", shares, "31-60").
product("mutual fund AXP", fluctuation, medium).
product("bond BX", shares, "0").
product("bond BX", fluctuation, medium).
product("bond RX", shares, "0").
product("bond RX", fluctuation, high).
product("mutual fund CXP", shares, "61-80").
product("mutual fund CXP", fluctuation, high).
```

Listing 1.2. Scoring and product data given as logic facts

² For an in-depth introduction to ASP please refer to [6].

For calculating the contributions of all products for all utility dimensions only the single ASP rule given in Listing 1.3 is needed. Like in other logic programming languages (e.g. Prolog) the ‘:-’ operator stands for left implication. To put it very simple, the left hand side of the rule (i.e. left from ‘:-’) specifies the atoms that are to be added to the solution, which in ASP is called answer set, based on the calculations done on the right hand side. The core of the calculation is performed by a *#sum* aggregate that sums up all scoring values for a product *Productname* on a dimension *Dimension*. Terms beginning with capital letters like *Productname* or *Dimension* stand for logic variables. Consequently, this rule ‘fires’ once for each combination of products and dimensions.

```
contribution(Productname, Dimension, Total):-
  product(Productname, _, _), product_score(_, _, Dimension, _),
  Total=#sum{Score, Attribute:product(Productname, Attribute, Value),
            product_score(Attribute, Value, Dimension, Score)}.
```

Listing 1.3. Calculation of product contributions in ASP

The atoms produced by the rule in Listing 1.3 and included in the answer set are the following:

```
contribution("mutual fund AXP",profit,10)
contribution("bond BX",profit,7)
contribution("bond RX",profit,9)
contribution("mutual fund CXP",profit,15)
contribution("mutual fund AXP",availability,11)
contribution("bond BX",availability,13)
contribution("bond RX",availability,11)
contribution("mutual fund CXP",availability,6)
```

```
weight(Username, Dimension, Total):-
  profile(Username, _, _), user_score(_, _, Dimension, _),
  Total=#sum{Score, Attribute:profile(Username, Attribute, Value),
            user_score(Attribute, Value, Dimension, Score)}.
```

Listing 1.4. Calculation of user weights in ASP

Similarly, all user weights based on the profile and scoring facts are calculated by the ASP rule defined in Listing 1.4. This rule produces the following two solution atoms:

```
weight("Simon",profit,9)
weight("Simon",availability,14)
```

```
utility(Username, Productname, Total):-
  profile(Username, -, -), product(Productname, -, -),
  Total=#sum{W*C, Dimension:weight(Username, Dimension, W),
            contribution(Productname, Dimension, C)}.
```

Listing 1.5. Utility calculation expressed in ASP

Finally, the rule in Listing 1.5 calculates the resulting utilities based on the product contributions and user weights. The solution atoms produced by the rule in Listing 1.5 are:

```
utility("Simon","mutual fund AXP",244)
utility("Simon","bond BX",245)
utility("Simon","bond RX",235)
utility("Simon","mutual fund CXP",219)
```

At this point we want to emphasize that the three rules depicted in Listing 1.3–1.5, which can be seen as a logic representation of the core of a utility-based recommendation engine, are totally generic and do not have to be changed in case of extending the set of products, product or user attributes, attribute values or scoring values.

Building on such an ASP implementation of a recommendation engine, additional rules can be easily added in order to produce solution atoms for explanations. For instance, if we want to support the top ranked item by a corresponding explanation, we can add the rule in Listing 1.6. This rule basically expresses that a product *Productname* with a utility *U* is top ranked if there is no other product *Productname1* with a higher utility *U1*.

```
top_ranked(Username,Productname,U):-utility(Username,Productname,U),
#count{Productname1:utility(Username,Productname1,U1),U1>U}=0.
```

Listing 1.6. Producing an argument for the top ranked product

The rule in Listing 1.6 produces the following solution atom:

```
top_ranked("Simon","bond BX",245)
```

```
top_in_dimension(Dimension,P,C):-contribution(P,Dimension,C),
#count{P1:contribution(P1,Dimension,C1),C1>C}=0.
```

Listing 1.7. Calculating the top item in a dimension

Similarly, we can add a rule for identifying whether a product is best in some dimension. Listing 1.7 shows such a rule, which produces the following atoms:

```
top_in_dimension(profit,"mutual fund CXP",15)
top_in_dimension(availability,"bond BX",13)
```

```
domination(P,"dominated by",P1):-product(P,-,-),product(P1,-,-),
#count{Dimension:contribution(P,Dimension,C),
contribution(P1,Dimension,C1),C>C1}=0,
#count{Dimension:contribution(P,Dimension,C),
contribution(P1,Dimension,C1),C1>C}>=1.
```

Listing 1.8. Calculating totally dominated alternatives

We can also produce counter arguments, for example, on totally dominated product items, i.e. those where there exists an item that is at least equally good in all dimensions and better in at least one dimension. The rule given in Listing 1.8 achieves these *negative* explanations and adds the following atom to the answer set:

```
domination("bond RX","dominated by","mutual fund AXP")
```


5 Evaluation

We evaluated the applicability of our proposed ASP approach with respect real life recommender system scenarios. A typical online recommendation scenario comprises two steps: (1) Based on a given user profile, the recommendation engine first must calculate utilities for all product items. (2) An evolved set of items with the highest utilities is then presented on some form of result page allowing the comparison and explanation of the different proposals. The number of presented items depends on the web page and is commonly between 5 and 20 items. Databases comprising some ten-thousands of product items subject to recommendation can be considered as (very) large yet possible cases. For instance, the streaming platform Netflix offers more than 10000 movies, series and documentations and Amazon Prime more than 20000, and increasing³.

Consequently, we produced two benchmark data sets, one for step 1 and one for step 2, reflecting the size of real recommender system scenarios. The first benchmark dataset consists of 48 generated problem instances comprising product item databases, a random user profile and random item and user scoring rules. The problem instances differ in (a) the number of products (1000, 10000 or 100000), (b) the number of utility dimensions (2, 4, 8 or 16), and (c) the number of user/item attributes (2, 4, 8 or 16). We assumed numeric product item attributes taking values in the range of $\{0, \dots, 10\}$. The benchmark dataset for recommendation step 2, i.e. the explanation of an evolved product item set, consists of 32 instances and is similar to the first dataset. However, the numbers of products are significantly smaller, that is 10 or 20, as there are hardly more items in an evolved set that is presented on the result/comparison page.

We solved each problem instance for benchmark 1 with an encoding consisting of Listing 1.3–1.5. We refer to this encoding as the *utility encoding*. Each problem instance of benchmark 2 (i.e. targeting at explanations for evolved sets) we solved with an encoding that consists of Listings 1.3–1.8. We refer to this latter encoding as *explanation encoding*. For each of the test cases we measured the total time and the peak main memory consumption that was needed by the ASP process for coming up with the solution (i.e. answer set). We ran the experiments on an AMD EPYC 7551 with 32 cores @ 2 GHz (turbo core max 3 GHz) and 256 GByte of main memory. As an ASP solver, we used Clingo 4.5.4 in single-core mode.

Summarizing the test cases related to benchmark 2 comprising only up to 20 evolved product items, we can say that calculation times are always below 0.1s such that also many more types of explanations would never be a problem also in online scenarios. Also concerning benchmark 1 targeting at the calculation of all utilities for all products for a given profile the measured calculation times allow online scenarios. For the most extreme case, that is 100000 products, 16 dimensions, 16 item attributes and 16 user attributes we measured a calculation time of 78s and a peak memory consumption of roughly 3GByte. Note that 100000 product items is about five times more than Amazon Prime

³ <https://www.techradar.com/news/netflix-vs-amazon-prime-video-which-streaming-service-is-best-for-you>.

offers movie and series items in total. Also the maximal number of 16 utility dimensions is very high, and it is hard to imagine a product category where this would be exhausted. For more common scenarios calculation times and memory consumption are much lower, e.g. for the 10000 product test case involving 4 utility dimensions, 8 item attributes and 8 user attributes we measured a calculation time of 1.3s and a peak memory consumption of 71 MByte. Thus, ASP represents an attractive alternative to efficiently represent the complete recommendation logic in a declarative and highly compact way.

References

1. Anelli, V.W., et al.: Knowledge-aware and conversational recommender systems. In: Proceedings of the 12th ACM Conference on Recommender Systems, pp. 521–522. ACM (2018)
2. Dyer, J.S.: Multiattribute utility theory (MAUT). In: Greco, S., Ehrgott, M., Figueira, J.R. (eds.) Multiple Criteria Decision Analysis. ISORMS, vol. 233, pp. 285–314. Springer, New York (2016). https://doi.org/10.1007/978-1-4939-3094-4_8
3. Felfernig, A., Friedrich, G., Jannach, D., Zanker, M.: An integrated environment for the development of knowledge-based recommender applications. *Int. J. Electron. Commer.* **11**(2), 11–34 (2006)
4. Felfernig, A., Teppan, E., Friedrich, G., Isak, K.: Intelligent debugging and repair of utility constraint sets in knowledge-based recommender applications. In: Proceedings of the International Conference on Intelligent User interfaces (IUI), pp. 217–226. Springer, Berlin Heidelberg (2008). <https://doi.org/10.1145/1378773.1378802>
5. Friedrich, G., Zanker, M.: A taxonomy for generating explanations in recommender systems. *AI Mag.* **32**(3), 90–98 (2011)
6. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Answer set solving in practice. *Synth. Lect. Artif. Intell. Mach. Learn.* **6**(3), 1–238 (2012). Morgan and Claypool Publishers
7. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Kowalski, R., Bowen, K. (eds.) Proceedings of the Fifth International Conference and Symposium of Logic Programming (ICLP 1988), pp. 1070–1080. MIT Press, Cambridge (1988)
8. Ielpa, S.M., Iiritano, S., Leone, N., Ricca, F.: An ASP-based system for e-tourism. In: Erdem, E., Lin, F., Schaub, T. (eds.) LPNMR 2009. LNCS (LNAI), vol. 5753, pp. 368–381. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04238-6_31
9. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender Systems: An Introduction. Cambridge University Press, Cambridge (2010)
10. Leite, J., Ilić, M.: Answer-set programming based dynamic user modeling for recommender systems. In: Neves, J., Santos, M.F., Machado, J.M. (eds.) EPIA 2007. LNCS (LNAI), vol. 4874, pp. 29–42. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-77002-2_3
11. McNee, S.M., Riedl, J., Konstan, J.A.: Being accurate is not enough: how accuracy metrics have hurt recommender systems. In: CHI 2006 Extended Abstracts on Human Factors in Computing Systems, pp. 1097–1101. ACM (2006)
12. Teppan, E., Felfernig, A.: Impacts of decoy elements on result set evaluations in knowledge-based recommendation. *Int. J. Adv. Intell. Paradigms* **1**, 358–373 (2009)

13. Teppan, E.C., Felfernig, A.: Calculating decoy items in utility-based recommendation. In: Chien, B.C., Hong, T.P., Chen, S.M., Ali, M. (eds.) Next-Generation Applied Intelligence. IEA/AIE 2009. Lecture Notes in Computer Science, vol. 5579, pp. 183–192. Springer, Berlin, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02568-6_19
14. Teppan, E.C., Zanker, M.: Decision biases in recommender systems. *J. Internet Commer.* **14**(2), 255–275 (2015)



Tailoring Random Forest for Requirements Classification

Andreas Falkner^(✉), Gottfried Schenner^(ID), and Alexander Schörghuber^(ID)

Siemens AG Österreich, Vienna, Austria

{andreas.a.falkner,gottfried.schenner,alexander.schoerghuber}@siemens.com

Abstract. Automated and semi-automated classifications of requirements (type and topics) are important for making requirements management more efficient. We report how we tailored a random forest approach in the EU funded project OpenReq, aiming for sufficient quality for practical use in bid projects. Evaluation with thirty thousand requirements in English from nine tender documents for rail automation systems in various countries show that user expectations are hard to meet.

Keywords: Random forest · Evaluation · Industrial application · Requirement classification · Text categorization

1 Introduction

Requirements management for large projects is a time-consuming and error-prone task which can be supported by artificial intelligence [10]. In the Horizon 2020 project OpenReq¹, we developed several solution approaches and evaluated them with data from bid projects in the domain of railway safety systems.

Requests for proposal (RFP) or tenders for large infrastructure systems are typically issued by national authorities and comprise natural language documents of several hundred pages with requirements of various kind (domain specific, physical, non-functional, references to standards and regulations, etc.). Preparing a proposal (bid) to answer a tender requires (1) to identify the requirements in the tender text and (2) to assign experts to assess the company's compliance to those requirements. The difficult part is the classification of the (real) requirements w.r.t. predefined topics (which are covered by the experts).

For both tasks, it is important to achieve a very high true positive rate (recall), because requirements which are not detected or are assigned to the wrong experts will not be assessed correctly and may lead to high non-compliance cost. On the other hand, the true negative rate shall also be high, so that unnecessary work is reduced.

The contribution of this work is twofold: Firstly, a new way to tailor the well-known random forest approach [5] by optimizing the model's configuration

¹ <http://openreq.eu/>.

to requirements classification in general. Secondly, its evaluation in the domain of rail automation (30,000 real-world requirements, 50 topics).

The remainder of this paper is structured as follows: In Sect. 2 we list previous approaches to solve this and similar problems. After presenting our solution in Sect. 3, we report on the evaluation results in Sect. 4. Section 5 summarizes the main outcome and its impact on the users.

2 Related Work

Many approaches to automatic text classification are not specific to requirements management. In the past they tended to be rule-based, but lately (supervised) machine learning has become increasingly popular [13].

Approaches specific to requirements classification vary in the preprocessing NLP pipeline and in their choice of used classifiers. For example, a micro-service for requirements classification developed in the OpenReq project uses Naïve Bayes classifiers [9]. [18] describes a NLP pipeline for extracting requirements from prescriptive documents and uses a SVM classifier to classify the requirements into disciplines.

[14] is an early paper on automatic topic categorization of requirements written in natural language using a bootstrapping approach with machine learning (Naïve Bayes). [17] uses automatic requirement categorisation in an industrial setting to support the review of large natural language specifications in the automotive domain.

Semantic approaches for text classification incorporate not only syntactic but also semantic information, e.g., provided by systems for automatic information and relation extraction [11]. For a survey of such approaches see [3]. [16] discusses the use of ontologies and semantic technologies in requirements management.

In contrast to new approaches which use pre-trained models, e.g. BERT [8], this work relies solely on traditional machine learning approaches and a model which has been in industrial use for two years.

3 Solution

Text categorization labels paragraphs of natural language documents with pre-defined categories (or classes). It is a typical application of supervised learning which relies on an initial set of labelled instances used for training [1]. We use binary classification for *type classification* (whether an instance is a requirement or not) and multi-label classification for *topics* (an instance can be assigned to either zero or one or several topics). For example, an input instance to classification is the paragraph “The power supply shall consist of the two sources: one main and one for backup.” and the corresponding output could be *requirement* = *yes* for binary type classification and *topics* = {*Power, Diesel*} for multi-label topic classification. Internally, we implemented multi-label classification as multiple isolated binary classification problems [21].

Our solution comprises: (1) a *Random Forest* approach which is a proven classifier for text categorization [1], (2) text preprocessing such as tokenization, n-grams, stop word removal and reduction of word inflections, (3) a feature engineering stage which includes calculating feature weights and selecting relevant features [2], and (4) various sampling strategies to overcome problems with imbalanced data [12].

For tailoring this solution, we evaluated various combinations for these steps and identified the most promising model configuration for application to bid projects (for more details, see [19]):

- As a sampling strategy, we analyzed random under-sampling (RUS) [12], SMOTE [6] and no rebalancing. RUS showed superior performance. For training, we apply RUS ten times with 10 different random seeds, resulting in ten training sets. One model is trained per training set and all ten models are finally aggregated using majority vote.
- To remove word inflection, lemmatization (StanfordNLP [15]) [20] and stemming (Porter Stemmer) [20] were compared. Although evaluation revealed that using the lemmatizer increases performance, we decided on the stemmer because of its less restrictive software licence.
- We evaluated usage of tokens based on n-grams, $n \in \{1\}$, $n \in \{1, 2\}$, ..., $n \in \{1, 2, 3, 4, 5\}$. This parameter has no significant influence on performance – therefore uni-grams are used to keep feature space small.
- Different feature weights were compared: set of words [21], term frequency (TF) [20], TF-IDF [20] and (R)TF-IGM [7]. As this parameter did not show significant influence on performance, we selected TF because of its algorithmic simplicity.
- Using a stop-word list [20] from the Natural Language Toolkit² increased performance.
- The following common feature selection methods were evaluated: information gain (IG), χ^2 and term frequency (TF) [21]. TF showed good results and a fast runtime. The algorithm is configured to keep 1,300 features.

Additionally to the described configurations above, a user can set a threshold for positive classification before starting the predictor. Only if the built model's probability for a requirement is greater than or equal to the given threshold, the requirement is classified as positive instance. By that, the priority of true positives versus true negatives can be decided [23].

4 Evaluation

After having tailored the random forest approach including text preprocessing, we evaluated it using previous bid projects provided by the bid group. In addition to a quantitative evaluation, we did a small field study with three experts (unstructured interviews, application to a new, yet unlabelled bid project).

² <http://www.nltk.org/>, accessed 09.01.2020.

4.1 Data Set

The data set used for evaluation comprises the text paragraphs of nine tender documents. All of them were written in English (most of them translated from a native language). Each entry was labelled by experts as a requirement (or non-requirement) and assigned to relevant topics (mostly between one and three, out of 52 potential topics). Thereafter, we randomly chose six documents as training data, resulting in a 47% test data split. Table 1 lists the numbers of requirements, non-requirements, and assigned topics for training data as a whole and for test data separately for each project³ and in total.

Concerning type classification, 14,714 out of 17,556 potential requirements are labeled as requirement, leading to a prevalence of 84%.

From 52 potential topics, 50 occur in the training data, and 34 occur in the test data. Depending on prevalence in the training data, we selected three groups of 5 topics each: A (5/6) comprises all topics which occur more than 1,000 times in the training data (and at least once in the test data). For B (5/17), we chose – from all topics which occur more than 200 times in the training data – those which occur at least 500 times in the test data or in all three test projects. For

Table 1. Test data – numbers of types and topics

		Training data	Test data	Project 1	Project 2	Project 3
Total		17,556	8,256	978	6,465	813
Req		14,714	6,923	867	5,336	720
Non-Req		2,842	1,333	111	1,129	93
52 topics		20,288	10,479	867	8,892	720
PM	A	4,710	1,663	683	432	548
IXL		2,073	338	0	338	0
MMI		1,590	1,287	0	1,287	0
Power		1,448	405	47	299	59
BidManager		1,141	56	0	56	0
LED	B	507	361	116	132	113
SCADA		476	1,317	0	1,317	0
Diagnosis		422	820	0	820	0
SystemMgmt		400	548	0	548	0
Engineering		203	1,328	0	1,328	0
GSMR	C	172	181	13	168	0
Commissioning		39	70	0	70	0
PIS		27	223	0	223	0
Diesel		8	135	0	135	0
EHS		8	195	0	195	0

³ Project names are confidential – therefore we use numbers 1 to 3.

C (5/27), we selected – from all topics which occur less than 200 times in the training data – those which occur most often in the test data.

4.2 Metrics

We use (standard) metrics which help to directly judge user benefit: recall (sensitivity, true positive rate, TPR), specificity (true negative rate, TNR) [22], receiver operating characteristics (ROC) curve analysis [4] and custom metrics for estimating time savings. For bid projects, a high recall is very important in order to reduce risk of high non-compliance cost due to ignorance of information. Specificity, on the other hand, is important to avoid unnecessary work due to wrongly assigned topics. All metrics are micro-averaged, i.e., summing all quantities and then calculating the metrics on the sum. This leads to a combined metric for all test data and a combined metric for multi-labels per topic.

For estimating the time savings, we compare our solution to the decisions by a requirements manager, using the metrics defined by Eqs. 1 and 2, which are based on the time to comprehend a requirement ($t_{analyze}$), the time to change a label (t_{change}) and standard evaluation quantities: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). Assuming that an expert does not make any mistakes, he or she needs to analyze each requirement and set only the positive labels. In the automated approach, true positives need not be analyzed nor set, but additional work is necessary: For type classification (td_{type}), true and false negatives are still analyzed by the requirements manager and false negatives are set to positive in order to get TPR high – this has no effect in Eq. 1. False positives must be changed to negative by topic experts. For topic classification (td_{topic}), false positives and negatives are corrected by the topic experts during assessment. If the values for $t_{analyze}$ and t_{change} are known (e.g., as seconds per requirement on average) then the difference in hours can be calculated.

$$td_{type} = (FP - TP) \times t_{change} - (TP + FP) \times t_{analyze} \quad (1)$$

$$td_{topic} = FP \times t_{analyze} + (FP - TP) \times t_{change} \quad (2)$$

4.3 Type Classification

We used various thresholds (20%, 30%, ..., 80%) to get a feeling of the balance of TPR and TNR – see the ROC curve in Fig. 1a. Projects 1 and 3 perform very well, probably because they have similar properties as projects in the training set (same author, different stations on the same railway line).

In our interviews, the requirements managers turned out to be very risk-averse. Therefore, they prefer a very high TPR (e.g., 99%, as achieved with threshold 20%) and accept the relatively low TNR of 72% compared to threshold 50% (with fairly balanced TPR of 91% and TNR of 87%) – see Table 2. Assuming $t_{analyze} = 30$ s and $t_{change} = 5$ s, the time savings are more than 60 working hours for the three test projects. This equates to savings of approximately one working day for each thousand requirements which was confirmed in a field study with a new bid project.

Table 2. Evaluation results – TPR, TNR and estimated time savings

	Threshold 50%				Threshold 20%			
	Micro-avg	P1	P2	P3	Micro-avg	P1	P2	P3
TPR (%)	91	99	89	98	99	100	99	100
TNR (%)	87	98	85	98	72	97	68	94
td_{type} (h)	-62	-8	-47	-7	-69	-8	-54	-7

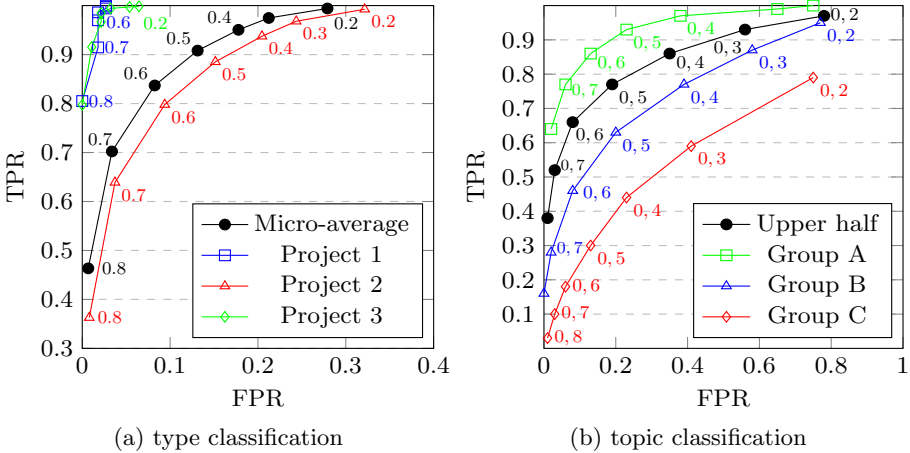


Fig. 1. ROC curves

4.4 Topic Classification

Prevalence of topics is very low – the average in the training data was lower than 2% and even for the 5 most common topics only around 6%. The ROC curve in Fig. 1b shows that prediction quality on average (for the upper half of topics) is not as good as for type classification. The groups B and C from Table 1 perform much worse than group A (with comparably higher prevalence).

In our interviews, the requirements managers preferred a threshold of 50% - see Table 3. However, they judged the achieved TPR of 73% (micro-average of all topics occurring in at least one test project) as too low for practical use. Even the TPR of 77% (and TNR 81%) for the upper half of topics was not sufficient, as nearly 2,100 topic assignments are missing and more than 16,300 assignments are wrong. Again, projects 1 and 3 perform much better than the more typical project 2. The new project from the field study performed similar to the latter.

Although the requirements managers do not need to spend any time for topic assignment, the metric td_{topic} estimates an additional effort of 127 h for the necessary manual adjustments by topic experts. Only for project 3 or with a high threshold can time savings be achieved.

Table 3. Evaluation results – TPR and TNR at threshold 50%

	TPR				TNR			
	Micro-avg	P1	P2	P3	Micro-avg	P1	P2	P3
34 topics in test	73	98	69	100	81	92	81	98
Upper half	77	99	73	100	81	97	80	98
5 topics A	93	100	89	100	77	96	75	97
5 topics B	63	99	61	100	80	99	78	99
5 topics C	30	38	30		87	76	87	

5 Conclusion

We chose the random forest approach for requirements classification because it is easier to maintain and deploy than advanced deep learning solutions. Training is less expensive and can be done on local servers.

The results were fairly good for type classification and topics with a prevalence $> 5\%$ (better than, e.g., an alternative approach based on Naïve Bayes). Application in a field study showed a high potential for reducing efforts for requirements managers (e.g., 80% of the time for type classification). However, improvements – especially for topics with a low prevalence $< 3\%$ – are necessary to fulfil the users’ demand for high TPR and TNR, i.e., both $\gg 95\%$.

Acknowledgments. The work presented here has been conducted in the scope of the Horizon 2020 project OpenReq, supported by the European Union under Grant Nr. 732463.

References

1. Aggarwal, C.C.: Machine Learning for Text. Springer, Cham (2018). <https://doi.org/10.1007/978-3-319-73531-3>
2. Allahyari, M., et al.: A brief survey of text mining: classification, clustering and extraction techniques. CoRR abs/1707.02919 (2017)
3. Altimel, B., Ganiz, M.C.: Semantic text classification: a survey of past and recent advances. Inf. Process. Manag. **54**(6), 1129–1153 (2018)
4. Bekkar, M., Djemaa, H.K., Alitouche, T.A.: Evaluation measures for models assessment over imbalanced datasets. J. Inf. Eng. Appl. **3**(10), 27–38 (2013)
5. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
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. Chen, K., Zhang, Z., Long, J., Zhang, H.: Turning from TF-IDF to TF-IGM for term weighting in text classification. Expert Syst. Appl. **66**, 245–260 (2016)
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint [arXiv:1810.04805](https://arxiv.org/abs/1810.04805) (2018)

9. Falkner, A., Palomares, C., Franch, X., Schenner, G., Aznar, P., Schoerghuber, A.: Identifying requirements in requests for proposal: a research preview. In: Knauss, E., Goedicke, M. (eds.) REFSQ 2019. LNCS, vol. 11412, pp. 176–182. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-15538-4_13
10. Fucci, D., et al.: Needs and challenges for a platform to support large-scale requirements engineering. A multiple case study. CoRR abs/1808.02284 (2018)
11. Gupta, P., Schütze, H., Andrassy, B.: Table filling multi-task recurrent neural network for joint entity and relation extraction. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 2537–2547 (2016)
12. Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., Bing, G.: Learning from class-imbalanced data: review of methods and applications. *Expert Syst. Appl.* **73**, 220–239 (2017)
13. Kadhim, A.I.: Survey on supervised machine learning techniques for automatic text classification. *Artif. Intell. Rev.* **52**(1), 273–292 (2019)
14. Ko, Y., Park, S., Seo, J., Choi, S.: Using classification techniques for informal requirements in the requirements analysis-supporting system. *Inf. Softw. Technol.* **49**(11–12), 1128–1140 (2007)
15. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Association for Computational Linguistics (ACL) System Demonstrations, pp. 55–60 (2014)
16. Moser, T., Winkler, D., Heindl, M., Biffi, S.: Requirements management with semantic technology: an empirical study on automated requirements categorization and conflict analysis. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 3–17. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21640-4_3
17. Ott, D.: Automatic requirement categorization of large natural language specifications at Mercedes-Benz for review improvements. In: Doerr, J., Opdahl, A.L. (eds.) REFSQ 2013. LNCS, vol. 7830, pp. 50–64. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37422-7_4
18. Pinquié, R., Véron, P., Segonds, F., Croué, N.: Requirement mining for model-based product design. *Int. J. Product Lifecycle Manag.* **9**(4), 305–332 (2016)
19. Schörghuber, A.: Classification of requirements in the tender process. Master's thesis, University of Technology Vienna, Vienna (2019)
20. Schütze, H., Manning, C.D., Raghavan, P.: Introduction to Information Retrieval, vol. 39. Cambridge University Press, Cambridge (2008)
21. Sebastiani, F.: Machine learning in automated text categorization. *ACM Comput. Surv. (CSUR)* **34**(1), 1–47 (2002)
22. Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. *Inf. Process. Manag.* **45**(4), 427–437 (2009)
23. Tosun, A., Bener, A.: Reducing false alarms in software defect prediction by decision threshold optimization. In: Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement, pp. 477–480 (2009)



On the Design of a Natural Logic System for Knowledge Bases

Troels Andreassen¹(✉), Henrik Bulskov¹, and Jørgen Fischer Nilsson²

¹ Computer Science, Roskilde University, Roskilde, Denmark
{troels,bulskov}@ruc.dk

² Mathematics and Computer Science, Technical University of Denmark,
Lyngby, Denmark
jfni@dtu.dk

Abstract. Natural logics are logics that take form of stylized natural language sentences within a selected fragment of natural language. Natural Logics are at the same time formal logics with a well-defined syntax and semantics. Therefore, natural logics may be advanced as knowledge base logics enhancing explainability of query answers. This paper is concerned with a natural logic, NATURALOG, having been proposed as a deductive knowledge base language. The paper briefly reviews and brings together in compact form the main points in our previously but separately published design proposals, systems functionalities and implementation principles.

Keywords: Natural logic · Knowledge base systems · Deductive querying · Life science applications

1 Introduction

In a historical perspective there are two development lines in logic, that is logic-of-language and mathematical logic. The logic-of-language tradition dates back to Aristotle and went through developments during the medieval times until the end of the 19th century. This development line was halted and largely abandoned by the advent of quantified predicate logic due mainly to G. Frege and B. Russell. Predicate logic and related logics, then, have become foundational concepts as well as important tools in computer science, in particular in computational logic, e.g. with logic programming.

However, the logic-of-language tradition recently has attracted renewed interest in connection with attempts to ease communication with computers. This paper discusses natural logics [9, 12] that are rooted in the logic-of-language tradition and describes undertakings aimed at adopting and adapting natural logics for logical knowledge base systems. The following sections briefly surveys and discusses the various aspects of design principles and implementation methods described in the here chronologically listed range of papers [1–5, 10, 11] followed up by [6, 7] forthcoming in 2020.

Our design concern throughout is to obtain a useful trade-off between expressivity and computational tractability, having in mind also the requirements for potentially useful application domains in the design. Although the proposed natural logic is meant as a general purpose specification language for real-world domains, we have had in mind in particular applications within the life sciences as it appears in the mentioned publications.

2 Designing a Natural Logic

Our natural logic proposal, termed NATURALOG, takes as point of departure syllogistic logic from the Aristotelian tradition, cf. [8,11]. This means that the basic so-called categorical sentence forms are **every C is D** and **some C is D** , where C and D are class- or concept terms. In the simplest cases C and D are common nouns representing classes aka concepts. Accordingly, these forms known as copula forms express, respectively, a subclass relationship and a class-class overlap relationship. In the following, for the copula “is” we follow the conventions in computer science and write “isa”. With the form **every C isa D** (or in convenient short form simply **C isa D**), one can specify hierarchically- as well as non-hierarchically structured formal ontologies as partial orders, with the isa relationship being transitive.

In addition to these affirmative sentence forms the old syllogistic logic also comprises negative forms as mentioned in [11], also known from the square-of-opposition. However, at present we refrain from admitting negative statements in a knowledge base itself, resorting instead to negation-as-non-provability as known from databases and logic programming.

2.1 Beyond Copula Forms: General Relationships

In addition to the copula isa in NATURALOG one can introduce transitive verbs (i.e. verbs taking a linguistic object) as one pleases. To this end in [2,3,11] we introduced the more general sentence forms with a verb R

$$(\text{every} \mid \text{some}) C R (\text{every} \mid \text{some}) D$$

giving four determiner constellations. The verb R represents a binary relationship between the two concepts represented by the subject and object. As convenient default form for the most common sentences in knowledge bases we propose

$$C R D \text{ for the full form } \text{every } C R \text{ some } D$$

Example: **persons like pets** is shorthand for **every person likes some pet** and **some persons drink beer** is shorthand for **some persons drinks some beer**. Actually, in NATURALOG we ignore linguistic inflection rules as seen in the following examples. For the **some** form we have

$$\text{some } C R D \text{ for } \text{some } C R \text{ some } D$$

In predicate logic *every C R some D* is construed as

$$\forall x(Cx \rightarrow \exists y(Rxy \wedge Dy))$$

and *some C R some D* is

$$\exists x(Cx \wedge \exists y(Rxy \wedge Dy)), \text{ which is equivalent to } \exists x\exists y(Cx \wedge Dy \wedge Rxy)$$

However, we stress that NATURALOG sentences are not translated into predicate logic in our systems proposal as explained in Sect. 3 and 4.

2.2 Compound Terms

In addition to the concepts given by common nouns in NATURALOG one can form expressions for creation of new concepts by attachment of restrictive modifiers to common nouns as in the sample sentence

betacell isa cell that produce insulin

The restrictive modifiers may take form of relative clauses as in the compound concept term *cell that produce insulin* or prepositional phrases such as *cell in gland*. Both forms semantically consist of a relationship given as a verb or as a preposition followed by a concept. Provision is made for nesting of such constructs reflecting the usual recursive syntax for modifiers in natural language phrases.

There are other forms of restrictive nominal modifiers in natural language, in particular adjectives (including participles such as “increased”) and noun-noun-compounds such as “heart disease”. The incorporation of these modifiers into natural logic is more problematic and is postponed since they, unlike the above ones, do not directly provide a modifying relationship. As a temporary solution noun-noun compounds may be rewritten using a relative clause, so that for instance “bacteria infection” would become *infection that is-caused-by bacteria*.

Verbs may also be modified restrictively using an adverbial prepositional phrase as in the verb form *produce in gland* as a restriction of the verb *produce*. Incorporation of this useful feature, which falls outside the simple predicate logical explanation in Sect. 2.1, is thoroughly discussed in [6].

The syntax for the current version of NATURALOG is specified in the form of a BNF grammar in [6]. In order to ensure that there be no structural ambiguities, parentheses are enforced in one production rule for stipulating the intended recursive phrase structure.

In [3] we discuss various language extensions intended for promoting the usability of NATURALOG by approaching some common forms in natural language. Examples are appositions and conjunctions. We distinguish semantically conservative extensions and non-conservative ones. The former ones do not extend the semantic coverage. The order of the natural logic sentences in a knowledge base is logically irrelevant; the sentences are syntactically independent of each other.

2.3 Active-Passive Voice and Existential Import

Now, consider the active voice sentence *betacell produce insulin* or in full form *every betacell produce some insulin*. The corresponding passive voice sentence is *some insulin is-produced-by some betacell*, where *is-produced-by* represents the inverse relation of *produce*. Although this latter sentence follows intuitively, the sentence does not follow logically, because in predicate logic the denotation of a monadic predicate may well be empty. This problem is overcome by appealing to the existential import principle known from the Aristotelian logic tradition, cf. [6, 11]. This principle declares that all mentioned classes be non-empty without being specific about any member entity. As a special consequence, the presence of a copula sentence of the form [every] C isa D implies availability also of the weakened converse sentence form *some D isa C* .

2.4 Remarks on Natural Logic and Description Logic

Today's most common logics for ontologies and knowledge bases are presumably the various description logic dialects. Both NATURALOG and description logics are examples of variable free logics covering small but useful fragments of predicate logic. A key difference between these two logics is that description logics offer sentences in copula form only (at the so-called T-box level of concepts), which seems awkward from the point of conventional use in natural language. This is in disagreement with common formulations in natural language. Another difference is that description logics have to resort to awkward reformulations for sentences beginning with the determiner “some”, cf. the discussion of active-passive forms in the previous section. A further comparison of the two logics is given in [6].

The endorsing in the natural logic of non-copula sentences (with verbs fetched from the target application) agrees well with an entity-relationship model view of a knowledge base: A NATURALOG knowledge base typically takes the form of an ontology formed by the stated copula sentences extended with non-copula sentences connecting concepts across the ontology with relations expressed by transitive verbs. This view further invites the introduction of a distinction between definitional and observational (i.e. empirical) statements mentioned in the next section.

3 The Metalogic Framework for NATURALOG

So far, NATURALOG may be conceived simply as a “sugared” fragment of predicate logic for describing the application domain of discourse. As a next step appropriate proof rules admitting computational derivation of logical consequences as NATURALOG sentences are to be introduced. Importantly, these rules are to be applied directly to natural logic sentences and terms, rather than to their would-be predicate logical translations.

Such rules enable deductive querying of the knowledge base giving answer results in the form of classes and more generally compound terms. In principle

deductive querying is achievable by introduction of variables ranging over the terms in the natural logic as the metalogical variable X the sample query form

X isa cell that produce hormone

supposed to give the answer `betacell`. These term variables should not be confused with the quantified entity- or individual variables of predicate logic that range over entities in the application domain of discourse. We account formally for these variables by introduction of a metalogic in which NATURALOG becomes embedded.

As metalogic we can choose a “domesticated” form of predicate logic: In [10, 11] we suggested DATALOG to this end, and [6] gives elaborate description of the metalogic inference engine, succeeding a more compact presentation in [5]. Recall that DATALOG consists of definite clauses without compound terms and enjoys decidability.

Let us exemplify the encoding of our natural logic into DATALOG: The sentence `betacell produce insulin` becomes the atomic metalogic clause

proposition(every, betacell, produce, insulin)

where the natural logic terms formally appear as, and are treated as, constants.

The sentence `betacell isa cell that produce insulin` becomes in the metalogic representation

proposition(every, betacell, isa, cell-that-produce-insulin)

where `cell-that-produce-insulin` is a new simple concept term that becomes defined by the following pair of defining metalogic clauses

definition(cell-that-produce-insulin, isa, cell)

definition(cell-that-produce-insulin, produce, insulin)

The decomposition applies recursively to nested concept terms.

The distinction between definitional and non-definitional contributions in the decomposition is internal to the system. However, [11] hints at further introducing an external epistemic distinction between definitional and observational sentences in the knowledge base.

3.1 The Encoded Knowledge Base as Graph

The metalogic knowledge base representation may be conceived as a labeled graph whose nodes are concept terms, and whose directed arcs represent relationships with accompanying determiners (quantifiers). The graph picture further supports the conception of the knowledge base as an extended ontology. The concept terms present in the KB sentences are uniquely represented as nodes in the graph across sentences. Moreover the graph view helps visualizing pathway querying, cf. Sect. 5. In the decomposition of sentences into clauses there is no loss of information.

4 Design of Inference Engine

Natural logics use high level inference rules reflecting “intuitive” rules applied by humans when reasoning with descriptions in natural language. This adds to the explainability of the deductive reasoning and hence the query processing. These rules are now to be formalized in the metalogic, exploiting the decomposed and encoded NATURALOG sentences.

As a key principle, answers to queries stated to the knowledge are computed by use of the inference rules. We refer to [5, 6] for the rules we apply for NATURALOG. These papers also contain references to the background literature dealing with deductive reasoning in natural logics.

We present here using DATALOG only a few rules. As key rules there are the so-called monotonicity rules:

$$\begin{aligned} \text{proposition}(\text{every}, C, R, D_{\text{super}}) &\leftarrow \\ &\text{proposition}(\text{every}, C, R, D) \wedge \text{proposition}(\text{every}, D, \text{isa}, D_{\text{super}}) \\ \text{proposition}(\text{every}, C_{\text{sub}}, R, D) &\leftarrow \\ &\text{proposition}(\text{every}, C, R, D) \wedge \text{proposition}(\text{every}, C_{\text{sub}}, \text{isa}, C) \end{aligned}$$

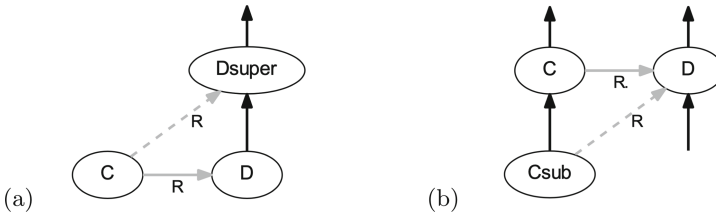


Fig. 1. Monotonicity rules: (a) inheritance and (b) generalization. Dashed relations are inferred.

The graphs for these are shown in Fig. 1. One observes that the latter rule provides “property inheritance”. Further, one may observe that the transitivity rule for *isa* obtains with the special case of *R* being instantiated to *isa*. As another distinctive feature we introduce a rule for obtaining the corresponding passive voice sentence from a given active voice sentence. This implies in particular that the sentence form *C isa D* gives rise to *some D isa C* as mentioned in Sect. 2.3 besides giving the weakened *some C isa D* by means of appropriate rules. In Sect. 5.1 we mention the potential for extension with “non-logical” application-specific inference rules.

4.1 Materialization of Deductive Closure

In [7] we propose that the part of the deductive closure of a knowledge base being relevant for query answers of a knowledge base is computed and stored in advance in a compilation process jointly involving the sentences. This means that

sentences and terms potentially appearing in a query answer is made present in advance in the compiled knowledge base.

Furthermore, in [7] (forthcoming 2020) we elaborate a version of the inference engine where `DATALOG` is replaced by relational database query operations. This enables use of a database system for efficient retrieval of sentences in the knowledge base, and in addition inference computations are made algorithmically more efficient by “bulk processing” applying database join operations.

The recursive `NATURALOG` syntax generally admits infinitely many terms. However, only a finite subset of these are known to have a non-empty denotation in the form of subconcepts in the knowledge base – namely either by their being explicitly present or by being a superclass of such a mentioned term.

5 Systems Functionalities

A range of systems functionalities can be obtained on basis of the relevant deductive closure computed by the inference rules. First of all there are answers in the form of sets of terms from instantiation of metalogical variables in query forms exemplified by

`proposition(every, X, produce, hormone)`

supposed to give as answer cells that produce hormone, such as betacells. Notice here that computation of such answers in general draws on inference rules, say, for combining the sentence `betacell produce insulin` with the sentence `insulin isa hormone` in a monotonicity inference rule. Query answer terms may well be compound terms stemming from a given sentence or having been computed in the compilation process.

So far, we accept only affirmative sentences in a knowledge base. Negative sentences may be accepted as query sentences in the form `no C R D` being logically contrary to `every C R D` and contradictory to `some C R D`, with a supporting inference rule appealing to negation by non-provability.

The graph conceptualization of a `NATURALOG` knowledge base as (usually) one coherent graph invites path-finding operations for retrieving shortest paths between two given terms as discussed and exemplified in [1,3,4]. Path-finding is of particular interest for tracing pathways in life-science knowledge bases.

5.1 Application-Specific Query Inference Rules

In our two level logic setup the natural logic level describes the application domain of discourse and the metalogic level prescribes the computing with natural logic sentences and terms. This opens for introducing application specific rules in the metalogic. For instance, one easily introduces a rule providing a verb, say, “causes” with the property of transitivity of the underlying relation.

As another example, [6] describes an additional metalogic rule for computing the commonalities of two given terms. When asking for instance about common properties of the two concepts `alphacell` and `betacell`, the deduced answer may

comprise informative compound terms such as **cell that produce hormone**. More sophisticated general rules may afford the computing of analogies, asking for instance which concept is related to **alphacell** as **insulin** is related to **betacell**.

Inference rules may be introduced to verify ad hoc consistency requirements formulated as rules expected to yield empty query answers in case of consistency fulfilment as known from logic programming.

6 Conclusion and Open Problems

The described natural logic with the accompanying realization principles attempts to strike a balance between on one hand language expressivity and interesting computational functionalities and on the other hand an acceptable computational tractability. NATURALOG covers basic essential application domain demands within the considered life-science domains, but additional useful features are to be included in coming versions. Among the possible semantical extensions let us just mention exception handling for non-monotonic blocking of unrestricted inheritance of properties, and introduction of generalized quantifiers such as “most” and “few”.

It remains to be verified that computational tractability can be obtained with the suggested relational database implementation, when scaling up to interesting large size knowledge bases.

An interesting but highly challenging problem is to conduct a computer-assisted, if not completely automatic, translation of essential parts of given natural language descriptive texts into NATURALOG. This complex and difficult problem of computationally extracting natural logic sentences from descriptions in free natural language is touched in [1,3,4]. To this end a syntactic-semantic analysis using NATURALOG as target language might be ameliorated by inductive machine learning methods. Eventually, more expressive versions of natural logic may come into use directly as logical specification languages in natural science domains.

References

1. Andreasen, T., Bulskov, H., Nilsson, J.F., Jensen, P.A.: Computing pathways in bio-models derived from bio-science text sources. In: Proceedings of the IWB-BIO International Work-Conference on Bioinformatics and Biomedical Engineering, Granada, April, pp. 217–226 (2014)
2. Andreasen, T., Bulskov, H., Nilsson, J.F., Jensen, P.A.: A system for conceptual pathway finding and deductive querying. Flexible Query Answering Systems 2015. AISC, vol. 400, pp. 461–472. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-26154-6_35
3. Andreasen, T., Bulskov, H., Nilsson, J.F., Jensen, P.A.: Partiality, Underspecification, and Natural Language Processing, chapter A Natural Logic for Natural-Language Knowledge Bases. Cambridge Scholars, Newcastle upon Tyne (2017)

4. Andreassen, T., Bulskov, H., Jensen, P.A., Nilsson, J.F.: Pathway computation in models derived from bio-science text sources. In: Kryszkiewicz, M., Appice, A., Ślęzak, D., Rybinski, H., Skowron, A., Raś, Z.W. (eds.) ISMIS 2017. LNCS (LNAI), vol. 10352, pp. 424–434. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-60438-1_42
5. Andreassen, T., Bulskov, H., Jensen, P.A., Nilsson, J.F.: Deductive querying of natural logic bases. In: Cuzzocrea, A., Greco, S., Larsen, H.L., Saccà, D., Andreassen, T., Christiansen, H. (eds.) FQAS 2019. LNCS (LNAI), vol. 11529, pp. 231–241. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-27629-4_22
6. Andreassen, T., Bulskov, H., Nilsson, J.F., Jensen, P.A.: Natural logic knowledge bases and their graph form, p. 34. (2020). Submitted to journal (under review)
7. Andreassen, T., Bulskov, H., Nilsson, J.F.: A natural logic system for large knowledge bases. In: The 30th International Conference on Information Modelling and Knowledge Bases, Ejc 2020, Hamburg, Germany, 8–12 June 2020 (2020)
8. Klima, G.: Natural logic, medieval logic and formal semantics. *Magyar Filozófiai Szemle* **54**(4), 58–75 (2010)
9. Moss, L.S.: Syllogistic logics with verbs. *J. Log. Comput.* **20**(4), 947–967 (2010)
10. Fischer Nilsson, J.: Querying class-relationship logic in a metalogic framework. In: Christiansen, H., De Tré, G., Yazici, A., Zadrozny, S., Andreassen, T., Larsen, H.L. (eds.) FQAS 2011. LNCS (LNAI), vol. 7022, pp. 96–107. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24764-4_9
11. Nilsson, J.F.: In pursuit of natural logics for ontology-structured knowledge bases. In: The Seventh International Conference on Advanced Cognitive Technologies and Applications (2015)
12. van Benthem, J.: Essays in logical semantics. In: *Studies in Linguistics and Philosophy*, vol. 29. D. Reidel, Dordrecht (1986)



Evaluation of Post-hoc XAI Approaches Through Synthetic Tabular Data

Julian Tritscher¹(✉), Markus Ring², Daniel Schlör¹(✉), Lena Hettinger¹(✉),
and Andreas Hotho¹(✉)

¹ University of Würzburg, Würzburg, Germany
{tritscher,schloer,hettinger,hotho}@informatik.uni-wuerzburg.de

² University of Coburg, Coburg, Germany
markus.ring@hs-coburg.de

Abstract. Evaluating the explanations given by post-hoc XAI approaches on tabular data is a challenging prospect, since the subjective judgement of explanations of tabular relations is non trivial in contrast to e.g. the judgement of image heatmap explanations. In order to quantify XAI performance on categorical tabular data, where feature relationships can often be described by Boolean functions, we propose an evaluation setting through generation of synthetic datasets. To create gold standard explanations, we present a definition of feature relevance in Boolean functions. In the proposed setting we evaluate eight state-of-the-art XAI approaches and gain novel insights into XAI performance on categorical tabular data. We find that the investigated approaches often fail to faithfully explain even basic relationships within categorical data.

Keywords: Explainable AI · Evaluation · Synthetic data

1 Introduction

Black box classifiers such as deep neural networks (DNNs) have been established as state-of-the-art in many machine learning areas. Even though they give strong predictions, their models contain lots of non-linear dependencies, causing their decisions to become untraceable. As a result, a branch of research in explainable artificial intelligence (XAI) has developed, aiming to give local explanations for single predictions of trained black box models in a post-hoc fashion [6].

Problem. While there are many approaches to acquire such explanations, no unified evaluation method has been proposed so far. While easily comprehensible domains like image and text classification use simple presentation of explanations [3] and user studies [6], XAI behavior on tabular data is largely unexplored.

Objective. Post-hoc XAI explanations are inherently approximations, giving simplified but not necessarily faithful insights into highly complex models [11].

Therefore, assessing the limits of these approaches is a central point of research interest. We take a first step to investigate post-hoc XAI performance on DNNs trained on tabular data, by developing a test setting for categorical tabular data, where feature relationships can often be expressed via Boolean functions.

Approach and Contribution. In this paper, we design synthetic datasets reflecting typical relationships of real-world data such as logical AND, OR and XOR connections between categorical attributes. We propose a definition of feature importance in Boolean functions in the context of XAI to generate gold-standard explanations for our datasets. Using this data, we present an evaluation setting for XAI approaches, that allows for evaluation of data with underlying complex feature relationships. This setting is used to analyze and compare eight state-of-the-art XAI approaches. Evaluation on an expert-annotated real dataset suggests that our results translate well to real data. We publish our datasets to facilitate comparison of XAI approaches in a standardized evaluation setting.

2 Related Work

Aside from subjective image explanations [3] and resource intensive user studies [6], several approaches have been used to evaluate XAI performance.

In [5, 9] model faithfulness of their approaches is evaluated by explaining predictions of inherently explainable linear models and comparing obtained explanations directly to the model. Performance of XAI approaches when explaining non-linear classifiers, however, can not be assessed in this evaluation setting.

Additionally, [9] also evaluate the faithfulness of their local approximation model by measuring if it corresponds to changed inputs in the same way as the classifier it approximates. This evaluation can, however, only be performed on XAI approaches that train a simplified classifier as a local approximation.

Perturbation-based evaluations, e.g. as used in [12], iteratively remove features with the highest relevance from data and re-classify. Explanations are rated higher, the faster the classification error increases. While perturbation-based evaluation can be applied to classification tasks where the removal of single features is expected to gradually impair the performance of the classifier, this assumption does not hold for categorical tabular data in general.

3 Investigated XAI Approaches

Perturbation-based explanation approaches mask or remove input features from data samples to observe the change in classifier output. While they pose no architectural constraints on the classifier, they are computationally intensive. LIME (Local Interpretable Model-agnostic Explanations) [9] uses perturbations to explore the classifier outputs locally around a given input. It trains a local linear classifier and uses the weights as scores of input feature relevance. Shapley Value sampling [4] is based on the Shapley value from cooperative game theory,

a unique solution for distributing an achieved score onto cooperating players, under a list of desirable criteria. Shapley value sampling approximates this NP-complete problem with a sampling approach, evaluating the output of possible feature value combinations through perturbation. (Kernel) SHAP (Kernel SHapley Additive exPlanation) [6] uses a custom kernel for the LIME XAI approach, in order to adapt LIME to approximate Shapley values.

Gradient-based XAI approaches use the gradient of a gradient descent based classifier to approximate explanations through few backpropagations, considerably saving runtime in comparison to perturbation-based approaches. For our evaluation, we use the implementations of [1]. Saliency maps [14] highlight the most influential pixels using a first order approximation of the absolute gradient of the predicted output with respect to the input for a specific data sample. Gradient×Input [13] builds on the Saliency approach, multiplying the signed result of Saliency with the corresponding input feature. Integrated Gradients [15] computes the average output gradients with respect to different inputs. Gradients are computed for values on a linear path between the data sample and an uninformative baseline input. ϵ -LRP (ϵ -Layerwise Relevance Propagation) [3] defines the relevance of a neuron as all influence it has on the neurons of the next layer, multiplied by these neurons’ activations for a specific data sample. In this work we use the reformulated implementation by [1]. DeepLIFT [12], like LRP, computes the relevance of a neuron by measuring the influence on neurons of the next layer, additionally subtracting the influence of an uninformative baseline.

4 Data Generation Approach

Since categorical attributes can be binarized (e.g. one-hot encoding) to Boolean features, we will focus on feature relationships modeled as Boolean functions.

In [7] a feature is considered influential in a sample if changing its value would also change the function output. While this is intuitive on some inputs, on others it assigns no influence to any feature. For example, consider the Boolean function $y = 0 \wedge 0 = 0$, where no single feature can be changed to change y . In the context of XAI, this would not allow to differentiate between the 0-inputs involved in the function and irrelevant features. To address this, we adapt the influence definition on basic Boolean operations (AND, OR, XOR) to assign influence to both features, if no single feature can be changed to change the function output. For more complex functions, we proceed as follows:

Definition 1. *Let the Boolean function y be represented by a Boolean binary expression tree [8]. For each data sample, we consider a child node c relevant to the explanation of its parent operation-node o , iff the value of the subtree formed by c , evaluated with respect to the sample, is relevant to the operation-node o .*

We thereby distribute the relevance of a complex function by decomposing it into its basic operations. We calculate their intermediate results for each data sample, propagating the relevance of the entire function through all basic operations down to its input features. The resulting explanations contain the input features that most determine the output of complex Boolean functions.

When assessing XAI performance, we have to take into account that non-matching explanations might be correct explanations of a weak classifier, instead of a poorly performing XAI approach. For this, we train our classifiers in a 5-fold stratified cross-validation setting, using only classifiers that reliably achieve 100% accuracy on training- and test-sets. Further, we generate synthetic datasets including every permutation of categorical attributes exactly once. This guarantees that the test-sets contain permutations not seen during training, ensuring that the classifier learned to perfectly generalize to the unseen test-data without sensitivity to irrelevant inputs.

Following these restrictions, we expect XAI approaches to give the highest scores to the relevant features. Thus, we consider a data sample to be correctly explained, if the top scoring features given by an XAI approach match the relevant features of the ground truth explanation.

5 Experiments

The following setup is used throughout all of our experiments.

Datasets are generated following the criteria of Sect. 4. We set a fixed dataset size of n binary features, for which we include every permutation once in the dataset, giving 2^n data samples. We then generate the label for each data sample with a Boolean function and generate the explanation of every data sample according to Sect. 4. Features that were not used in the generation of the label hereby act as noise that XAI approaches may falsely consider relevant. All following experiments use 12 binary features and $2^{12} = 4096$ data samples.

Classifier & XAI setup also follow Sect. 4. We encode our Boolean input data with the values 1 for True and -1 for False, and train a feed forward neural network with 5 layers, 20 neurons per layer, and ReLU activations in a 5-fold stratified cross-validation setting. The classifiers reliably achieve 100% accuracy on training- and test-sets for all evaluated datasets. For each cross-validation fold, we compute the explanations of the test-data. We repeat the evaluation 10 times per dataset, reporting the average over the results.

Baseline Some XAI approaches replace classifier input features with uninformative values to observe classifier behavior with missing information. We let LIME and SHAP extract their own baseline from the cross-validation training data, using k-means clustering with $k = 20$ for SHAP. For the gradient-based approaches, we use 0 as baseline value, as discussed in [2].

5.1 Evaluation of Basic Boolean Operations

We initially evaluate the behavior of XAI approaches on datasets where two features are linked by common Boolean operations AND (\wedge), OR (\vee) or XOR (\otimes) in Table 1. We find that most XAI approaches fail to fully explain even the linear Boolean AND and OR operations, with only LIME and SHAP finding the most relevant features for each sample. Results on the XOR dataset show that LIME, due to training a local linear model around the sample, fails to give good

explanations when the underlying local function is non-linear. SHAP appears to improve on LIMEs behavior, correctly matching the gold standard explanations with its kernel-based Shapley value adaptation of LIME.

Detailed Analysis. We take a closer look at the input permutations causing problems to the XAI approaches. We find that falsely explained samples for all gradient-based approaches and Shapley sampling on the AND and OR datasets are caused by issues with $y = 0 \wedge 0$ and $y = 1 \vee 1$. In this case, the mentioned approaches consider one of the two features as irrelevant, even though both features are equally important to the label. Additionally, the Saliency approach shows issues on unequal inputs, where one feature speaks against the prediction outcome. Since the gradient of the output with respect to this feature is negative, the Saliency’s absolute gradient causes this negatively influential feature to overshadow the relevant feature. On the non-linearly separable XOR dataset, all approaches show a similar amount of errors for each input permutation.

Table 1. XAI performance in percent correctly explained samples after Definition 1.

Approach	\wedge	\vee	\otimes	$(\otimes) \wedge (\otimes)$	Synthetic	Real
LIME	100.00	100.00	43.76	10.12	100.00	84.78
Shapley sampling	99.83	99.92	69.10	56.34	79.79	77.15
SHAP	100.00	100.00	100.00	95.01	98.12	93.34
Saliency	95.81	95.69	85.99	55.55	59.86	59.12
Gradient \times input	97.66	97.06	75.81	57.31	69.47	69.49
Integrated gradients	99.64	99.56	73.02	57.73	76.46	76.27
ϵ -LRP	97.66	97.06	75.84	57.34	69.48	69.71
Deeplift	99.67	99.59	73.28	58.83	75.48	76.77

5.2 Evaluation of Boolean Functions with Multiple Variables

Using our relevance definition (Definition 1), we investigate XAI performance on complex Boolean functions. Results of the function $y = (f_1 \otimes f_2) \wedge (f_3 \otimes f_4)$, shown as $(\otimes) \wedge (\otimes)$ in Table 1, indicate that XAI performance deteriorates with an increased number of relevant features involved. To test this, we create similar datasets with increasing numbers of variables that may impact the output label.

Linearly Separable Boolean Functions. Since XAI performance on basic operations suggests different XAI behavior on linear and non-linear Boolean functions, we first investigate XAI performance with increasing function complexity on linear functions. For this, we generate eight datasets using the function $y = (((f_1 \wedge f_2) \vee f_3) \wedge f_4) \vee \dots$), appending 3 to 10 relevant features as label.

To validate that the used datasets are linearly separable, we ensure that a linear Support Vector Machine can perfectly separate each dataset.

The average results on each dataset are shown in Fig. 1a. We find LIME to be able to fully explain all samples of datasets with up to 6 relevant features. Both LIME and SHAP are capable of explaining a large amount of samples in all tested datasets. Shapley sampling and all gradient-based methods show difficulties with explaining functions with more than 2 variables involved, with performance declining further with more than 3 variables. The small inclines in explanation score with increased function complexity may be explained by all datasets consisting of a total of 12 variables. This means that when 10 variables are involved in the function, randomly assigning the 2 non relevant variables the lowest scores may occur more often than with 6 relevant and irrelevant variables.

Non-linearly Separable Boolean Functions. We also evaluate performance on eight non-linearly separable datasets generated using the function $y = (((f_1 \otimes f_2) \wedge f_3) \otimes f_4) \wedge \dots$ with 3 to 10 relevant features used for label generation. As seen in Fig. 1b, all approaches show lower performance compared to the non-linear XOR dataset (see Table 1). While SHAP still maintains stronger performance than other approaches on non-linear datasets throughout the experiment, its performance deteriorates when more than 3 features influence the label. All other approaches show poor performance on all datasets. The fluctuation between scores with increasing complexity may be caused by the alternating label-generation: If the last operator in the outermost brackets of the function is an AND, then for all samples that evaluate to $x \wedge 0$ with the entire previous term $x = 1$, the only relevant variable for this sample is the last 0. Therefore XAI approaches only have to find the last variable 0 as explanation, simplifying the problem down to a basic AND operation for several input permutations.

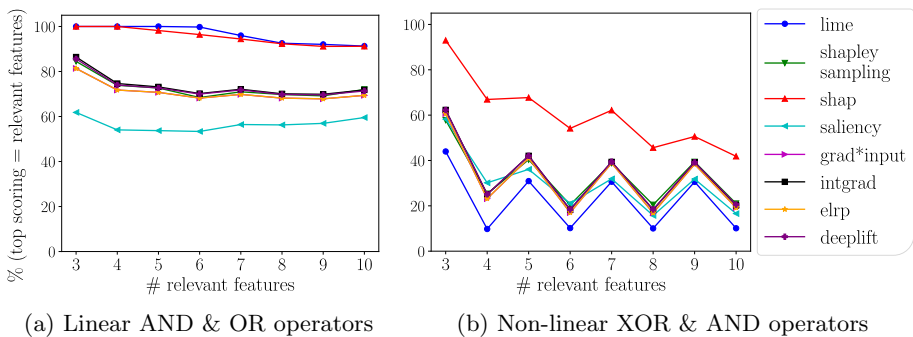


Fig. 1. XAI performance on datasets with multiple relevant features. Performance in percent of correctly explained samples according to Definition 1.

5.3 Application Scenario

Next, we choose a setting from the intrusion detection domain, to show that our findings can be applied to realistic settings. A method to synthetically create flow-based network traffic is proposed in [10], where a flow describes a network connection between two hosts and contains attributes like transport protocol and TCP-flags. In this setting, only flows which represent TCP traffic are allowed to set any TCP-flags, creating the task of validating whether samples resemble valid network traffic. While transport protocol and TCP-flags can be represented as binary attributes with a set of predefined rules, the complexity of the problem is low enough to create expert-annotated labels. In this experiment, we use N-WGAN-GP from [10] to create 2048 correct and 2048 incorrect flows. Each flow is represented by six categorical (*TCP flags*) and two categorical, one-hot-encoded features (*weekday, protocol*), six numerical features (*bytes, packets, duration, time, source- and destination-port*) and two values encoded with multiple numeric features (*IP-addresses*). The six *TCP flags* and the *protocol* are considered as relevant. To create a comparable synthetic setting, we generate a dataset using the function $y = (f_1 \vee \neg(f_2 \vee f_3 \vee f_4 \vee f_5 \vee f_6 \vee f_7))$. We then evaluate the XAI approaches on both datasets.

The results, marked as “synthetic” and “real” in Table 1, indicate a similar ranking of the XAI approaches with respect to their performance on both datasets. We observe that, while all perturbation-based approaches perform worse on real data, LIME achieves considerably better results on synthetic data in comparison to the real setting. This may be due to its local linear approximation that benefits more from the equal distribution of different sample permutations in the synthetic data. This experiment suggests that XAI performance on our synthetic datasets closely resembles real world application scenarios.

6 Discussion

Evaluation of eight post-hoc XAI approaches shows that many approaches fail to give satisfactory explanations even on basic categorical tabular data. The gradient-based approaches used in this work all show weaker performance than perturbation-based methods. While the approaches LIME and SHAP are both capable of well explaining basic linearly separable Boolean functions, only SHAP is capable of explaining non-linearly separable functions with up to 3 variables. Overall, we find that investigated approaches struggle to explain more complex, as well as non-linear Boolean functions. The datasets generated for these experiments may be used to gain first insights into XAI performance on categorical tabular data and will therefore be made available as benchmark datasets¹.

7 Conclusion

In this paper, we investigated XAI performance on categorical tabular data, proposing a setting in which XAI approaches can be evaluated independently

¹ <http://www.dmir.uni-wuerzburg.de/projects/deepsan/xai-eval-data/>.

of classifier performance using synthetic datasets with gold standard explanations. We generated benchmark datasets containing typical relationships between binary attributes such as AND, OR and XOR, as well as explanations according to a novel definition of relevance of features in Boolean functions.

Using these datasets, we empirically evaluated eight state-of-the-art XAI approaches. We found that many approaches fail to capture simple feature relationships such as non-linear XOR connections, with performance decreasing with increasing relationship complexity. Overall, we found the tested gradient-based approaches to yield worse results than the perturbation-based methods. By evaluating an expert-annotated dataset from the intrusion detection domain and comparing the results to explanations from synthetic data, we showed that the findings on our synthetic datasets can be applied to realistic data.

Acknowledgement. The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany as part of the DeepScan project (01IS18045A).

References

1. Ancona, M., Ceolini, E., Öztireli, C., Gross, M.: A unified view of gradient-based attribution methods for deep neural networks. In: NIPS 2017 - Workshop on Interpreting, Explaining and Visualizing Deep Learning. ETH Zurich (2017)
2. Ancona, M., Ceolini, E., Öztireli, C., Gross, M.: Gradient-based attribution methods. In: Samek, W., Montavon, G., Vedaldi, A., Hansen, L.K., Müller, K.-R. (eds.) Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. LNCS (LNAI), vol. 11700, pp. 169–191. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-28954-6_9
3. Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R., Samek, W.: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* **10**(7), e0130140 (2015)
4. Castro, J., Gómez, D., Tejada, J.: Polynomial calculation of the shapley value based on sampling. *Comput. Oper. Res.* **36**(5), 1726–1730 (2009)
5. Kindermans, P.J., et al.: Learning how to explain neural networks: Patternnet and patternattribution. In: International Conference on Learning Representations (2018)
6. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: Advances in Neural Information Processing Systems, pp. 4765–4774 (2017)
7. O’Donnell, R.: Analysis of Boolean Functions. Cambridge University Press, Cambridge (2014)
8. Preiss, B.: Data Structures and Algorithms with Object-Oriented Design Patterns in Java (1999)
9. Ribeiro, M.T., Singh, S., Guestrin, C.: Why should i trust you?: explaining the predictions of any classifier. In: 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data mining, pp. 1135–1144. ACM (2016)
10. Ring, M., Schlr, D., Landes, D., Hotho, A.: Flow-based network traffic generation using generative adversarial networks. *Comput. Secur.* **82**, 156–172 (2019)
11. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.* **1**(5), 206 (2019)

12. Shrikumar, A., Greenside, P., Kundaje, A.: Learning important features through propagating activation differences. In: 34th International Conference on Machine Learning, vol. 70, pp. 3145–3153. JMLR. org (2017)
13. Shrikumar, A., Greenside, P., Shcherbina, A., Kundaje, A.: Not just a black box: Learning important features through propagating activation differences. arXiv preprint [arXiv:1605.01713](https://arxiv.org/abs/1605.01713) (2016)
14. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: visualising image classification models and saliency maps. In: Bengio, Y., LeCun, Y. (eds.) ICLR (Workshop Poster) (2014)
15. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: 34th International Conference on Machine Learning, vol. 70, pp. 3319–3328. JMLR. org (2017)



SimLoss: Class Similarities in Cross Entropy

Konstantin Kobs^(✉), Michael Steininger^(✉), Albin Zehe^(✉),
Florian Lautenschlager^(✉), and Andreas Hotho^(✉)

Julius-Maximilians University Würzburg, Würzburg, Germany
{kobs,steininger,zehe,lautenschlager,hotho}@informatik.uni-wuerzburg.de

Abstract. One common loss function in neural network classification tasks is Categorical Cross Entropy (CCE), which punishes all misclassifications equally. However, classes often have an inherent structure. For instance, classifying an image of a rose as “violet” is better than as “truck”. We introduce SimLoss, a drop-in replacement for CCE that incorporates class similarities along with two techniques to construct such matrices from task-specific knowledge. We test SimLoss on Age Estimation and Image Classification and find that it brings significant improvements over CCE on several metrics. SimLoss therefore allows for explicit modeling of background knowledge by simply exchanging the loss function, while keeping the neural network architecture the same. Code and additional resources are available at <https://github.com/konstantinkobs/SimLoss>

Keywords: Cross entropy · Class similarity · Loss function.

*Roses are red, violets are blue,
both are somehow similar, but the classifier
has no clue.*

(Common proverb)

1 Introduction

One common loss function in neural network classifiers is Categorical Cross Entropy (CCE). CCE tries to maximize the assigned target class probability and punishes every misclassification in the same way, independent of other information about the predicted class. Often, however, classes have a special order or are similar to each other, such as different flowers in image classification. Including class similarities using the inherent class structure (e.g., class order), class properties (e.g., class names) or external information about the classes (e.g., knowledge graphs) in the training procedure would allow the classifier to make less severe mistakes as it learns to predict similar classes.

In this work, we modify Categorical Cross Entropy and propose Similarity Based Loss (SimLoss) as a way to explicitly introduce background knowledge

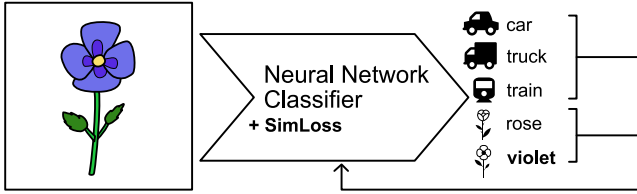


Fig. 1. SimLoss includes knowledge about class relations in the loss function.

into the training process, as visualized in Fig. 1. For this, we augment CCE with a matrix containing class similarities and propose two techniques in order to prepare such matrices that exploit certain class relations: class order and general class similarities. We show on two tasks, Age Estimation (exploiting class order) and Image Classification (exploiting semantic similarities using word embeddings), that SimLoss can significantly outperform CCE. We also show that tuning the hyper-parameters of both generation techniques influences the model’s performance on metrics measuring either more or less specific predictions.

Our contribution is twofold: First, we introduce a drop-in replacement for CCE that incorporates class similarities to support the training of neural network classifiers. Second, we describe two techniques to convert task-specific knowledge into matrices that can be used in the proposed loss function.

2 Related Work

Previous work on including task-specific knowledge in classification is mostly designed for specific use cases, requires modifications to the model architecture or training procedure, or implicitly learns the information while training. Sukhbaatar et al. *implicitly* learn a probability instead of a similarity matrix (we provide an analysis of the relationship in the online material) that indicates the chance of a falsely assigned class label in order to compensate for noise [13]. This, however, requires changes in the network architecture and a special training procedure. An analysis of the relation between probability and similarity based matrices is further analyzed in the online resources. Related to tasks with similar classes are tasks where classes have a taxonomic structure, which is called hierarchical classification. Specifically designed loss functions and/or model architectures use the fact that classes that belong to the same category are more similar than others [1, 14]. Izbicki et al. exploit the geospatial relation between areas on earth to automatically geotag input photos [5]. Their model learns to predict a mixture of densities that spread across multiple areas instead of specific classes/areas. A number of task-specific methods try to use the inherent class order of so-called ordinal classification tasks [3, 4]. For example, Niu et al. use multiple binary classifications each indicating whether the value is greater than the class value [11]. Model architectures incorporating semantic similarities using word embeddings were shown to usually predict more similar classes if

they fail compared to models without similarity information [2, 12]. In contrast to the related work, the use of SimLoss does not require special model architectures and works on any common neural network classifier. This makes it easy to explicitly support the training procedure with background knowledge.

3 Similarity Based Loss

Our proposed Similarity Based Loss (SimLoss) is based on the Categorical Cross Entropy (CCE). CCE assumes that only one class is correct and is defined as $L_{\text{CCE}} = -\frac{1}{N} \sum_{i=1}^N \log(\mathbf{p}_i[y_i])$, where N is the size of the dataset and $\mathbf{p}_i[y_i]$ is the probability vector output of the network at the target index y_i for the i th example. To model additional knowledge, SimLoss adds a matrix \mathbf{S} , which gives

$$L_{\text{SimLoss}} = -\frac{1}{N} \sum_{i=1}^N \log \left(\sum_{c=1}^C \mathbf{S}_{y_i, c} \cdot \mathbf{p}_i[c] \right), \quad (1)$$

where $\mathbf{S} \in [0, 1]^{C \times C}$ encodes class relations. $\mathbf{S}_{i, j}$ is the similarity between classes i and j . $\mathbf{S}_{i, j} = 1$ if and only if classes i and j are identical or interchangeable.

SimLoss is equal to CCE if $\mathbf{S} = I_c$ (identity matrix). Non zero values lead to smaller losses when the network gives a high score to classes similar to the correct one. For misclassifications, this leads the network to predict similar classes.

Matrix Generation. We now propose two techniques to generate the matrix \mathbf{S} , which explicitly captures background knowledge about class relations. Our techniques allow the modeling of class order and general class similarities.

Class Order: If classes have an inherent order, we can calculate class similarities based on the distance between the class indices. As classes lying next to each other are more similar, we construct the similarity matrix \mathbf{S} as follows: Assuming the same distance between neighboring classes, we define the reduction factor $r \in [0, 1)$ to be the rate at which the similarity will get smaller given the distance to the correct class. The similarity matrix is then

$$\mathbf{S}_{i, j} = r^{|i-j|} \quad \forall i, j \in \{1, \dots, C\}. \quad (2)$$

The smaller the reduction factor, the faster the entries converge to 0 with increasing distance to the target class. If the reduction factor is set to 0, the matrix becomes the identity, resulting in the CCE loss. The reduction factor is a hyperparameter of this technique, which can be tuned using a validation dataset to optimize the model for different metrics, as we show in Sect. 4. As SimLoss is equivalent to CCE when $r = 0$ (assuming $0^0 = 1$), an optimized r will always perform at least as good as CCE unless we overfit.

General Class Similarity: For some classification tasks, a similarity between classes, such as class names, is available or can be defined. Then, we can use an appropriate similarity measure $\text{sim} : C \times C \rightarrow [0, 1]$ that returns the similarity for two classes $i, j \in \{1, \dots, C\}$ and calculate all entries of the similarity

matrix \mathbf{S} . Such similarity measures can be manual, semi- or fully-automatic. Additionally, we define a lower bound $l \in [0, 1)$ as a hyper-parameter that controls the minimal class similarity that should have an impact on the network punishment. We cut all similarities below l and then scale them such that l becomes 0:

$$\mathbf{S}_{i,j} = \frac{\max(0, \text{sim}(i, j) - l)}{1 - l} \quad \forall i, j \in \{1, \dots, C\}. \quad (3)$$

Assuming only the diagonal of \mathbf{S} are ones, converging $l \rightarrow 1$ leads to the CCE loss, as only the ones in the diagonal are preserved by the lower bound cut-off.

4 Experiments

In the following, we compare SimLoss to CCE by applying them to the same neural network model with the same hyper-parameters for Age Estimation and Image Classification. *Age Estimation* is an ordinal classification task with the goal of predicting the age of a person given an image of their face. The classes have an inherent order: two classes are more similar if they represent similar ages. A misclassification is thus less harmful for nearer classes. In *Image Classification*, the goal is to recognize an object shown in an image. Here, we use class name word embeddings to model semantic class similarities. For example, classifying an image of a rose as “violet” is less harmful than classifying it as “truck”.

Datasets and Resources. For *Age Estimation*, we train neural networks on the UTKFace [15] and AFAD [11] datasets, both containing human face images annotated with their age. For UTKFace, we use all images for ages 1 to 90, while AFAD has 61 age classes. We randomly sample training/validation/test sets using 60/20/20 splits. For *Image Classification*, we use the CIFAR-100 dataset [7]. We also use word embeddings from a word2vec model pretrained on Google News [9] to calculate the semantic similarity between class names. Four class names do not yield a word embedding and are therefore eliminated. Each remaining class has 450 training, 50 validation, and 100 test examples.

Evaluation Metrics. To evaluate our method, we employ task-dependent evaluation metrics that focus both on correct predictions and the similarity of predicted and target class. For *Age Estimation*: Accuracy (Acc), Mean Absolute Error (MAE), and Mean Squared Error (MSE). Accuracy captures exact predictions, while MAE and MSE capture the distance to the target class, thus considering class order.: *Image Classification*: Accuracy, Superclass Accuracy (SA), and Failed Superclass Accuracy (FSA). Every example in the CIFAR-100 dataset has a main class and a superclass (e.g., classes “rose” and “orchid” have the superclass “flower”). Superclass Accuracy is the fraction of examples that are correctly put into the corresponding superclass. This value is always at least as high as Accuracy, as a correctly assigned class implies the correct superclass. Failed Superclass Accuracy only observes misclassified examples, thus measuring the similarity of misclassifications compared to the target class. A high FSA means that if the model predicts the wrong class, the predicted class is at least

similar to the correct class. Accuracy only counts exact predictions, while SA and FSA focus on the semantic similarity of the prediction to the target class.

Generating the Similarity Matrix. Since *Age Estimation* has equidistant classes, the similarity matrix can be built using Eq. (2) without any modifications. In *Image Classification*, we define the similarity matrix as the cosine similarity $sim_{cos} : w \rightarrow [-1, 1]$ between class name embeddings, where $sim_{cos}(w, w) = 1$. To ensure compatibility with the definition in Sect. 3, we set $sim(i, j) = \max(0, sim_{cos}(w_i, w_j))$ in Eq. (3).

Experimental Setup. Since SimLoss is a drop-in replacement for CCE, we investigate the effects of changing the loss function on our example tasks. Recall that we do not focus on task specific models, but rather on the evaluation of SimLoss as a general loss function which can be used on various tasks. Both classification tasks are typical examples for using CCE. For *Age Estimation*, we take the Convolutional Neural Network (CNN) from [11] and change the output size to be the dataset’s number of classes. The input images are resized to 60 px by 60 px and the values of all color channels are standardized. We use the softmax function and apply the SimLoss loss function using the similarity matrix introduced above. We study the effect of the reduction factor r by performing grid search for $r \in \{0.0, 0.1, \dots, 0.9\}$ on the validation set. Optimizing the network using Adam [6] with a learning rate of 0.001 and a batch size of 1024, we employ early stopping [10] with a patience of 10 epochs on the validation MAE. We smooth random differences (e.g., by weight initialization) by averaging over 10 runs. For *Image Classification*, the LeNet CNN [8] is used. Global standardization is applied to the color channels of the input images. We stop early if the Accuracy on the validation set plateaus for 20 epochs of the Adam optimizer with a learning rate of 0.001, and a batch size of 1024. We optimize the matrix generation technique’s lower bound $l \in \{0.0, 0.1, \dots, 0.8, 0.9, 0.99\}$ with grid search and average 10 runs per configuration. $l = 0.99$ makes the loss equivalent to CCE, cutting all similarities except the diagonal.

5 Results

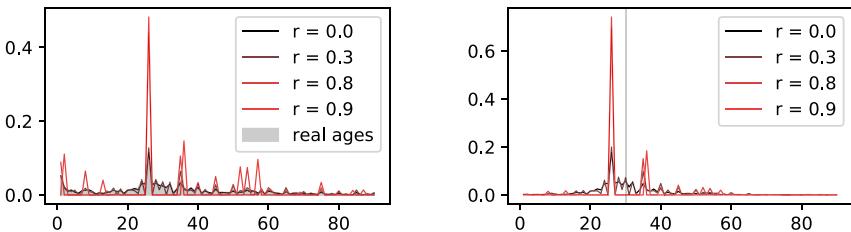
Table 1 shows the resulting mean metrics for the validation and test sets given a reduction factor r for both *Age Estimation* datasets. The best performing reduction factors on the validation and test set are always higher than 0.0, meaning that SimLoss outperforms CCE. Choosing the reduction factor then depends on the metric to optimize for. For UTKFace, a reduction factor of 0.3 leads to the best validation Accuracy, while 0.8 or 0.9 optimize MAE and MSE, respectively. For AFAD, $r = 0.5$ yields the best validation result on Accuracy, while $r = 0.7$ results in the best MAE and MSE. Overall, choosing a smaller reduction factor $r \approx 0.4$ optimizes the Accuracy, while larger $r \approx 0.8$ optimizes MAE and MSE. This is because large r lead to higher matrix values and thus smaller punishments for estimating a class near the correct age. A model optimized for that is favored by metrics that accept approximate matches, such as MAE or MSE.

Table 1. Validation and test results averaged over 10 runs on UTKFace and AFAD. Accuracy (Acc) is given in percent. Best validation values are written in bold. Statistically significantly different test values are marked by + or -, if they are on average better or worse than CCE (i.e. $r = 0.0$).

r	UTKFace						AFAD					
	Validation			Test			Validation			Test		
	Acc	MAE	MSE	Acc	MAE	MSE	Acc	MAE	MSE	Acc	MAE	MSE
0.0	15.23	7.09	122.12	14.47	7.39	131.65	11.17	4.05	32.61	11.22	4.10	33.64
0.1	15.43	7.06	119.87	14.48	7.29	127.18	11.21	4.06	32.75	11.30	4.10	33.73
0.2	15.94	7.06	121.28	14.57	7.27	127.13	11.40	4.09	33.52	11.37	4.15 ⁻	34.60 ⁻
0.3	16.25	6.95	117.67	15.17 ⁺	7.19 ⁺	125.70	11.34	4.10	33.53	11.38 ⁺	4.16 ⁻	34.53 ⁻
0.4	16.13	6.95	117.52	15.46 ⁺	7.18 ⁺	125.74	11.33	4.10	33.44	11.45 ⁺	4.16 ⁻	34.56 ⁻
0.5	16.10	6.89	115.59	15.09	7.18 ⁺	123.94	11.44	4.06	33.02	11.49 ⁺	4.13	34.21
0.6	15.62	6.83	112.85	14.34	7.09 ⁺	120.34 ⁺	11.26	4.01	31.99	11.31	4.05 ⁺	32.84 ⁺
0.7	14.39	6.79	110.12	13.07	7.08 ⁺	121.19 ⁺	11.22	3.95	31.17	11.11	4.02 ⁺	32.36 ⁺
0.8	13.50	6.74	108.80	12.57 ⁻	7.01 ⁺	117.99 ⁺	8.58	4.58	38.69	8.55 ⁻	4.64	39.78
0.9	9.69	6.90	106.23	9.16 ⁻	7.18 ⁺	117.62 ⁺	6.55	5.09	44.87	6.47 ⁻	5.15 ⁻	45.82 ⁻

A Wilcoxon-Signed-Rank-Test with a confidence interval of 5% shows that optimizing the reduction factor always leads to significant improvements over CCE. Sometimes, however, choosing the reduction factor based on a specific metric also results in a trade-off between the chosen and other metrics.

For the *Image Classification* task, Table 2 shows the results for the validation and test set of the CIFAR-100 dataset given a lower bound l . On average, the best performing model always has a lower bound of less than 0.99, again showing that SimLoss outperforms CCE. Also, a statistical test reveals that $l = 0.9$ gives significantly better results on the test set in terms of Accuracy and Superclass Accuracy. Smaller lower bounds tend to reduce the Accuracy as the loss function hardly punishes any misclassification. For $l \approx 1$, the loss is equivalent to CCE, forcing the network to predict the correct class, thus increasing Accuracy. In between, the network is guided to predict the correct class but is also not punished severely for misclassifications of similar classes. This improves Superclass Accuracy, which pays attention to more similar classes.



a) All examples. CCE fits the real data distribution the best. **b)** All examples of class “30”. The grey line indicates the target age.

Fig. 2. Mean probability distribution output for different r . High reduction factors lead the network to choose only few representative classes.

Table 2. Validation and test results over 10 runs with early stopping on the modified CIFAR-100 dataset. Best validation values are written in bold. Statistically significantly different test values are marked by + or -, if they are on average better or worse than CCE (i.e. $l = 0.99$).

l	Validation			Test		
	Accuracy	SA	FSA	Accuracy	SA	FSA
0.99	46.89 %	55.78 %	16.73 %	39.51 %	49.22 %	16.05 %
0.90	47.42 %	56.32 %	16.95 %	40.15 % ⁺	49.93 % ⁺	16.36 %
0.80	46.37 %	55.38 %	16.80 %	39.49 %	49.32 %	16.22 %
0.70	46.95 %	55.92 %	16.90 %	39.86 %	49.63 %	16.25 %
0.60	47.28 %	56.44 %	17.39 %	40.00 %	50.00 %	16.67 % ⁺
0.50	46.36 %	56.18 %	18.28 %	39.26 %	49.40 %	16.70 % ⁺
0.40	38.03 %	50.58 %	20.28 %	32.18 % ⁻	44.58 % ⁻	18.30 % ⁺
0.30	28.65 %	43.76 %	21.18 %	24.43 % ⁻	38.90 % ⁻	19.13 % ⁺
0.20	21.66 %	37.97 %	20.80 %	18.54 % ⁻	33.68 % ⁻	18.58 % ⁺
0.10	16.40 %	31.68 %	18.31 %	14.25 % ⁻	28.70 % ⁻	16.85 %
0.00	2.80 %	8.37 %	5.77 %	2.53 % ⁻	8.06 % ⁻	5.71 % ⁻

Analysis. To understand the effect of SimLoss, we focus on Age Estimation whose one dimensional classes are easy to visualize. We compare the best models for UTKFace trained using SimLoss and CCE, i.e. $r \in \{0.0, 0.3, 0.8, 0.9\}$. For each r , we plot the mean output distribution for all examples in the dataset as well as the real age distribution, which is shown in Fig. 2a. CCE ($r = 0.0$) resembles the real age distribution the best, while higher reduction factors tend to aggregate groups of multiple age classes. With a higher reduction factor, the number of spikes decreases and the distances between them increase: The model chooses representative classes to which it mainly distributes the output probability mass. This becomes apparent in Fig. 2b, where we plot the mean output distribution for all examples of age 30. The network with $r = 0.9$ focuses its probability output to the two nearest representative classes, in this case “26” and “35”. The Accuracy of the network decreases, as the output probability mass is not on the correct class, but the distance of the prediction to the correct class is smaller than for CCE. Representative classes are apparently chosen such that frequent items receive more probability mass from the model. A higher reduction factor therefore leads to a coarser class selection. This can be explained by the optimization objective of the loss function. The loss should be smaller for misclassifications of similar classes than for dissimilar classes. Representing multiple similar classes as one class and predicting it more often for similar classes does not lead to the smallest possible loss value. However, the loss gets smaller compared to predicting dissimilar classes, as the punishment should be smaller for classifying a similar class. In the case of Age Estimation, predicting an age that lies close to the correct age will decrease the Accuracy, but perform

better than CCE on MAE and MSE. In Image Classification, selecting one or multiple representative classes leads to smaller Accuracy but to higher Superclass Accuracy and Failed Superclass Accuracy than CCE. Higher similarities in the matrix thus guide the network to make coarser predictions, improving metrics that accept predictions of similar classes. The results from Sect. 4 also show that keeping the loss near CCE by choosing the similarity matrix conservatively can improve on specific prediction metrics such as Accuracy as well.

6 Conclusion

In this work, we have presented SimLoss, a modified Categorical Cross Entropy loss function that incorporates background knowledge about class relations in form of class similarities. We have introduced two techniques to prepare similarity matrices to exploit class order and general class similarity that can be used to significantly improve the performance of neural network classifiers on different metrics. Also, SimLoss helped with predicting more similar classes if the model misclassified an example. In our analysis, we found that SimLoss forced the model to focus on choosing representative classes. The number of representative classes can be implicitly tuned by a hyper-parameter. While finding the best hyper-parameter and similarity metric can be computationally expensive and non-trivial, SimLoss can incorporate arbitrary similarity metrics into a classifier.

References

1. Cesa-Bianchi, N., Gentile, C., Zaniboni, L.: Incremental algorithms for hierarchical classification. *J. Mach. Learn. Res.* **7**, 31–54 (2006)
2. Frome, A., Corrado, G.S., Shlens, J., Bengio, S., Dean, J., Mikolov, T., et al.: Devise: a deep visual-semantic embedding model. In: *NIPS* (2013)
3. Fu, Y., Huang, T.S.: Human age estimation with regression on discriminative aging manifold. *IEEE Trans. Multimed.* **10**(4), 578–584 (2008)
4. Guo, G., Mu, G., Fu, Y., Huang, T.S.: Human age estimation using bio-inspired features. In: *CVPR*. IEEE (2009)
5. Izbicki, M., Papalexakis, E.E., Tsotras, V.J.: Exploiting the earth’s spherical geometry to geolocate images. In: Brefeld, U., Fromont, E., Hotho, A., Knobbe, A., Maathuis, M., Robardet, C. (eds.) *ECML PKDD 2019*, vol. 11907, pp. 3–19. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-46147-8_1
6. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
7. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Technical report, Citeseer (2009)
8. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
9. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint [arXiv:1301.3781](https://arxiv.org/abs/1301.3781) (2013)
10. Morgan, N., Bourlard, H.: Generalization and parameter estimation in feedforward nets: some experiments. In: *NIPS* (1990)

11. Niu, Z., Zhou, M., Wang, L., Gao, X., Hua, G.: Ordinal regression with multiple output CNN for age estimation. In: CVPR (2016)
12. Norouzi, M., et al.: Zero-shot learning by convex combination of semantic embeddings. arXiv preprint [arXiv:1312.5650](https://arxiv.org/abs/1312.5650) (2013)
13. Sukhbaatar, S., Bruna, J., Paluri, M., Bourdev, L., Fergus, R.: Training convolutional networks with noisy labels. arXiv preprint [arXiv:1406.2080](https://arxiv.org/abs/1406.2080) (2014)
14. Wu, C., Tygert, M., LeCun, Y.: Hierarchical loss for classification. arXiv preprint [arXiv:1709.01062](https://arxiv.org/abs/1709.01062) (2017)
15. Zhang, Z., Song, Y., Qi, H.: Age progression/regression by conditional adversarial autoencoder. In: CVPR (2017)



Efficient and Precise Classification of CT Scannings of Renal Tumors Using Convolutional Neural Networks

Mikkel Pedersen¹, Henning Christiansen^{1(✉)}, and Nessn H. Azawi^{2,3,4}

¹ Roskilde University, Roskilde, Denmark
{mikpped,henning}@ruc.dk

² Zealand University Hospital, Roskilde, Denmark
nesa@regionsjaelland.dk

³ University of Copenhagen, Copenhagen, Denmark

⁴ Odense University Hospital, Odense, Denmark

Abstract. We propose a new schema for training and use of deep convolutional neural networks for classification of renal tumors as benign or malign from CT scanning images. A CT scanning of a part of the human body produces a stack of 2D images, each representing a slice at a certain depth, and thus comprising a 3D mapping. An additional temporal dimension may be added by injection of contrast fluid with CT scannings performed at certain time intervals. We reduce dimensionality – and thus computational complexity – by ignoring depth and temporal information, while maintaining an ultimate accuracy. Classification of a given scan is done by majority voting over the classifications of all its 2D images. Images are divided into training and validation sets on a patient basis in order to reduce overtraining. Current experiments with scans for 369 patients, yielding almost 20,000 2D images, demonstrate an accuracy of 93.3% for single images and 100% for patients.

1 Introduction

A renal tumor may be benign or malign, and in the latter case, immediate surgery is likely needed. Biopsy procedures are currently the only way for precise classification of renal tumors, but non-invasive methods are preferable. The decision of whether or not to operate a given patient is typically done by manual inspection of CT scanning images, perhaps complemented by other medical tests. However, this practice has a high error rate, and especially false positives are problematic as they result in unnecessary surgery, leading to lost life quality for the patients and waste of resources in the health sector. Recent studies indicate rates of 11–17% of surgery for patients with renal tumors, turning out to be benign [7, 10, 12]. A precise automatic classification from CT scan images may, thus, become an important decision support tool.

In the recent years, deep learning methods for convolutional neural networks (CNN) have lead to numerous reports of impressive results in medical image

analysis; CNN and deep learning are able to identify subtle but indicative features that are difficult to recognize for a human eye as, e.g., indicated above.

A CT scanning produces a 3D mapping of the tumor, consisting of a stack of 2D images, each focused at a specific depth. An additional temporal dimension may be added by injection of contrast fluid with scannings performed at certain time intervals, leading to so-called multiphase images. We present an experiment of training and testing a CNN on a substantial number of CT images collected from 369 Danish patients, yielding more than 20,000 2D images. Compared with other approaches, we reduce dimensionality – and thus computation time – by ignoring depth and temporal information while maintaining a very high accuracy.

For classification of single images, our accuracy amounts 93.3%, and by a majority voting over all 2D images for a given patient, we obtain 100% accuracy. The selection procedure for separation the total set of images is important for these results. We can show that the separation of images into training and validation sets should be done on the level of patients, rather than randomly over all 2D images as a whole, which results in a significant overtraining.

Section 2 gives an account on related work, focusing specifically on CNN in relation to CT scanning. In Sect. 3, we explain more about the datasets produced by CT scannings, as well as the dataset available for our experiments. Our specific CNN model is explained in Sect. 4 and test results presented in Sect. 5. Section 6 gives a summary and our directions for future work.

2 Related Work

Convolution Neural Networks (CNNs), introduced by LeCun et al in 1989 [8] (for a recent overview, see [4, chap. 9]), have developed into a standard for medical image analysis. The migration into standard equipment for medical usage is still in its infancy, but a lot of research effort is invested in the field all over the world. Even when restricted to CT scan images, the DBLP bibliography reports more than 300 scientific papers since 2015 on the topic.¹ CNNs are supported by several software platforms, including the TensorFlow [1] program library and its high level API Keras [3] that we have used for our experiments.

There are some recent works on classification of renal tumors. Pan et al [9] apply a complex network structure based on CNN, involving explicit phases of segmentation (“where is the tumor?”) and classification, where we rely on CNN’s ability automatically to identify the image features that are important for classification. They test out different CNN structures for scans of a total of 131 patients for training and test, each given by 300 2D images; accuracy up to 100% is reported for the best setup. The referenced paper emphasizes the importance of using multi-phasic CT images, but it is not clear whether or how this information is employed in their model and in training and test.

Han et al [5] analyze multiphase CT images with three phases, using manual segmentation and manual selection of one “best” 2D image from each phase;

¹ DBLP, The Computer Science Bibliography, <https://dblp.uni-trier.de>; searches for “convolution tomography” and “convolution CT”; March 2020.

each triplet of matching phase images is then combined into one in an ad-hoc manner. A total of 169 patients were used for training and test, and an accuracy of 85% is reported from the experiment.

In comparison, our model is trained and tested on a larger set of 369 patients, but compared with [9], a smaller number of 2D images (20–100, vs. their fixed 300) for each patient. Moreover, our approach has removed all time and phasic information related to the CT images, reducing the complexity of our network structure, shown in Fig. 1, and thus also training and classification time and hardware cost. The fact that we can do with perhaps as few as 20 images for classification may further speed up computation times. Unfortunately, none of the mentioned experiments (incl. our own) have reported figures for computation times, so we cannot make a precise comparison.

The mentioned approaches, including our own, involve known CNN structures adapted with new classification layers for the purpose. Pre-trained weight may be used unchanged (and only the classification layers are trained with CT scan images), as initial with weights for a retraining, or the network may be trained from scratch. We have used ResNet50V2 [6] as it performed best among those we tried out.

3 Data Set

Our datasets consist of CT scans for 369 patients (20% benign, 80% malign) in total performed at different Danish hospitals collected from 2015 until beginning of 2020; the possible influence of regional differences and changes in equipment and clinical procedures over the years are not considered in the present study.

True labelings as benign/malign were determined by subsequent surgery or other medical tests. No records are maintained of whether a scan involves different phases (nor how many). The number of 2D images per patient ranges from 20 to 100, yielding more than 20,000 2D-images. Original images with three colour channels of 512×512 pixel images are downscaled to 224×224 (for easy fit with the ResNet network structure).

We split data into 70% for training, 10% for validation (to check convergence during training) and 20% for independent test (of the trained model). To form a balanced training set (50% benign, 50% malign) we applied oversampling² to the original 20% benign ones.

Considering the set of 2D images from a single CT scan, and thus related to one patient, we note that 2D images at adjacent or close depths may be nearly identical, and the downscaling emphasizes this phenomenon. This increases the risk of over-training that might not be detected by a standard comparison of accuracy (or, alternatively, loss) of a trained model on its training data vs. the disjoint sets of validation and test data: if the splitting is done completely at random over all 2D images, each validation/test image will likely have several

² See [2] for an overview; overtraining is a potential disadvantage of oversampling, but shown by our tests, this is not the case in our experiments.

almost identical companions in the training set. To eliminate this phenomenon, we perform the random splitting at the patient level, rather than on individual images. Thus each such group of “similar” images belongs entirely to either the training set or to the test set. After that, each set is viewed as a collection of arbitrary images, each carrying a true classification as benign/malign but no information about which patient or possible phase. These decisions obviously throw away some information, dimensionality is reduced and so is the time complexity, and our test results, shown in Sect. 5, below, shows that we still obtain very good results.

4 Experiment and Model Structure

We use a modified version of the ResNet50V2 [6] CNN implemented with the TensorFlow and Keras software running on standard, affordable hardware (AMD Ryzen 2700x CPU with 16GB RAM, GeForce TTX 980ti 4GB and 1050ti 2GB GPUs).

Figure 1 shows the original ResNet50V2 structure together with our version. While ResNet50V2 classifies images into 1000 different categories according to the object depicted, we are interested in a binary classification into benign/malign, and thus the final, fully connected network layers can be much simpler; the convolutional layers are reused. We use transfer learning from ResNet50V2, applying its trained weights as initial one (rather than random weights), and the new fully connected layers for classification need to be trained from scratch.

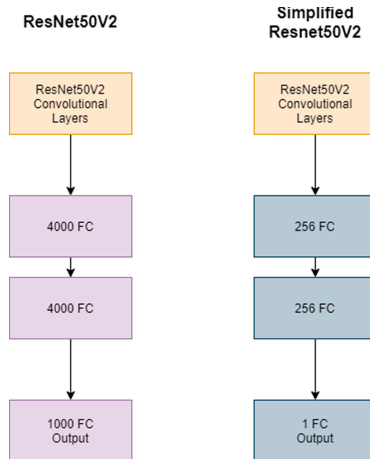


Fig. 1. ResNet50V2 and our simplified version. The convolutional layers are identical to ResNet50V2 [6]; the fully connected layers (FC) are reduced from 4000 nodes to 256 nodes and the final output layer from 1000 nodes to 1 node.

5 Results

Our adaptation of ResNet50V2 has been trained on the data explained above, with no overlap between images from single patients across the training, validation and test set. Here we report the results for (1a) the randomly per patient chosen 20%, and (1b) a small set of 12 patients collected in 2019 independently from the other data. To check the hypothesis that splitting per patient rather than randomly over all 2D images, we retrained the model with this selection methods and tested with (2a) 20% images selected in a similar manner, and (2b) the same 12 patients as in (1b). In addition, we performed in both cases, what we refer to as complete test, where the training and validation data are combined (1c); comparing these with (1a) and (2a) may also give indication of possible over-training.

Tests (1a) and (1b) (and (1c) as well) showed a 100% accuracy when each patient were classified as benign/malign by a majority voting of individual image classifications. In the following, we concentrate on the finer detail of the image classification, showing ROC curves and AUC values for the different tests. Figure 2 shows the results for (1a) and (1c), and Fig. 3 for (1b). Test (1a) indicates an impressive AUC of 0.973, and (1b) an AUC of 0.992, indicating a small and insignificant overtraining. Test (1b) indicates, as one would expect, a slightly lower, but still satisfactory, results on single images, but not enough to degrade the majority voting's 100% accuracy. However, this 12 patient set is too small for drawing any firm conclusions, and it motivates our planned future studies of influences of regional and historical variations.

To check our hypothesis presented in Sect. 3 that overtraining is reduced by our splitting principle, of separating the 2D image sets into training and validation parts randomly on a patient basis rather than on all images, we re-did training and tests using the latter principle for data set splitting, i.e., with a random

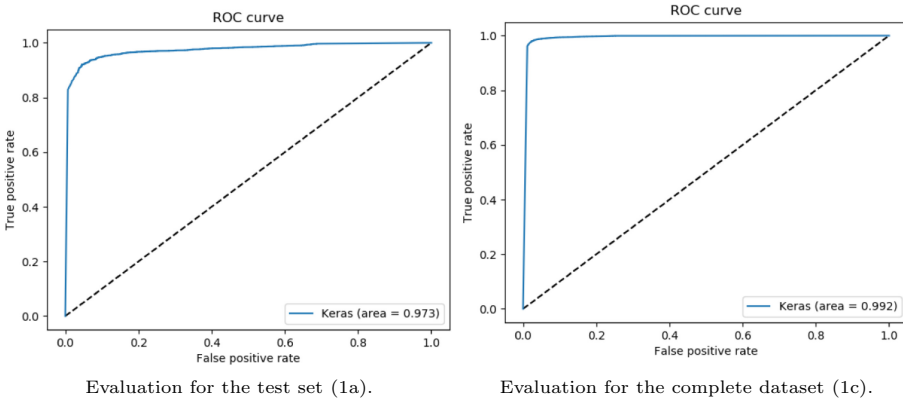


Fig. 2. ROC curve and AUC value for the (1a) test set and the complete dataset (1c). The accuracy amounts to 93.3% for (1a) and 97.7% for (1c).

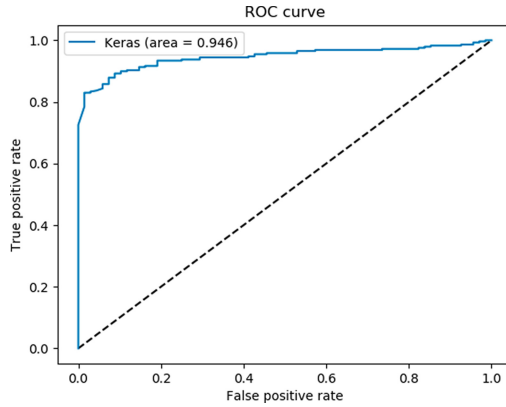


Fig. 3. ROC curve and AUC value for the (1b) test with an accuracy of 90.4%.

70/30 splitting at image level. ROC curves and AUC values for (2a–c) are shown in Fig. 4. As we expected, the performance on a test set – in which each image statistically will have a number of similar companions in the training set – is “too good”, and for unseen 12 patient data set, the performance degrades heavily. It appeared that the patient classification based on majority voting (not shown in the graphs) now classified 2 out of 12 patients incorrectly.

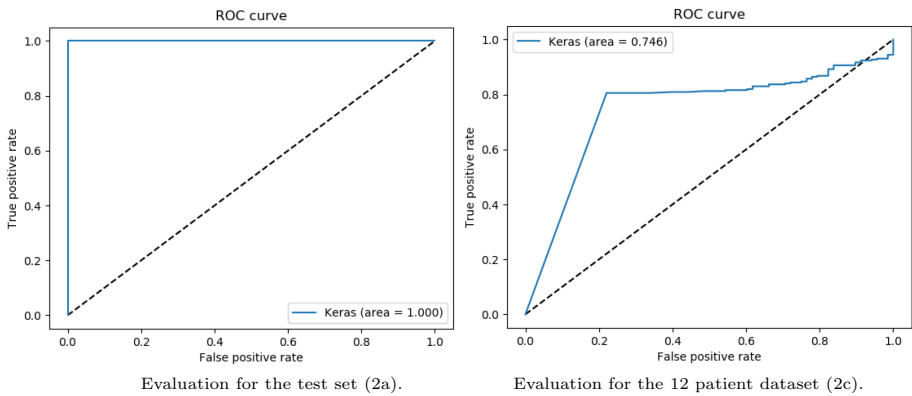


Fig. 4. Test (2a–2b) for the traditional splitting method that we do not recommend for these specific data. There is a severe disparity in indicated performance versus tests (1a–b).

6 Conclusion and Future Work

We have shown our first experiment of training and using a Convolutional Neural Network based on ResNet50V2 for automatic classification of renal tumors from

multi-phasic CT scan images. We used a substantially larger dataset than those related works we have compared with, cf. Sect. 2, and we tested on independent tests sets not included in the training and validation sets. In contrast to the compared works, we discard information about phases as well as depth information of each 2D image; as it appears in our tests, this does not affect the accuracy, so no essential information is lost. On the other hand, it reduces complexity of the neural network and thus time complexity for training and classification. We used oversampling to obtain balanced training sets as there were about four times as many malign patients as benign ones.

We presented a hypothesis that the overtraining can be reduced by separating the 2D image sets into training and validation parts randomly on a patient basis rather than on all images; this was clearly verified by our test results. Classification using the trained model yielded an accuracy per image ranging from 90.4% to 97.7% in the different tests, leading to a consistent 100% classification accuracy per patient using majority voting. All in all, we consider the method we have developed as a worthy candidate to be matured into an effective and efficient decision support tool for medical experts, giving an instant and reliable proposal for a diagnosis.

Our plans for future work include collecting a much larger dataset based on all available CT scan images from Danish hospitals for a critical assessment of the observed 100% classification, and we need also study the details of possible influences regional difference and historical developments in CT scanning equipments. We plan a detailed assessment and interpretation of the feature maps extracted from the image data through network training, aiming at a visualization, e.g., along the lines of Selvaraju et al [11]. From a clinical standpoint, interpretation an visualization of the feature maps may provide some clarity to areas of interest on CT imaging that are not currently understood when diagnosing renal cancer, and it may diminish the magic blackbox flavour of CNN-based classification and promote its acceptance as a practical decision support tool.

References

1. Abadi, M., et al.: Tensorflow: a system for large-scale machine learning. In: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2016), pp. 265–283 (2016), <https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf>
2. Buda, M., Maki, A., Mazurowski, M.A.: A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks* **106**, 249–259 (2018). <https://doi.org/10.1016/j.neunet.2018.07.011>
3. Chollet, F., et al.: Keras (2015). <https://keras.io>
4. Goodfellow, I., Bengio, Y., Courville, A.: *Deep Learning*. MIT Press (2016). <http://www.deeplearningbook.org>
5. Han, S., Hwang, S.I., Lee, H.J.: The classification of renal cancer in 3-Phase CT images using a deep learning method. *J. Digit. Imaging* **32**(4), 638–643 (2019). <https://doi.org/10.1007/s10278-019-00230-2>
6. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks (2016)

7. Johnson, D.C., et al.: Preoperatively misclassified, surgically removed benign renal masses: a systematic review of surgical series and United States population level burden estimate. *J. Urol.* **193**(1), 30–35 (2015). <https://doi.org/10.1016/j.juro.2014.07.102>
8. LeCun, Y., et al.: Backpropagation applied to handwritten zip code recognition. *Neural Comput.* **1**(4), 541–551 (1989). <https://doi.org/10.1162/neco.1989.1.4.541>
9. Pan, T., et al.: A multi-task convolutional neural network for renal tumor segmentation and classification using multi-phasic CT images. In: 2019 IEEE International Conference on Image Processing, ICIP 2019, Taipei, Taiwan, 22–25 September, 2019, pp. 809–813. IEEE (2019). <https://doi.org/10.1109/ICIP.2019.8802924>
10. Pedersen, C.L., Winck-Flyvholm, L., Dahl, C., Azawi, N.H.: High rate of benign-histology in radiologically suspect renal lesions. *Danish Med. J.* **61**(10) (2014)
11. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-CAM: visual explanations from deep networks via gradient-based localization. *Int. J. Comput. Vis.* **128**(2), 336–359 (2019). <https://doi.org/10.1007/s11263-019-01228-7>
12. Srougi, V., Kato, R.B., Salvatore, F.A., Ayres, P.P.M., Dall’Oglio, M.F., Srougi, M.: Incidence of benign lesions according to tumor size in solid renal masses. *Int. Braz. J. Urol.* **35**(4), 427–431 (2009). <https://doi.org/10.1590/S1677-55382009000400005>



Deep Autoencoder Ensembles for Anomaly Detection on Blockchain

Francesco Scicchitano, Angelica Liguori, Massimo Guarascio^(✉),
Ettore Ritacco, and Giuseppe Manco

ICAR-CNR, Via P. Bucci, 8/9c, Rende, Italy
{francesco.scicchitano,angelica.liguori,massimo.guarascio,
ettore.ritacco,giuseppe.manco}@icar.cnr.it

Abstract. Distributed Ledger technologies are becoming a standard for the management of online transactions, mainly due to their capability to ensure data privacy, trustworthiness and security. Still, they are not immune to security issues, as witnessed by recent successful cyber-attacks. Under a statistical perspective, attacks can be characterized as anomalous observations concerning the underlying activity. In this work, we propose an Ensemble Deep Learning approach to detect deviant behaviors on Blockchain where the base learner, an encoder-decoder model, is strengthened by iteratively learning and aggregating multiple instances, to compute an outlier score for each observation. Our experiments on historical logs of the Ethereum Classic network and synthetic data prove the capability of our model to effectively detect cyber-attacks.

Keywords: Blockchain · Anomaly detection · Sequence to sequence models · Encoder-decoder models · Ensemble learning

1 Introduction

There is a growing interest in adopting *Distributed Ledger Technology* (DLT) due to its capability to ensure privacy, trustworthiness and security in online transactions and payments. In particular, the Blockchain represents the most known and widespread DLT [5] and allows to save data under the form of permanent and verifiable transactions between two parties.

However, as discussed in [11], Blockchain is not immune to security issues, therefore the early detection of in-progress attacks represents a challenging and important problem. In particular, all processes within the blockchain are logged and, since the ledger is open, it is natural to ask whether these logs can be exploited to the early detection challenge. In this paper we focus on the information collected by the activities of *Ethereum Classic*¹ (ETC), a public blockchain. ETC blockchain has experienced two significant attacks: [1] reports an attack on

¹ <http://ethereumclassic.org/>.

18 June 2016 (referred as DAO from now on) and in January 2019 researchers confirmed a successful 51% attack². Our purpose is then to identify the core features from ETC logs that allow to early detect attacks.

The current literature focused on the machine learning techniques as a powerful tool to identify cyber-attacks and detect anomalous behaviors real-time or for post-incident analysis. Notably, both supervised and unsupervised machine learning algorithms have been successfully employed to support intrusion detection and prevention systems, as well as to detect system misuses and security breaches. However, these techniques were seldom applied to blockchain, with few exceptions where preliminary machine learning based approaches have been proposed to improve the security on blockchain [4]. A visual analytical approach of attack discovery is proposed in [2], where a set of statistics, collected from the Ethereum blockchain, is used as input to an unsupervised anomaly detection system: the work shows an anomalous peak close to the corresponding DAO attack date. This approach has been extended in [10], where authors propose an encoder-decoder deep learning model to detect ETC attacks. Notably, several anomaly detection techniques has been proposed in literature for different types of scenarios [3,9], but we mainly focused on the current approaches adopted to early detect anomalies on blockchain.

Specifically, in this paper we investigate the adoption of autoencoder ensembles to the analysis of ETC activities. Ensembles can strengthen the capabilities of the basic autoencoders which tend to overfit, especially in noisy contexts (like the ETC logs). Our strategy, based on sequence-to-sequence ensemble, consisting in by progressively training a weak learner with a Snapshot Procedure [8], i.e., a cyclic alteration of the learning rate. As shown in [8], this approach allows to guarantee a better exploration of the search space by a combination of multiple local minima, without affecting the training performance.

Our main contributions can be summarized as follow: (i) an unsupervised neural network architecture is devised for the early detection of anomalies. Basically, multiple sequence-to-sequence models are discovered by exploiting a Snapshot procedure and combined according to a suitable strategy, (ii) an ETC real dataset is used to assess the outlier detection capability of the proposed model. In addition, a robustness analysis is performed on synthesized data.

The paper is organized as follows: Sect. 2 is devoted to describe the ensemble architecture and the snapshot procedure; in Sect. 3, first, we illustrate the experimentation performed on a real (from ETC blockchain) and a synthetic dataset. Finally, in Sect. 4, we discuss relevant open issues and future works.

2 Methodology

In this section we describe the machine learning approach proposed to detect attacks in progress on blockchain-based systems and, more in general, in cybersecurity scenarios. Due to the lack of labeled data, an unsupervised method is

² See <http://tiny.cc/fri3iz>.

adopted to identify outliers: the underlying assumption is that successful attacks represents extremely rare events and, typically, do not share common patterns.

As a consequence, frequently supervised techniques fail to detect new incoming/in-progress attacks, resulting in poor performances.

Our proposal adapts the snapshot ensemble method defined in [8] to a sequential unsupervised scenario exploiting an encoder-decoder model as a base model. In detail, it aims at replicating an input sequence by producing a reconstructed copy from the compressed representation of the input, therefore the reproduction error can be used as anomaly indicator. Although the idea to use the reconstruction error as anomaly score to identify deviant behaviors is not new itself³, it has been not fully explored to monitor malicious behaviors on blockchain and new research lines can be analyzed. In [10], the authors proposed a preliminary approach able to recognize attacks on blockchain. However this method could be affected by the concept drift problem, while our approach, by considering different models, is able to handle smooth changes.

In our framework we instantiate several base models, which outlieriness score is finally averaged. These models are generated according to a simple algorithm, which takes advantage of the iterative gradient-descent optimization procedure of the neural network learning phase. The latter assumes that, at each step, the weights of the underlying network are updated in the direction opposite to the gradient, in order to progressively approach a local minimum. The learning rate determines the speed of convergence and it is progressively adapted to guarantee convergence. The local optimum obtained by the Stochastic Gradient Descent procedure and its variants depends both on the initialization and the choice of the learning rate. As noticed in [8], a cyclic reset of the latter has the same effect of re-initializing the network and restarting the optimization from another spot in the search space, allowing to identify different variants of the model.

Formally, we modeled data as temporally-sorted multi-dimensional events (i.e. each event is composed of several features). Let $X = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ be the sequence of events observed in a window with length N , and \mathbf{x}_t the feature vector of the t -th occurrence in the sequence X . An anomaly \mathbf{x}_t over X is an abnormal event significantly different from close events. The encoder-decoder is composed by two sub-networks. The encoder Θ compresses the input \mathbf{x} into a latent space $\Theta(\mathbf{x}) = \mathbf{z} \in \mathbb{R}^K$, generating an embedding of the original input into a latent vector of size K . By converse, the decoder Φ , given a K -dimensional vector \mathbf{z} , aims at generating an output $\Phi(\mathbf{z}) = \mathbf{y}$ as close as possible to the original input. Θ and Φ are modeled as Recurrent Neural Networks (RNNs) [6], which represent a natural choice to handle sequential data: by iterating over the sequence a recurrent network is able to store (partial) memory of each event. In our implementation we use Long Short-Term Memories (LSTM) [7]. Thus, given an input sequence $\mathcal{I} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, a single encoder-decoder learner computes

³ <https://bit.ly/32QRz00>.

an output sequence $\mathcal{O} = \{\mathbf{y}_1, \dots, \mathbf{y}_n\}$ as follows:

$$\begin{aligned}
 \mathbf{h}_t^{(e)} &= \text{RNN}_\theta(\mathbf{x}_t, \mathbf{h}_{t-1}^{(e)}) \\
 \mathbf{z} &= \text{mlp}_\vartheta(\mathbf{h}_t^{(e)}) \\
 \mathbf{h}_t^{(d)} &= \text{RNN}_\phi(\mathbf{z}, \mathbf{h}_{t-1}^{(d)}) \\
 \mathbf{y}_t &= \text{mlp}_\varphi(\mathbf{h}_t^{(d)})
 \end{aligned}
 \tag{1}$$

where RNN_θ and mlp_ϑ represent the encoder, with internal state $\mathbf{h}_t^{(e)}$ given the t -th event; symmetrically, RNN_ϕ and mlp_φ represent the decoder, with inner state $\mathbf{h}_t^{(d)}$. Further, mlp_ϑ and mlp_φ represent multilayer networks parameterized by θ and ϕ , respectively. Since the main purpose of the autoencoder is to reconstruct the input from a compact representation, the model can be trained by considering a reconstruction loss:

$$\ell(\mathcal{I}, \mathcal{O}) = \frac{1}{n} \sum_{t=1}^n \|\mathbf{x}_t - \mathbf{y}_t\|_2
 \tag{2}$$

Input subsequences are obtained from X through a sliding window mechanism. Each timestep within X is associated with a subsequence $W_t = \{\mathbf{x}_{t-m+1}, \dots, \dots, \mathbf{x}_t\}$, where m is the window size. As shown in [10], the autoencoder can be trained on a set $\{W_m, \dots, W_N\}$ of subsequences that can be obtained from X , by learning to reconstruct them in a way that minimizes the specified loss. The distance between the input and the output is used to measure the outlierness of the analyzed sequence. The final score is computed as the average of the outlierness scores of all the involved windows.

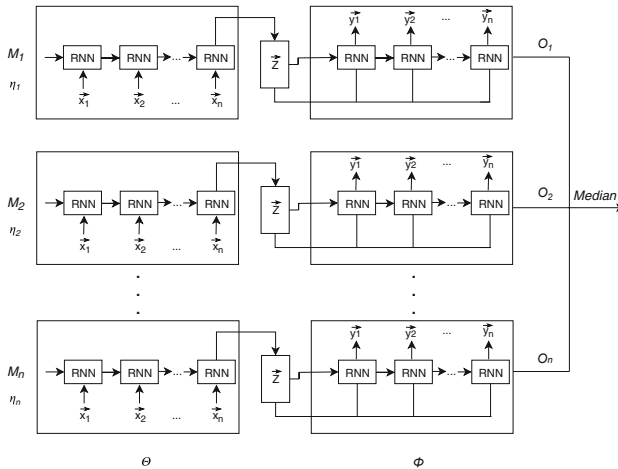


Fig. 1. Snapshot ensemble encoder-decoder model.

The Snapshot Ensemble Encoder-Decoder (SEED) is trained as follows. An autoencoder is randomly initialized and devised as M_1 . Then, the procedure iteratively learns model M_i by re-training M_{i-1} with the initial learning rate η for a fixed number of epochs. At each epoch η is progressively lowered. M_i is then collected in the ensemble and the learning rate is reinitialized. The final architecture is shown in Fig. 1, where the different encoder-decoder weak learners $\{M_1, M_2, \dots, M_n\}$ are shown. In the prediction stage, each sequence is passed as input to all instances and the reconstructed sequences are obtained. The final outlieriness score is the median of the n reconstruction errors $\{O_1, \dots, O_n\}$ produced by the instances.

3 Experimental Evaluation

In this section, first, we specify the values of the main parameters characterizing the DNN architecture, then the experimentation on the real and the synthetic data are shown. In detail, two LSTM layers (each one composed of 32 cells) are employed for encoding and decoding the input data and a *Hyperbolic Tangent* is adopted as activation function in each cell of these layers. The reconstruction error is measured in terms of *Mean Squared Error* (MSE) which is the loss function minimized by the optimizer (in our case we used the Adaptive Moment Estimation (Adam) algorithm). Finally, we set 200 epochs to learn the base models composing the ensemble.

3.1 Analysis of the Ethereum Classic Network

In this section, we apply SEED to the ETC network to identify attacks. As discussed in Sect. 1, ETC has experienced two known attacks: DAO (18 June 2016) and 51% attack. Documentation concerning the latter, is scarce but the reports state that it occurred in a period within the interval 5–8 January 2019. Our goal is thus to adopt SEED to highlight possible anomalies in the given time intervals. Our experiments⁴ have been performed on a four year sample of ETC blockchain by using preprocessing steps and data split (Fig. 2) like in [10]. At the end of the preprocessing phase, we obtained the following subset of relevant features, computed on a daily basis: (i) `block_size_average`, the average size (in bytes) of a block; (ii) `provided_gas_average`, referring to the average *provided gas* needed to perform the transaction; (iii) `block_difficulty_average`, the average effort necessary to validate a block; (iv) `transaction_average_per_block`, the average number of transactions contained in a block; (v) `gas_used_sum`, the total amount of employed gas; (vi) `transactions_number`, the total number of transactions in all blocks.

⁴ https://github.com/francescovicchitano/Anomaly_Detection_On_Blockchain.

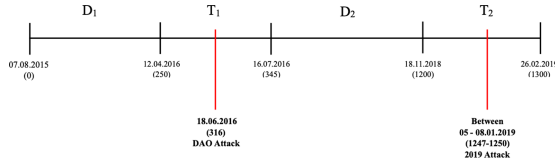


Fig. 2. Data split according to the attacks, as indicated in [10].

We consider two different training sets, namely D_1 (relating to the period prior DAO) and D_2 (covering the period post DAO and prior the 51% attack). T_1 and T_2 (the periods within a range of about two months from the attacks, exact dates are listed in Fig. 2) are used as test sets, for evaluating the outlieriness scores. We performed three different tests by training two instances of SEED model. The first two use D_1 as training set and scored all the events within T_1 and T_2 , respectively. In the third experiment SEED is trained on $D_1 \cup D_2$ and scores events in T_2 . We used 10 weak learners, obtaining scores in the Fig. 3.

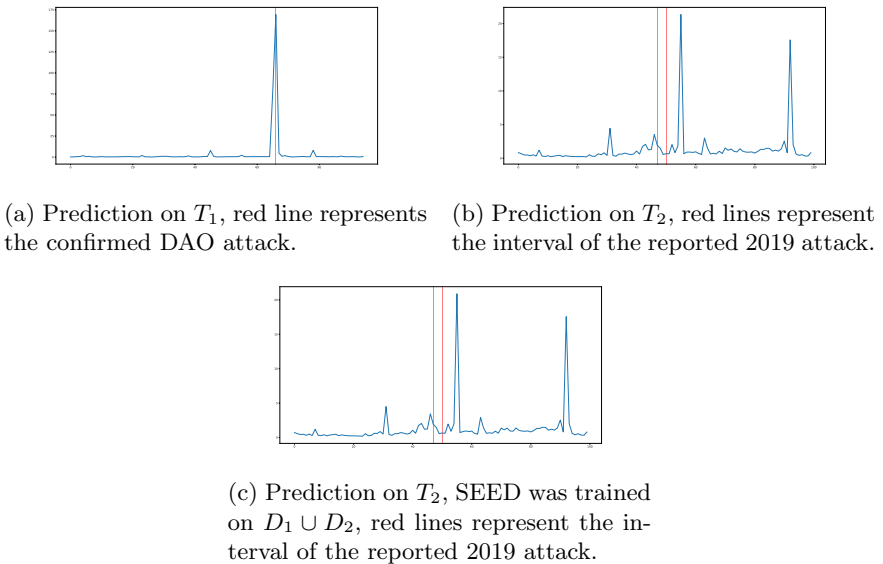


Fig. 3. Outlieriness score on ETC. (Color figure online)

Figures 3a and 3b plot the outlieriness scores computed by our model on both T_1 and T_2 . We can see that the DAO is perfectly detected, as shown in Fig. 3a. In fact, outlieriness score exhibits a peak on day 316, corresponding to DAO. The result is consistent with the findings of [2] but in our case the model is capable of perfectly detect the exact day of the attack.

In Figs. 3b and 3c we can see that the peak is translated of a few days respect to vertical red lines (the presumed starting and ending days of the 2019 attack): in fact, outlierness score changes its pattern as the attack period approaches; later, many companies frozen all activities on ETC network and blockchain didn't register a core amount of transactions, which were, instead, restarted in the forthcoming days^{5,6,7}, triggering the registered peak in day 55. The 51% attack highlighted by SEED was also confirmed in [10].

However, the selected features seem not fully sufficient to detect some types of attacks occurring on blockchain but we figure out that integrating data from other sources (as show in [2]) could improve the detection capability of SEED.

3.2 Experiments on Synthetic Data

In this section we perform a sensitivity analysis of SEED in a controlled scenario where were both the dimensionality of the data and the number of outliers is tuned. Data were generated according to the following procedure: (1) first a sequence D (as a matrix $1440 \times nFeat$) is generated; then (2) n points are randomly selected and a candidate feature is extracted for each point; finally

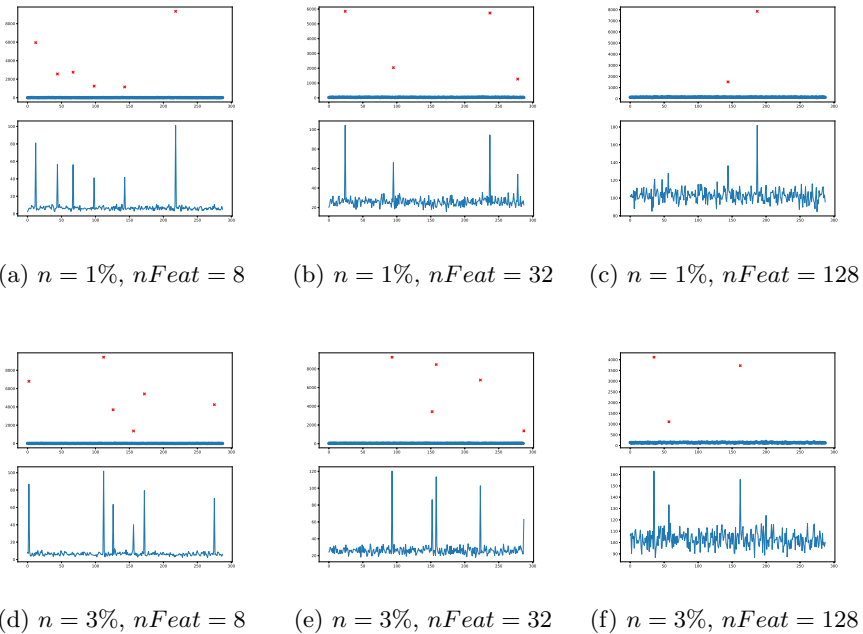


Fig. 4. Outlierness score on synthetic data. (Color figure online)

⁵ <http://shorturl.at/giz14>.

⁶ <https://bit.ly/30Qs9P8>.

⁷ <http://cryptonomist.ch/2019/01/07/ethereum-classic-attacco-del-51/>.

(3) an uniform noise is injected in the candidate feature. In our experiments, n ranges into $\{1\%, 3\%\}$ of the entire dataset and $nFeat \in \{8, 32, 128\}$, while we set the number of weak learners to 10, $|D| = 1440$ as in ETC and $\gamma = 200$. The evaluation is performed by adopting an hold-out validation protocol (80%–20%). The results are highlighted in Fig. 4. For each experiment we show two graphs: respectively, the topmost one represents, for each time-sorted event in the test set, the anomaly score assigned by the model; blue dots are regular events, while red crossed dots are the detected anomalies; while, the bottom graph, shows the overlap of all features, thus highlighting the peaks where we injected noise. These results highlight the capability of SEED to detect outliers in different settings.

4 Concluding Remarks

In this paper we proposed an unsupervised ensemble deep architecture for anomaly detection. The ensemble schema was obtained by exploiting the Snapshot Procedure, defined in [8] as a strategy to improve classifiers based on neural networks. In particular, we defined a snapshot ensemble encoder-decoder model to identify anomalies in sequential data, without providing to it any prior knowledge about outliers. An extensive evaluation on real and synthetic data proves the capability of the approach in detecting incoming attacks.

As future work, we plan to study new effective methods to select the weak learners composing the ensemble, so to improve the overall performances of the approach.

Acknowledgment. This work has been partially supported by MIUR - PON Research and Innovation 2014–2020 under project Secure Open Nets.

References

1. Atzei, N., Bartoletti, M., Cimoli, T.: A survey of attacks on Ethereum smart contracts (SoK). In: Maffei, M., Ryan, M. (eds.) POST 2017. LNCS, vol. 10204, pp. 164–186. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54455-6_8
2. Bogner, A.: Seeing is understanding: anomaly detection in blockchains with visualized features. In: UbiComp 2017, pp. 5–8. ACM (2017)
3. Corizzo, R., Ceci, M., Japkowicz, N.: Anomaly detection and repair for accurate predictions in geo-distributed big data. *Big Data Res.* **16**, 18–35 (2019)
4. Dey, S.: Securing majority-attack in blockchain using machine learning and algorithmic game theory: a proof of work. CoRR, abs/1806.05477 (2018)
5. El Ioini, N., Pahl, C.: A review of distributed ledger technologies. In: Panetto, H., Debruyne, C., Proper, H.A., Ardagna, C.A., Roman, D., Meersman, R. (eds.) OTM 2018. LNCS, vol. 11230, pp. 277–288. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-02671-4_16
6. Graves, A.: Supervised Sequence Labelling with Recurrent Neural Networks. *SCI*, vol. 385. Springer, Heidelberg (2012). <https://doi.org/10.1007/978-3-642-24797-2>
7. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)

8. Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J.E., Weinberger, K.Q.: Snapshot ensembles: train 1, get M for free. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017, Conference Track Proceedings (2017)
9. Munir, M., Siddiqui, M.S.A., Dengel, A., Ahmed, S.: DeepAnT: a deep learning approach for unsupervised anomaly detection in time series. *IEEE Access* **7**, 1991–2005 (2018)
10. Scicchitano, F., Liguori, A., Guarascio, M., Ritacco, E., Manco, G.: A deep learning approach for detecting security attacks on blockchain. In: Italian Conference on Cybersecurity, ITASEC (2020). CEUR-WS, <http://ceur-ws.org/Vol-2597/paper-19.pdf>
11. Ye, C., Li, G., Cai, H., Gu, Y., Fukuda, A.: Analysis of security in blockchain: case study in 51%-attack detecting. In: 2018 5th International Conference on Dependable Systems and Their Applications (DSA), pp. 15–24 (2018)



A Parallelized Variant of Junker's QUICKXPLAIN Algorithm

Cristian Vidal Silva¹(✉), Alexander Felfernig³, Jose Galindo², Müslüm Atas³,
and David Benavides²

¹ Catholic University of the North, Antofagasta, Chile
`cristian.vidal@ucn.cl`

² University of Sevilla, Seville, Spain
`{jagalindo,benavides}@us.es`

³ Graz University of Technology, Graz, Austria
`{afelfernig,muatas}@ist.tugraz.at`

Abstract. Conflict detection is used in many scenarios ranging from interactive decision making to the diagnosis of potentially faulty hardware components or models. In these scenarios, the efficient identification of conflicts is crucial. Junker's QUICKXPLAIN is a divide-and-conquer based algorithm for the determination of preferred minimal conflicts. Motivated by the increasing size and complexity of knowledge bases, we propose a parallelization of the original algorithm that helps to significantly improve runtime performance especially in complex knowledge bases. In this paper, we introduce a parallelized version of QUICKXPLAIN that is based on the idea of predicting and executing parallel consistency checks needed by QUICKXPLAIN.

1 Introduction

Conflict detection is used in many applications of constraint-based representations (and beyond). Examples thereof are *knowledge-based configuration* [13] where users define requirements and conflict detection is in charge of figuring out minimal sets of potential changes to the given requirements in order to restore consistency (if the configurator is not able to identify a solution), *recommender systems* [3, 10], and many other applications of model-based diagnosis [9]. Especially in interactive settings, there is often a need of identifying preferred conflicts [7, 11], for example, users of a car configurator or a camera recommender who have strict preferences regarding the upper price limit, are more interested in relaxations related to technical features (e.g., related to the availability of a skibag in a car or a wide-aperture lens in a pocket camera).

Conflict detection helps to find combinations of constraints in the knowledge base that are responsible for an inconsistency. QUICKXPLAIN is such a conflict detection algorithm which is frequently used and works for constraint-based representations, description logics, and SAT solvers [7]. The algorithm is based on a divide-and-conquer approach where consistency analysis operations are based

on the division of a constraint set $C = \{c_1..c_m\}$ into two subsets $C_a = \{c_1..c_k\}$ and $C_b = \{c_{k+1}..c_m\}$ assuming, for example, $k = \lfloor \frac{m}{2} \rfloor$. If C_b is *inconsistent*, the consideration set C can be reduced by half since C_a must not be analyzed anymore (at least one conflict exists in C_b). Depending on the QUICKXPLAIN variant, either C_a or C_b is checked for consistency.

Conflict detection is typically applied in combination with conflict resolution which helps to resolve all existing conflicts. In this context, the minimality (irreducibility) of conflict sets is important since this allows to resolve each conflict by simply deleting one of the elements in the conflict set. The elements to be deleted to restore global consistency are denoted as *hitting set* and can, for example, be determined on the basis of a hitting set directed acyclic graph [9]. Due to the increasing size and complexity of knowledge bases, there is an increasing need to further improve the performance of solution search and conflict detection/hitting set calculation [2, 4, 5, 8]. Parallelizations of algorithms in these scenarios have been implemented in different contexts. Approaches to parallelization have, for example, been proposed on the *reasoning level* [2] where the determination of a solution is based on the idea of identifying subproblems which can be solved to some degree independently by the available cores. Due to today's multi-core CPU architectures, such parallelization techniques become increasingly popular in order to be able to better exploit the offered computing resources.

A similar motivation led to the development of parallelization techniques in model-based diagnosis [9]. J. Marques-Silva et al. [6] propose a parallelization approach for hitting set determination where Reiter's approach to model-based diagnosis is parallelized by a level-wise expansion of a breadth-first search tree with the goal of computing minimal (cardinality) diagnoses. On each level, (minimal) conflict sets are determined in parallel, however, the determination of individual conflict sets is still a sequential process (based on QUICKXPLAIN [7]). In diagnosis search, the efficient determination of minimal conflicts is a core requirement [6]. Especially in constraint-based reasoning scenarios, the identification of minimal conflict sets is frequently based on QUICKXPLAIN [7]. Compared to iterative approaches of removing elements from inconsistent constraint sets [1], QUICKXPLAIN follows a divide-and-conquer strategy that helps to reduce the number of needed consistency checks. Although the algorithm is often used in interactive settings with challenging runtime requirements, up-to-now no parallelized version has been proposed. In this paper, we propose an algorithm that enables efficient parallelized minimal conflict detection and thus helps a.o. to significantly improve the runtime performance of conflict detection in interactive applications.

The contributions of this paper are the following. *First*, we show how to parallelize conflict detection with look-ahead strategies that scale with the number of available computing cores. *Second*, we show how to integrate our approach with the QUICKXPLAIN algorithm that is often used in constraint-based applications. *Third*, we show the applicability and improvements of our approach on the basis of performance evaluations. Finally, we point out in which way the proposed approach can help to improve the performance of existing diagnosis approaches. The remainder of the paper is organized as follows. In Sect. 2, we

introduce the basic idea of Junker’s QUICKXPLAIN using a working example. Thereafter, in Sect. 3 we introduce a parallelized variant of the algorithm. In Sect. 4, we analyze the proposed approach and report the results of a performance evaluation which shows significant improvements compared to standard QUICKXPLAIN. The paper is concluded with Sect. 5.

2 Calculating Minimal Conflicts

In the remainder of this paper, we introduce our approach to parallelized conflict detection on the basis of constraint-based knowledge representations [14]. A *conflict set* can be defined as a minimal set of constraints that is responsible for an inconsistency, i.e., a situation in which no solution can be found for a given constraint satisfaction problem (CSP) (see Definitions 1–2).

Definition 1. A Constraint Satisfaction Problem (CSP) is a triple (V, D, C) with a set of variables $V = \{v_1..v_n\}$, a set of domain definitions $D = \{dom(v_1)..dom(v_n)\}$, and a set of constraints $C = \{c_1..c_m\}$.

Definition 2. Assuming the inconsistency of C , a conflict set can be defined as a subset $CS \subseteq C : CS$ is inconsistent. CS minimal if $\neg \exists CS' : CS' \subset CS$.

Examples of a CSP and a conflict set are the following (see Examples 1–2).

Example 1. An example of a CSP for car configuration is the following: $V = \{cartype, fuel, pdc, color, skibag\}$, $D = \{dom(cartype) = [s, c], dom(fuel) = [p, d], dom(pdc) = [y, n], dom(color) = [b, r, g], dom(skibag) = [y, n]\}$, and $C = \{c_1 : fuel = d, c_2 : skibag = y, c_3 : color = b, c_4 : cartype = c, c_5 : pdc = y, c_6 : skibag = y \rightarrow cartype = s, c_7 : cartype = c \rightarrow color = b\}$.

Note that PCD refers to the Park Distance Control implemented in recent vehicles. It is convenient to distinguish between a *consistent background knowledge* B of constraints that cannot be relaxed (in our case, $B = \{c_6, c_7\}$) and a *consideration set* C of relaxable constraints (in our case, requirements $C = \{c_1..c_5\}$).

Example 2. In Example 1, there is one minimal conflict which is $CS = \{c_2, c_4\}$, since $\{c_2, c_4\} \subseteq C$ and inconsistent (CS). Furthermore, CS is minimal since there does not exist a CS' s.t. $CS' \subset CS$. The execution trace of QUICKXPLAIN for this working example is depicted in Fig. 1.

QUICKXPLAIN [7] (a variant is shown in Algorithms 1 and 2) supports the determination of minimal (irreducible) conflicts in a given set of constraints (C). QUICKXPLAIN is activated if the background knowledge B (often assumed to be empty or a consistent set of constraints) is *inconsistent* with the set of constraints C (we assume this consistency check to be performed by a direct solver call). The core algorithm is implemented in the function QX (Algorithm 2) that determines a minimal conflict which is a minimal subset of the constraints in C with the conflict set property.

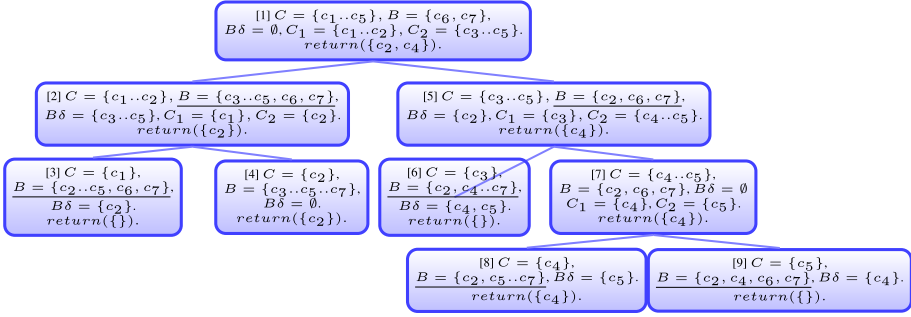


Fig. 1. QX execution trace for $C = \{c_1..c_5\}$ and $B = \{c_6, c_7\}$ assuming a minimal conflict set $CS = \{c_2, c_4\}$. Underlined B s denote QX consistency checks. For example, in the incarnation [2] of the QX function, the consistency check activated is $\{c_3..c_5, c_6, c_7\}$.

The function QX (Algorithm 2) focuses on isolating those constraints that are part of a minimal conflict. If C includes only one element ($C = \{c_\alpha\}$), this element can be considered as element of the conflict - this is due to the invariant property $inconsistent(C \cup B)$. If the B is consistent and C has more than one element, C is divided into two separate sets, where (in our QX variant) the second part (C_b) is added to B in order to analyse further elements of the conflict. The function QX activates a consistency check (INCONSISTENT) to figure out whether the considered background knowledge is inconsistent, i.e., no solution exists. $B\delta$ indicates constraints added to B .

Algorithm 1. QUICKXPLAIN(C, B) : CS

```

1: if CONSISTENT( $B \cup C$ ) then
2:   return('no conflict')
3: else if  $C = \emptyset$  then
4:   return( $\emptyset$ )
5: else
6:   return(QX( $C, B, \emptyset$ ))
7: end if

```

Algorithm 2. QX($C = \{c_1..c_m\}, B, B\delta$) : CS

```

1: if  $B\delta \neq \emptyset$  and INCONSISTENT( $B$ ) then
2:   return( $\emptyset$ )
3: end if
4: if  $C = \{c_\alpha\}$  then
5:   return( $\{c_\alpha\}$ )
6: end if
7:  $k = \lfloor \frac{m}{2} \rfloor$ 
8:  $C_a \leftarrow c_1..c_k; C_b \leftarrow c_{k+1}..c_m$ ;
9:  $\Delta_2 \leftarrow$  QX( $C_a, B \cup C_b, C_b$ );  $\Delta_1 \leftarrow$  QX( $C_b, B \cup \Delta_2, \Delta_2$ );
10: return( $\Delta_1 \cup \Delta_2$ )

```

In many of the mentioned application scenarios, there exists an exponential number of conflicts and ways to resolve a conflict [7]. Especially in interactive scenarios, it is extremely important to identify *preferred conflicts*, i.e., conflicts with a high probability of being the basis of a relaxation acceptable for the user. For example, if a user is strongly interested in low-priced digital cameras, a conflict set that includes a price limit might be of low relevance for the user (since the user is not willing to change the price limit). QUICKXPLAIN [7] supports the determination of *preferred conflicts*. Although our discussions focus on constraint-based representations, the approach can be applied to any kind of satisfiability problem such as propositional satisfiability (SAT) and description logics (DL). We assume *monotonic satisfiability* (see Proposition 1).

Proposition 1. *If a solution for a CSP satisfies all constraints $c_i \in C$ then it also satisfies every proper subset $C' \subset C$.*

QUICKXPLAIN determines one *minimal preferred conflict* at a time which includes constraints one might be willing to relax. In the line with [7], an explanation X is preferred over an explanation Y under the following condition (see Definition 3).

Definition 3. *We define a total order $<$ on the constraints in $C = \{c_1..c_m\}$ which is represented as $[c_1 < c_2 < .. < c_m]$. If $c_i < c_j$, i.e., the importance of c_i is higher than c_j , then $i < j$. If X and Y are two lexicographical constraint orderings of $c_1..c_n$, Y is preferred over X ($Y >_{lex} X$) iff $\exists k : c_k \in X - Y$ and $X \cap \{c_{k-1}..c_1\} = Y \cap \{c_{k-1}..c_1\}$.*

Example 3. Given the constraint ordering $[c_3 < c_4 < c_5 < c_6]$ and two binary conflict sets $X = \{c_3, c_4\}$ and $Y = \{c_4, c_5\}$, Y is preferred over X since $c_3 \in X - Y$ with $X \cap \{c_2, c_1\} = Y \cap \{c_2, c_1\}$.

3 Parallelizing QUICKXPLAIN

Our approach to parallelize the consistency checks in QX substitutes the *direct* solver call INCONSISTENT(B) in QX with the activation of a lookahead function (QXGEN) in which consistency checks are not only triggered to directly provide feedback to QX requests, but also to be able to provide fast answers for consistency checks potentially relevant in upcoming states of a QX instance. In the parallelized variant of QUICKXPLAIN, consistency checking is activated by QX with INCONSISTENT($C, B, B\delta$) (see Algorithm 3). This also activates the QXGEN function (see Algorithm 4) that starts to generate and trigger (in a parallelized fashion) further consistency checks that might be of relevance in upcoming QX phases. For the description of QXGEN, we employ a two-level *ordered set* notation which requires, for example, to embed the QX B into $\{B\}$, etc. In QXGEN, C , $B\delta$, and B are interpreted as *ordered sets*.

QXGEN-generated consistency checking tasks are stored in a LOOKUP table (see, e.g., Table 1). Thus, in the parallelized variant, QX has to activate the consistency check with INCONSISTENT($C, B, B\delta$). In contrast to the original QUICKXPLAIN approach, C and $B\delta$ are needed as additional parameters to conduct

Algorithm 3. $\text{INCONSISTENT}(C, B, B\delta): \text{Boolean}$

```

1: if  $\neg \text{EXISTSCONSISTENCYCHECK}(B)$  then
2:    $\text{QXGEN}(\{C\}, \{B\delta\}, \{B - B\delta\}, \{B\delta\}, 0)$ 
3: end if
4: return( $\neg \text{LOOKUP}(B)$ )

```

inferences about needed future consistency checks. While in the standard QX version $B\delta \subseteq B$, we assume $B\delta$ and B to be separate units in QXGEN.

Table 1. LOOKUP table indicating the consistency of individual constraint sets. The consistency checking tasks have been generated by ADDCC in the QXGEN function (see Fig. 2) and are executed in parallel. The ‘-’ entry for the constraint set $\{c_4, c_5, c_6, c_7\}$ indicates that the corresponding consistency check is still ongoing or has not been started up to now. Algorithm 3 uses the LOOKUP function to test the consistency of a constraint set.

Node-id	Constraint set	Inconsistent
1	$\{c_3, c_4, c_5, c_6, c_7\}$	<i>true</i>
1.1	$\{c_2, c_3, c_4, c_5, c_6, c_7\}$	<i>false</i>
1.1.1	$\{c_1, c_6, c_7\}$	<i>true</i>
1.2.1	$\{c_4, c_5, c_6, c_7\}$	-

The QXGEN function (see Algorithm 4) predicts future potentially relevant consistency checks needed by QX and activates individual consistency checking tasks in an asynchronous fashion using the ADDCC (*add consistency check*) function. The ADDCC function triggers an asynchronous service that is in charge of adding consistency checks (parameter of ADDCC) to a LOOKUP table and issuing the corresponding solver calls. The global parameter *lmax* is used define the maximum search depth of one activation of QXGEN.

In QXGEN, $|f(X)|$ denotes the number of constraints c_i in X (it is introduced due to the subset structure in C). Furthermore, $\text{SPLIT}(C, C_a, C_b)$ splits C at position $\lfloor \frac{|C|}{2} \rfloor$ if $|C| > 1$ or C_1 (the first element of C) at position $\lfloor \frac{|C_1|}{2} \rfloor$ if $|C| = 1$ and $|C_1| > 1$ into C_a and C_b . Otherwise, no split is needed (C_1 is a singleton).

The first inner condition of QXGEN ($|f(\delta)| > 0$) generates a consistency check if this is needed. A consistency check is needed, if $B\delta$ gets extended from C or a singleton C has been identified which then extends B . The function ADDCC is used to add consistency check tasks which can then be executed asynchronously in a parallelized fashion. Thus, a consistency check in the LOOKUP table can be easily identified by an ordered constraint set that has also been used as parameter of ADDCC, for example, $\text{ADDCC}(\{\{c_4, c_5\}, \{c_6, c_7\}\})$ results in the LOOKUP table entry $\{c_4, c_5, c_6, c_7\}$ which is internally represented with 4567.

Algorithm 4. QXGEN($C, B\delta, B, \delta, l$)
 $C = \{C_1..C_r\} \dots$ consideration set (subsets C_α)
 $B\delta = \{B\delta_1..B\delta_n\} \dots$ added knowledge (subsets $B\delta_\beta$)
 $B = \{B_1..B_o\} \dots$ background (subsets B_γ)
 $\delta = \{D_1..D_p\} \dots$ to be checked (subsets D_π)
 $l \dots$ current lookahead depth

```

1: if  $l < lmax$  then
2:   if  $|f(\delta)| > 0$  then
3:     ADDCC( $B\delta \cup B$ )
4:   end if
5:    $\{B\delta \cup B$  assumed consistent $\}$ 
6:   if  $|f(C)| = 1 \wedge |f(B\delta)| > 0$  then
7:     QXGEN( $B\delta, \emptyset, B \cup \{C_1\}, \{C_1\}, l + 1$ )
8:   else if  $|f(C)| > 1$  then
9:     SPLIT( $C, C_a, C_b$ )
10:    QXGEN( $C_a, C_b \cup B\delta, B, C_b, l + 1$ )
11:   end if
12:    $\{B\delta \cup B$  assumed inconsistent $\}$ 
13:   if  $|f(B\delta)| > 0 \wedge |f(\delta)| > 0$  then
14:     QXGEN( $\{B\delta_1\}, B\delta - \{B\delta_1\}, B, \emptyset, l + 1$ )
15:   end if
16: end if

```

If $B\delta \cup B$ is assumed to be *consistent*, additional elements from C have to be included such that an inconsistent state can be generated (which is needed for identifying a minimal conflict). This extension of $B\delta$ can be achieved by dividing C (if $|f(C)| > 1$, i.e., more than one constraint is contained in C) into two separate sets C_a and C_b and to add C_b to $B\delta$. If $|f(C)| = 1$, this singleton can be added to B which is responsible of collecting constraints that have been identified as being part of the minimal conflict. This is the case due to the already mentioned invariant property, i.e., $C \cup B\delta \cup B$ is inconsistent. If C contains only one constraint, i.e., C_1 is a singleton, it is part of the conflict set. If $B\delta \cup B$ is assumed to be *inconsistent*, it can be reduced and at least one conflict element will be identified in the previously added $B\delta_1$. If δ does not contain an element, no further recursive calls are needed since $B\delta - B\delta_1$ has already been checked.

The QXGEN function (Algorithm 4) is based on the idea of issuing recursive calls and adapting the parameters of the calls depending on the two possible situations 1) *consistent*($B\delta \cup B$) and 2) *inconsistent*($B\delta \cup B$). In Table 2, the different parameter settings are shown in terms of FOLLOW sets representing the settings of $C, B\delta, B$, and δ in the next activation of QXGEN.

Optimizations. To further improve the performance of QX, we have included mechanisms that help to identify irrelevant executions of consistency checks (see Table 3). If a consistency check has been completed, we are able to immediately decide whether some of the still ongoing or even not started ones can be canceled since they are not relevant anymore. For example, if the result of a consistency check is $\{c_3..c_7\}$ is *true* (see Fig. 2), the check $\{c_4..c_7\}$ does not have to be

Table 2. FOLLOW sets of the QXGEN function. Depending on the assumption about the consistency of $B\delta\cup B$, the follow-up activations of QXGEN have to be parameterized differently.

COND.	FOLLOW SETS			
	C	$B\delta$	B	δ
$ f(C) = 1$	$B\delta$	\emptyset	$B \cup \{C_1\}$	$\{C_1\}$
$ f(C) > 1$	C_a	$C_b \cup B\delta$	B	C_b
$ f(B\delta) > 0$	$\{B\delta_1\}$	$B\delta - \{B\delta_1\}$	B	\emptyset

executed anymore (see also Proposition 1) or has to be canceled since QUICKXPLAIN will not need this check (see Fig. 2). The deletion criteria for ongoing or even not started consistency checks can be pre-generated for $lmax$. An example for $lmax = 3$ is provided in Table 3.

Table 3. Optimizing the execution of consistency checks by detecting irrelevant ones. If the result of a specific check is known, LOOKUP (Algorithm 3) triggers the corresponding delete (*del*) operation. Nodes without associated consistency check are ignored.

NODE-ID	RESULT OF CONSISTENCY CHECK	
	<i>true</i>	<i>false</i>
1	<i>del</i> (1.2.x)	<i>del</i> (1.1.x)
1.1	<i>del</i> (1.1.2.x)	<i>del</i> (1.1.1.x)
1.2	<i>del</i> (1.2.2.x)	<i>del</i> (1.2.1.x)

In Table 3, the used *node-ids* are related to QXGEN instances, for example, in Fig. 2 the consistency check $\{c_{3..c7} = true\}$ has the *node-id* 1.

4 Analysis

QX Complexity. Assuming a splitting $k = \lfloor \frac{m}{2} \rfloor$ of $C = \{c_1..c_m\}$, the worst case time complexity of QUICKXPLAIN in terms of the number of consistency checks needed for calculating one minimal conflict is $2k \times \log_2(\frac{m}{k}) + 2k$ where k is the minimal conflict set size and m represents the underlying number of constraints [7]. Since consistency checks are the most time-consuming part of conflict detection, the runtime performance of the underlying algorithms must be optimized as much as possible.

QXGEN Complexity. The number (nc) of different possible minimal conflict sets that could be identified with QXGEN for an inconsistent constraint set C consisting of m constraints is represented by Formula 1.

$$nc(C) = \binom{m}{1} + \binom{m}{2} + \dots + \binom{m}{m} \tag{1}$$

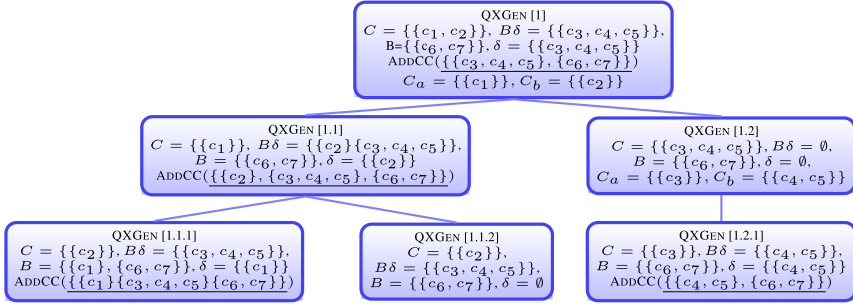


Fig. 2. QXGEN execution trace for $C = \{\{c_1..c_5\}\}$, $B\delta = \emptyset$, $B = \{\{c_6, c_7\}\}$, $\delta = \emptyset$, and $lmax = 3$. The consistency checks $\{c_3, c_4, c_5, c_6, c_7\}$ and $\{c_2, c_3, c_4, c_5, c_6, c_7\}$ (flattened list generated by ADDCC) can be used by the QUICKXPLAIN instance of Fig. 1. The QXGEN nodes $\{[1], [1.1], [1.1.2]\}$ represent the first part of the QUICKXPLAIN search path in Fig. 1.

The upper bound of the space complexity in terms of recursive QXGEN calls for $lmax = n$ is in the worst case $2^{n-1} - 1$. Due to the combinatorial explosion, only those solutions make sense that scale $lmax$ depending on the available computing cores (see the reported evaluation results).

If the non-parallelized version of QUICKXPLAIN is applied, only sequential consistency checks can be performed. The approach presented in this paper is more flexible since $\#processors$ consistency checks can be performed in parallel. Assuming a maximum QXGEN search depth of $lmax = 4$, the maximum number of generated consistency checks is $\frac{2^{lmax-1}}{2}$ due to the binary structure of the search tree, i.e., 4 in our example. Out of these 4 checks, a maximum of 3 will be relevant to QX, the remaining ones are irrelevant for identifying the conflict in the current QX session. Thus, the upper bound of relevant consistency checks generated by QXGEN is $lmax$, i.e., one per QXGEN search level (see the outer left search path in Fig. 2), the minimum number of relevant consistency checks is $\lceil \frac{lmax}{2} \rceil$, i.e., 2 in our example. Assuming $\#processors = 16$, our approach can theoretically achieve a performance boost of factor 3 since max. 3 relevant consistency checks can be performed in parallel. It is important to mention, that within the scope of these upper and lower bounds, a performance improvement due to the integration of QXGEN can be guaranteed independent of the underlying knowledge base.

Termination of QXGEN. If the parameter $lmax = n$, recursive calls of QXGen stop at level $n - 1$. In every recursive step, based on the inconsistency invariant between $C \cup B\delta \cup B$, either 1) C is reduced to C_a and $B\delta$ gets extended with C_b (if $B\delta \cup B$ is consistent) or to $B \cup C$ if C is a singleton, or 2) $B\delta$ is further reduced (if $B\delta \cup B$ is inconsistent). Obviously, in the second case, the constraints in C are not relevant anymore, since a conflict can already be found in $B\delta$ (which is inconsistent with B).

QX-conformance of QXGEN. QXGEN correctly predicts QX consistency checks. It follows exactly the criteria of QX. If $|f(C)| = 1$, i.e., C includes only one constraint c_α , $B\delta \cup B$ is consistent and - as a consequence of the inconsistency invariant - c_α is a conflict element and therefore has to be added to B . The issued consistency check is $B \cup C$ - if inconsistent, a conflict has already been identified. If $|f(C)| > 1$, $C_b \cup B\delta \cup B$ has to be checked, since $B\delta \cup B$ is consistent and by adding C_b we follow the goal of restoring the inconsistency of $B\delta \cup B$. Finally, if $B\delta \cup B$ is inconsistent, no check has to be issued since $B\delta - \{B\delta_1\} \cup B$ has already been checked in the previous *QXGen* call and obviously was considered consistent. Thus, QXGEN takes into account all QX states that can occur in the next step and exactly one of the generated consistency checks (if needed) will be relevant for QX. Finally, the generated consistency checks are irredundant since each check is only generated if new constraints from C are added to $B\delta$ or a new constraint (singleton C) is added to B .

Runtime Analysis. The following evaluations have been conducted on the basis of a *Java 8* based implementation of the parallelized QUICKXPLAIN (QX) version presented in this paper. For the implementation, we applied the *ForkJoin* framework for running parallel tasks in *Java*. This, while being less automatic than newer java implementations allowed us to fully control when threads are created and destroyed. For representing our test knowledge bases and conducting the corresponding consistency checks, we have used the SAT solving environment *SAT4j* (see www.sat4j.org). All experiments reported in the following have been conducted using an *Intel Xeon multi-core (16 cores) E5-2650 of 2.60 GHz computer and 64 GB of RAM* that supports hyperthreading simulation of 64 cores which would allow us to run up to $lmax = 6$. We have selected configuration knowledge bases (feature models) from the publicly available BETTY toolsuite [12], which allows for a systematic testing of different consistency checking and conflict detection approaches for knowledge bases. The knowledge base instances (represented as background knowledge B in QUICKXPLAIN) that have been selected for the purpose of our evaluation, range from 500 to 5.000 binary variables and also vary in terms of the number of included constraints (B 25-1.500 binary constraints). On the basis of these knowledge bases, we randomly generated requirements ($c_i \in C$) that covered 30%–50% of the variables included in the knowledge base. These requirements have been generated in such a way that conflict sets of different cardinalities could be analyzed (see Table 4). In order to avoid measuring biases due to side effects in thread execution, we repeated each evaluation setting $3\times$.

The results of our QXGEN performance analysis are summarized in Table 4. On an average, the runtime needed by standard QUICKXPLAIN ($lmax = 1$) to identify a preferred minimal conflict of cardinality 1 is $3.2\times$ higher compared to a parallelized solution based on *QXGen* ($lmax = 5$). In Table 4, each entry represents the average runtime in *msec* for all knowledge bases with a preferred conflict set of cardinality n , where the same set of knowledge bases has been evaluated for $lmax$ sizes 1–6 ($lmax = 1$ corresponds to the usage of standard QX without QXGEN integration). In this context, we have introduced an additional

baseline version $6(rd)$, where only a randomly selected set of QXGEN consistency checks has been evaluated. It can be observed that with an increasing $lmax$ the performance of QX increases. Starting with $lmax = 6$, a performance deterioration can be observed which can be explained by the fact that the number of pre-generated consistency checks starts to exceed the number of physically available processors. In the line of our algorithm analysis, the number of relevant consistency checks that can be performed with $lmax = 5$ is between 5 and 3. Taking into account the overheads for managing the parallelized consistency checks, the shown results support our theoretical analysis of QXGEN.

Table 4. Avg. runtime (in *msec*) of parallelized QX when determining minimal conflicts.

$lmax$	Conflict cardinality				
	1	2	4	8	16
1	103993.0	286135.4	11006.4	30995.3	177354.7
2	69887.3	193354.2	9528.0	26258.7	154093.7
3	49823.5	130657.5	11094.6	28639.6	154155.0
4	50042.8	136150.8	7577.0	19753.1	120992.0
5	32481.4	88963.7	7750.0	18975.6	98242.7
6	34946.2	94783.0	6367.6	17743.7	95816.3
6(rd)	105678.3	195987.5	112546.8	28676.1	179876.2

5 Conclusions

We have introduced a parallelized variant of the QUICKXPLAIN algorithm that is used for the determination of minimal conflict sets. Example applications are the model-based diagnosis of hardware designs and the diagnosis of inconsistent user requirements in configuration and recommender applications. Current approaches to the detection of minimal conflicts do not take into account the capabilities of multi-core architectures. Our parallelized variant of QUICKXPLAIN provides efficient conflict detection especially when dealing with large and complex knowledge bases.

Acknowledgements. This work has been partially funded by the EU FEDER program, the MINECO project OPHELIA (RTI2018-101204-B-C22); the TASOVA network (MCIU-AEI TIN2017-90644-REDT); and the Junta de Andalucía METAMORFOSIS project.

References

1. Bakker, R., Dikker, F.: Diagnosing and solving over-determined constraint satisfaction problems. In: 13th International Joint Conference on Artificial Intelligence (IJCAI 1993), Chambéry, France, pp. 276–281 (1993)

2. Bordeaux, L., Hamadi, Y., Samulowitz, H.: Experiments with massively parallel constraint solving. In: 21st International Joint Conference on Artificial Intelligence, pp. 443–448. Morgan Kaufmann Publishers, San Francisco (2009)
3. Felfernig, A., Burke, R.: Constraint-based recommender systems: technologies and research issues. In: ACM International Conference on Electronic Commerce (ICEC 2008), Innsbruck, Austria, pp. 17–26 (2008)
4. Gent, I., et al.: A review of literature on parallel constraint solving. *Theory Pract. Log. Program.* **18**(5–6), 725–758 (2018)
5. Hamadi, Y., Sais, L. (eds.): *Handbook of Parallel Constraint Reasoning*. Springer, Cham (2018). <https://doi.org/10.1007/978-3-319-63516-3>
6. Jannach, D., Schmitz, T., Shchekotykhin, K.: Parallelized hitting set computation for model-based diagnosis. In: 29th AAAI Conference on Artificial Intelligence, pp. 1503–1510. AAAI Press, Austin (2015)
7. Junker, U.: QuickXPlain: preferred explanations and relaxations for over-constrained problems. In: 19th National Conference on Artificial Intelligence, pp. 167–172. AAAI Press, San Jose (2004)
8. Marques-Silva, J., Heras, F., Janota, M., Previti, A., Belov, A.: On computing minimal correction subsets. In: 23rd International Joint Conference on Artificial Intelligence, Beijing, China, pp. 615–622 (2013)
9. Reiter, R.: A theory of diagnosis from first principles. *Artif. Intell.* **23**(1), 57–95 (1987)
10. Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.): *Recommender Systems Handbook*. Springer, Boston (2011). <https://doi.org/10.1007/978-0-387-85820-3>
11. Rossi, F., Venable, K., Walsh, T.: *A Short Introduction to Preferences: Between Artificial Intelligence and Social Choice*. Morgan & Claypool Publishers, San Rafael (2011)
12. Segura, S., Galindo, J., Benavides, D., Parejo, J., Ruiz-Cortés, A.: BeTTY: benchmarking and testing on the automated analysis of feature models. In: 6th International Workshop on Variability Modeling of Software-Intensive Systems, VaMoS 2012, pp. 63–71. ACM, New York (2012)
13. Stumptner, M.: An overview of knowledge-based configuration. *AI Commun.* **10**(2), 111–125 (1997)
14. Tsang, E.: *Foundations of Constraint Satisfaction*. Academic Press, London (1993)

Author Index

- Agoun, Juba 277
Aiello, Ivano 205
Andreasen, Troels 413
Antoine, Violaine 342
Appice, Annalisa 161
Atas, Muesluem 287
Atas, Müslüm 457
Azawi, Nessn H. 440
- Benavides, David 457
Biecek, Przemysław 45
Biesialska, Katarzyna 32
Biesialska, Magdalena 32
Bloice, Marcus D. 255
Bochicchio, Mario A. 235
Breskvar, Martin 383
Bulskov, Henrik 413
- Cacho Aboukhalil, Jeffry 318
Ceci, Michelangelo 362
Chernyavskiy, Alexander 90
Christiansen, Henning 440
Ciecierski, Konrad A. 192
- Dobrakowski, Adam Gabriel 45
Dong, Ning 214
Duricic, Tomislav 181
Džeroski, Sašo 362, 383
- Erdeniz, Seda Polat 287
- Fadjrimiratno, Muhammad Fikko 214
Falini, Antonella 161
Falkner, Andreas 405
Felfernig, Alexander 457
Ferilli, Stefano 308
- Galindo, Jose 457
Gao, Chao 372
Gelich, Anna 11
Gembariski, Paul Christoph 56
Gordea, Sergiu 102
- Grekow, Jacek 150
Guarascio, Massimo 448
- Hacid, Mohand-Saïd 277
Hajja, Ayman 245
Hatae, Yusuke 214
Helic, Denis 181
Heller, Philip 205
Hettinger, Lena 422
Holzinger, Andreas 255
Hotho, Andreas 422, 431
Hussain, Hussain 181
- Ilvovsky, Dmitry 90
Isaac, Antoine 102
- Jaworski, Wojciech 45
- Kahn, Amanda S. 205
Kersting, Kristian 362
Kiefer, Christoph 318
Klampfl, Lorenz 298
Kłopotek, Mieczysław Alojzy 352
Kłopotek, Robert Albert 352
Kobs, Konstantin 431
Kocev, Dragi 66, 331
Koko, Jonas 342
Kostek, Bozena 225
Kowald, Dominik 181
- Lachmayer, Roland 56
Lacic, Emanuel 181
Langenberg, Benedikt 318
Lardy, Romain 342
Lautenschlager, Florian 431
Lemmerich, Florian 79, 318
Lex, Elisabeth 181
Li, Xianghua 372
Liguori, Angelica 448
Liu, Jiming 21, 171, 372
Liu, Xingjian 372
Liu, Yang 21, 171

- Loglisci, Corrado 235
Lomuscio, Francesco 161
Longo, Antonella 235
- Malerba, Donato 161, 235
Manco, Giuseppe 448
Marciniak, Małgorzata 45
Masciari, Elio 113
Matsukawa, Tetsu 214
Mayer, Axel 318
Mazzia, Francesca 161
Mendes-Moreira, João 139
Mendes-Neves, Tiago 139
Mialon, Marie-Madeleine 342
Moscato, Vincenzo 113
Muškardin, Edi 267
Mykowiecka, Agnieszka 45
- Nilsson, Jørgen Fischer 413
- Paramita, Monica Lestari 102
Pedersen, Mikkel 440
Petković, Matej 362
Picariello, Antonio 113
Pill, Ingo 267
Plappert, Stefan 56
Popović, Radomir 79
Powell, Laurel 11
- Ras, Zbigniew W. 11
Redavid, Domenico 308
Ren, Jinfu 21
Ring, Markus 422
Ritacco, Ettore 448
Roth, Peter M. 255
Rybinski, Henryk 32
- Samer, Ralph 287
Saxena, Saketh 205
Schenner, Gottfried 405
Schlr, Daniel 422
Schörghuber, Alexander 405
Scicchitano, Francesco 448
Silva, Cristian Vidal 457
Sperli, Giancarlo 113
Steininger, Michael 431
Stepišnik, Tomaž 66, 331
Strohmaier, Markus 79
Suzuki, Einoshin 214
- Tamborrino, Cristiano 161
Tan, Qi 171
Teppan, Erich 395
Tkalčič, Marko 3
Tritscher, Julian 422
- Veissier, Isabelle 342
- Wagner, Nicolas 342
Willis, Justin 245
Wotawa, Franz 267, 298
- Yao, JingTao 123
- Zanker, Markus 395
Zaporowski, Szymon 225
Zappatore, Marco 235
Zehe, Albin 431
Zhang, Fan 372
Zhang, Yan 123
Zhou, Yue 123